Speech Processing with Linear and Neural Network Models

Tina-Louise Burrows

Cambridge University Engineering Department
Trumpington Street
Cambridge CB2 1PZ
England

This dissertation is submitted for consideration for the degree of Doctor of Philosophy at the University of Cambridge

Summary

This dissertation investigates some aspects of speech processing using linear models and single hidden layer neural networks. The study is divided into two parts which focus on speech modelling and speech classification respectively.

The first part of the dissertation examines linear and nonlinear vocal tract models for synthesising high quality speech with adjustable pitch. A source-filter framework for analysis and synthesis is used, in which the source is a representation of the glottal volume velocity waveform. Two families of linear model are considered, ARX (autoregressive with external input) and OE (output error). Their performance in estimating vocal tract transfer functions is compared on synthetic speech data, and the difference is explained in terms of the parameter estimation procedure, the frequency distribution of bias in the estimate and the assumptions about the spectrum of the noise in the vocal tract system. The noise spectrum for ARX models is shown to be perceptually significant for speech synthesis applications because it exploits auditory masking. Methods for improving poor quality syntheses from OE models are proposed. Nonlinear vocal tract models, implemented as feed-forward or recurrent neural networks, are investigated. Methods for initialising networks from linear models are developed. A modified recurrent architecture is introduced which permits initialisation from ARX models. The use of regularization, for imposing continuity between models of adjacent speech segments, and learning rate adaptation, for improving back-propagation training, are discussed. For synthesising real speech utterances, an audio tape demonstrates that ARX models produce the highest quality synthetic speech and that the quality is maintained when pitch modifications are applied.

The second part of the dissertation studies the operation of recurrent neural networks in classifying patterns of correlated feature vectors. Such patterns are typical of speech classification tasks. The operation of a hidden node with a recurrent connection is explained in terms of a decision boundary which changes position in feature space. The feedback is shown to delay switching from one class to another and to smooth output decisions for sequences of feature vectors from the same class. For networks trained with constant class targets, a sequence of feature vectors from the same class tends to drive the operation of hidden nodes into saturation. It is demonstrated that saturation defines limits on the position of the decision boundary resulting in context-sensitive and context-insensitive regions of the feature space. While saturation persists, it is shown that networks have reduced sensitivity to the order of presentation of feature vectors because movement of the decision boundary is inhibited. To improve this within-class sensitivity, training with ramp-like class targets is investigated. The operation of small recurrent networks is demonstrated for two tasks; classification of speech utterances into voiced and unvoiced segments, and classification of clockwise and anti-clockwise trajectories of vectors produced by two autoregressive processes.

Acknowledgements

I would like to thank everyone in the Fallside Lab for making my time in Cambridge an experience. In particular, I would like to mention Julian for his practical advice, Rob for all his help with the fiddly tape-recording, and Xtof for his patience with my faltering Spanish. Special thanks to my supervisor, Dr. Mahesan Niranjan, for his guidance, and to Dr. Ljung and Dr. Maciejowski for helpful discussions on system identification theory. The biggest thank-you of all goes to my sister, Tanya, for all her love and support, especially while writing up.

This work has been funded by the Science and Engineering Research Council with some useful top-ups from the Engineering Department and Queens' College.

Dedication

To Mum and Dad. Thank you for supporting me in all the mad things I do.

Declaration

This 54,000 word dissertation is entirely the result of my own work and includes nothing which is the outcome of work done in collaboration.

Tina-Louise Burrows
Queens' College
March 20, 1996

Contents

List of Figures

List of Tables

List of Notation

Abbreviations

AR	autoregressive model
ARX	autoregressive (AR) model with external input(X)
OE	output error model
LP	linear prediction model
CELP	code excited linear prediction
FNN	feedforward neural network
RNN	recurrent neural network
HMM	hidden Markov model
ETFE	empirical transfer function estimate
SNR	signal-to-noise ratio
MSE	mean squared error

Symbol Definitions

$R(z), R(q)$	lip radiation characteristic
$P(z), P(q)$	pre-emphasis filter
$\hat{H}(z)$, $H(z)$, $H(q)$	vocal tract transfer function
$H(e^{j\omega}, \boldsymbol{\theta})$	vocal tract frequeny response
$\hat{\hat{H}}(q)$	Empirical Transfer Function Estimate
$Q(\omega, \boldsymbol{\theta})$	frequency bias function for transfer function estimate
$N(q)$, $N(e^{j\omega}, \boldsymbol{\theta})$	model noise and corresponding specturm
$\Phi_{\mathrm{ER}}(\omega, \boldsymbol{\theta})$	spectrum of synthesis error
$L(t)$, $dL(t)$	laryngograph signal and first difference
$x(t)$, $dx(t)$	glottal volume velocity wave model and first difference
$X(e^{j\omega})$	spectrum of input waveform ($x(t)$ or $dx(t)$)
$y(t)$	speech waveform
$Y(e^{j\omega})$	speech spectrum
$\hat{y}_s(t)$	model synthesis
$\hat{y}_p(t)$	model prediction
$\boldsymbol{x}(t)$	network input vector
$\boldsymbol{h}(t)$	hidden node output
$\hat{y}(t)$	network output (prediction or synthesis)
$\boldsymbol{U}$, $\boldsymbol{V}$, $\boldsymbol{W}$	network weights (output, input and feedback)
q^{-1}	backward shift operator, $q^{-1}x(t) = x(t-1)$
z, z^{-1}	z-transforms
$(.)^{\mathrm{T}}$	denotes matrix transpose
$\|\|.\|\|$	denotes Euclidean norm

Chapter 1

Introduction

"In the beginning was the Word, and the Word was with God, and the Word was God."
St. John 1:1.

Speech is the acoustic realisation of a language. Our knowledge of how we speak, hear, recognise and understand a language can be increased by studying the speech signal and attempting to model these functions. This thesis investigates some issues for speech processing with linear and neural network models. In this chapter, the speech production mechanism is described and some of the terminology applicable to speech processing is introduced. Previous relevant research in speech processing with linear and neural network models is reviewed and the research presented in this thesis is outlined.

1.1 The Speech Production Mechanism

The mechanism for speech production, shown in Fig. 1.1, consists of the trachea, vocal cords, tongue, vocal tract (oral and nasal cavities), lips, teeth and nostrils, in addition to the diaphragm and lungs. A speech utterance begins as an air stream or volume velocity wave from the lungs, which travels along the trachea and vocal tract to be radiated as an acoustic pressure waveform from the lips or the lips and nostrils.

Speech is classified as *voiced* or *unvoiced*, depending on the nature of the excitation of the vocal tract. For voiced phones, the excitation of the vocal tract originates at the glottis and is by the periodic vibration of the vocal cords. The frequency of vibration, or *pitch*, is controlled by the tension in the vocal cords and the air pressure from the lungs. Typical pitch values lie in the range 50-500Hz for adults, and can rise to 1000Hz in children. Due to its periodic nature, the spectrum of voiced excitation contains discrete components at harmonics of the pitch frequency. For unvoiced sounds, the excitation is due to turbulence generated by airflow past a narrow constriction and tends to be random in nature, with a flat, continuous spectrum. The noise is known as aspiration if the constriction is at the glottis and frication if it occurs at some point along the vocal tract. Mixed excitation

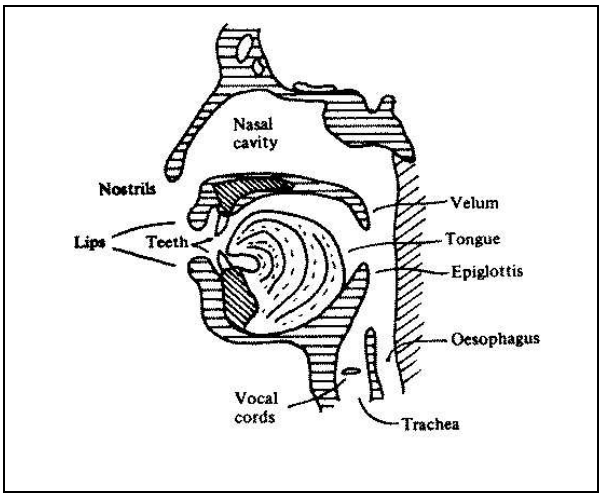


Figure 1.1: Speech Production Mechanism.

is also possible for the class of sounds known as voiced fricatives, in which turbulent excitation is amplitude modulated periodically by the vibration of the vocal cords.

The acoustic signal can be represented by a transcription of phonemes, which are the smallest units which convey linguistic meaning of a language. The actual sounds which are produced in speaking a string of target phonemes are called *phones*. Each phone of an utterance corresponds to a segment of the acoustic waveform which has a characteristic time-varying vibratory pattern. Vibratory patterns are superimposed on the air stream by the vibration of the vocal cords and resonance of the vocal tract. The resonant properties of the vocal tract are modified by changing the position of the articulators (the lips, tongue, jaw and velum, shown in Fig. 1.1.) Due to the physical constraints of the vocal tract, the positions of the articulators can only change slowly with time and individual realisations of a phone are strongly influenced by previous and future phones in an utterance. This phenomenon is known as *co-articulation* and is important for both accurate speech recognition and natural sounding speech synthesis.

Due to the slowly time-varying nature of the acoustic waveform for each phone, the resulting spectrum of the speech varies with time. The time variability of the spectrum is captured by calculating the spectrum of overlapping short-time segments of the acoustic waveform and is displayed using a spectrogram. A spectrogram plots the frequency of successive short-time spectra using the intensity of the plot to indicate the energy of the frequency components at a particular instant. Most of the energy in the speech spectrum is between 80-8000 Hz. Intelligibility tests on band-pass filtered speech show that intelligibility is not impaired when speech is low-pass filtered to remove all frequencies above 5kHz (French & Steinberg 1947, Klatt 1980). This permits a lower sampling rate of 10kHz. Within the frequency range 0-5kHz, the vocal tract for voiced phones typically has 4-5 resonant frequencies (Klatt 1980) which are called *formants*. Formants are visible as dark horizontal bands on a spectrogram. Examples of wideband and narrowband spectrograms for the utterance ‘Belgium’ are shown in Fig. 1.2. Wideband and narrowband spectrograms represent a tradeoff between time and frequency resolution of the spectrum. Narrowband spectorgrams use short-time speech segments of a couple of pitch periods in duration. The resulting spectrogram has high frequency resolution (y axis) and individual pitch harmonics appear as closely spaced horizontal bands, as illustrated in Fig. 1.2(b). However, time resolution is poor and rapid formant transitions are averaged over time. For the class of sounds called stops, the 'b' in 'Belgium' for example, the vocal tract becomes completely occluded by the tongue or lips for part of the utterance. Rapid movement of the articulators to release the occlusion, which may be accompanied by a burst of noise, gives rise to sounds that are short in duration and highly transient in nature. The time-variability in the spectrum of such phones may not be accurately represented by a narrowband spectrogram. Wideband spectrograms use short-time segments of roughly one pitch period in duration and give much better time resolution at the expense of frequency resolution. For voiced speech, vertical striations at the pitch period are visible, as illustrated in Fig. 1.2(c).

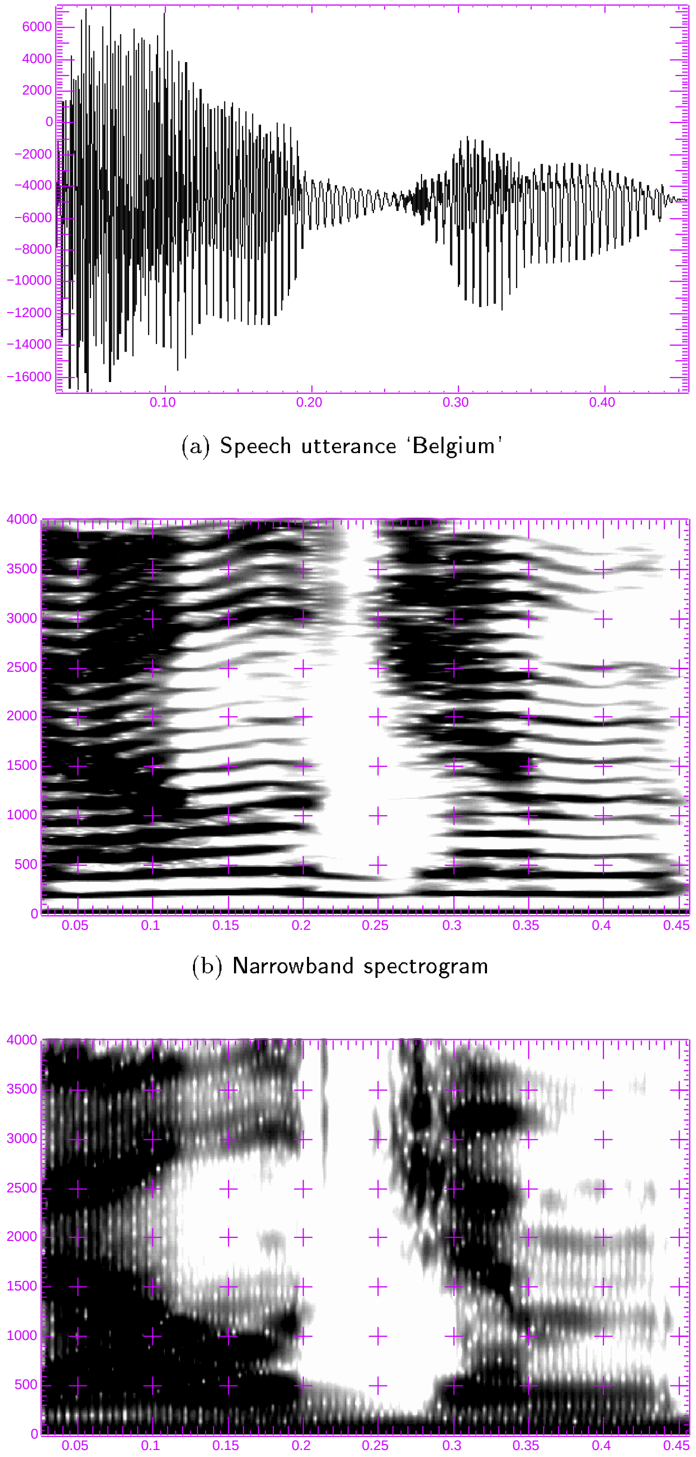
During nasals, such as 'm' or 'n', the air flow is diverted into the nasal cavity by the lowering of the velum. With the lips closed, the nasal cavity forms the principal resonant path which determines the formants and the vocal tract acts as a closed side-branch which introduces an anti-resonance (spectral valley) into the spectrum. In nasalised vowels, both the nasal and oral cavities are open and sound is radiated from the lips and nostrils simultaneously. The main resonances are due to the oral cavity, which determines the location of the formants, and the nasal branch is considered as the side-branch.

On reaching the lips and nostrils, the effect of directional sound propagation from these apertures is to convert the volume velocity wave into an acoustic pressure waveform which radiates away from the head. The pressure wave measured directly in front of the head is proportional to the time derivative of the resultant volume velocity wave from the lips and nostrils, and is inversely proportional to the distance from the lips (Fant 1960). The radiation effect can be approximated as that of radiation from a circular aperture in a sphere or infinite plane (Flanagan 1972) and the amplitude spectrum of the resultant acoustic waveform is approximately modified by $+6dB/$octave when compared to that of the volume velocity wave at the end of the vocal tract.

Additional features which add intelligibility, meaning and naturalness to speech are *stress* and, over longer phrasal durations, *prosody*. In addition to pitch, duration and intensity (loudness) constitute the parameters of stress and prosody which are used to emphasise important acoustic events and break speech up into meaningful units. At a higher phrasal level, specific prosodic patterns can also convey emotion and attitude.

1.2 Speech Processing

The two areas of speech processing considered in this thesis are signal modelling (for speech synthesis) and signal classification (for speech recognition). In modelling the speech signal, the aim is to parametrize speech waveforms in such a way that they can be stored



(a) Speech utterance 'Belgium'

(b) Narrowband spectrogram

(c) Wideband spectrogram

Figure 1.2: Typical speech waveform and spectrograms. For spectrograms, horizontal axis shows time in seconds, vertical axis shows frequency in Hz.

efficiently and reproduced (synthesised) at a later date. Parameters for models can be found by performing a time or frequency domain match between the original speech signal and that generated by the model.

In classification, models are developed to assign class labels to segments of the acoustic signal based on the distinguishing features of a parametric representation of each segment. In speech recognition, for example, the class labels are linguistic units of the language such as phones, diphones or triphones. The linguistic units can form the input for higher level natural language processing, in which syntactic and semantic constraints on possible linguistic sequences are applied and the meaning of the intended utterance extracted. Lower level classes are also possible, such as classifying the speech signal into voiced and unvoiced segments.

Speech processing typically involves the use or calculation of a parametric representation of acoustic waveforms. Speech signals are non-stationary and when processing long utterances, a time-varying parametric representation is needed. A quasi-stationary approach is usually adopted, in which an utterance is divided into a sequence of overlapping segments and assumed to be stationary for the duration of each segment. Since the speech production mechanism can change only slowly with time, parametric representations of adjacent segments of speech show a high degree of correlation. For modelling acoustic waveforms, this implies continuity in the values of model parameters for adjacent segments. For speech classification, it implies that the class label assigned to a particular feature vector is dependent on the context in which that feature vector occurs in an input sequence (context-dependent classification). Exploiting the correlation between segments of speech is highly beneficial for speech processing applications and is a way of representing co-articulation effects.

1.2.1 Review of Research in Modelling Speech Signals

The most widely used technique for speech analysis is linear prediction analysis (Makhoul 1975, Markel & Gray 1976), and forms the basis of most speech coding systems, such as vocoders (Markel & Gray 1974), CELP (code-excited linear prediction) coders (Schroeder & Atal 1985), multi-pulse coders (Atal & Remde 1982) and a host of variants which differ in the nature of the excitation of the linear prediction model at the decoder. The popularity of linear prediction is due to ease of analysis and implementation and low computational requirements. An alternative approach is to model the transfer function of the vocal tract system (vocal tract modelling). ARX (autoregressive with external input) (Lobo & Ainsworth 1992, Fujisaki & Ljungqvist 1987), OE (output error) (Wang, Guan & Fujisaki 1990) and state-space (Morikawa & Fujisaki 1984) parametrizations for the vocal tract filter have been used and differ in their underlying structure of the model and the nature of the error which is minimised in the parameter estimation procedure.

Modelling the vocal tract transfer function directly allows inclusion of zeros in the model and has been shown to improve prediction gain even for a simple impulse (Fujisaki & Ljungqvist 1987), or multi-pulse excitation (Singhal & Atal 1983). The use of a more realistic representation of the vocal tract excitation, based on glottal volume velocity wave pulse models, has been shown to improve the prediction gain by 2-10dB when compared with linear prediction analysis (Fujisaki & Ljungqvist 1986, Thomson 1992, Hedelin 1984), and gives improved naturalness of synthetic speech generated from both formant synthesisers and vocal tract models (Holmes 1973, Rosenberg 1971, Fujisaki & Ljungqvist 1986). Alternative approaches to modelling the excitation signal include linear and nonlinear inverse filtering techniques (Alku 1992, Milenkovic 1986, Denzler, Kompe, Kießling, Niemann & Nöth 1993), incorporating an all-zero model directly in the vocal tract transfer function (Mathews, Miller & David 1961, Funaki & Mitome 1990) or incorporating a more general function-based model of the excitation within the vocal tract transfer function (Thomson 1992, Cheng & O'Shaughnessy 1989). Speech coding systems using ARX models and a pulse-based excitation have been shown to give improved naturalness and prediction performance over linear prediction based coders (Hedelin 1984, Cheng & O'Shaughnessy 1993).

Speech coders require a parameter estimation procedure that is robust to the effects of noise. In the speech enhancement work by Lim & Oppenheim (1978) and Hansen & Clements (1994), MAP estimation was used to improve the estimation of linear prediction parameters in noisy environments. The correlation between the models of adjacent segments was exploited by using the model parameters from previous segments as initial estimates of the parameters for the current segment. Using a Bayesian framework to calculate model parameters, prior assumptions about the expected values of the parameters can be incorporated into the estimation procedure. Saleh, Niranjan & Fitzgerald (1994) have used this approach for linear prediction analysis, to obtain smoothed estimates of the formant tracks of noisy speech utterances.

There is experimental and theoretical evidence that the speech production mechanism is nonlinear (Teager & Teager 1990). Nonlinearities in the speech data are caused by rapid transitions between and during phones, especially plosives where there is occlusion of the vocal tract, and by turbulent excitation during unvoiced segments. Glottal opening and closure during the pitch periods of voiced speech causes coupling at the back of the throat which introduces additional energy loss. Linear models of the vocal tract system have a limited performance because they may not capture the structure of the data or the underlying system dynamics. The application of nonlinear models to the prediction of speech has shown 2–3dB improvement in prediction gain over linear models (Tishby 1990, Townshend 1991, Wu & Fallside 1992). Tishby (1990) and Lowe & Webb (1989) have demonstrated the ability of neural network based nonlinear models to generate limit cycles which capture the pitch of voiced speech. Recurrent neural networks, in which feedback around layers is introduced, have a greater ability to model the dynamics of a signal than feed-forward networks (Back & Tsoi 1991*b*), and for speech data, have been shown to give improved prediction gains over both linear and feed-forward networks with either the same number of parameters or the same predictive order (Wu & Fallside 1992).

Feed-forward and recurrent networks have been used in speech coding applications as vocal tract models. Wu & Fallside (1994) used an excitation generated by a code-excited neural network and Burrows (1992) used a CELP-like excitation.

Models of speech waveforms are of potential use in text-to-speech synthesis systems. In existing text-to-speech synthesis systems, synthetic speech is generated by parallel formant synthesisers (Klatt 1980, Holmes 1973), linear prediction models (Allen, Hunnicutt & Klatt 1987) or by direct manipulation of acoustic waveforms using overlap-add procedures (Charpentier & Stella 1986, Hamon, Moulines & Charpentiers 1989, Moulines & Charpentier 1990). Although time-domain overlap-add techniques generate high quality synthetic speech, they require actual acoustic waveforms to be stored. For data compression, a model based approach to speech synthesis is more efficient. The poor quality synthetic speech generated by linear prediction models, and the degradation in quality caused by manipulation of the pitch of the synthetic speech from that of the original, inherently limits the naturalness and flexibility of text-to-speech systems based on such models. Models producing more natural sounding synthetic speech, for which the pitch can be altered without loss of quality, would be more suitable. Artificial neural networks have been used in text-to-speech systems to provide text-to-phoneme mappings (Sejnowski & Rosenberg 1986), and to generate the control parameters to drive the speech synthesiser (Howard & Breen 1989, Tuerk & Robinson 1993). Neural networks have also been incorporated into articulatory synthesisers, to provide a nonlinear mapping between acoustic and articulatory parameters (Xue, Hu & Milenkovic 1990).

1.2.2 Review of Research in Classification with Neural Networks

The interest in applying neural networks to pattern classification tasks is due to their many advantages over conventional methods. Their interpolative capabilities in the case where there is insufficient training data are an advantage over simple methods such as clustering. The ability to perform nonlinear discriminant analysis in a possibly higher dimensional space (Gallinari, Thiria, Badran & Fogelman Soulié 1991, Webb & Lowe 1990) is an advantage over linear discriminant analysis techniques and allows the formation of disjoint class boundaries. No assumptions about the statistical distributions or independence of feature vectors are needed and training to minimise a mean-squared error criterion is discriminative, so models are trained to suppress incorrect classes while modelling correct classes as accurately as possible. The parallel architecture of neural networks allows efficient implementation in hardware.

Feed-forward networks (or multilayer perceptrons) form a static mapping between input feature vectors and class targets, where classes are typically determined by a 1-out-of-M encoding at the network output. Under certain assumptions[1], the outputs of feed-

[1] The interpretation of the outputs of a feed-forward network as posterior probabilities of class membership is subject to the assumption that the network architecture is sufficiently complex to model the optimal Bayes discriminant functions and that networks are trained to minimise a mean-squared or cross-entropy error function for a 1-out-of-M output encoding. The interpretation only holds at the global minimum of

forward networks have been shown to estimate the posterior probability of class membership (Richard & Lippman 1991, Ruck, Rogers, Kabrisky, Oxley & Suter 1990, Wan 1990). The outputs can be constrained to sum to 1 by replacing the sigmoid nonlinearities by the softmax function (Bridle 1989).

The disadvantage of the feed-forward architecture for classification of speech patterns is that the static mapping between input and output cannot account for local context of feature vectors and all class decisions are independent. Several approaches to incorporating local context of feature vectors into feed-forward networks include augmenting the feature vector with a parametrization of neighbouring feature vectors (Robinson 1994) and presenting the network with an input window containing several adjacent feature vectors (Bourlard & Wellekens 1990). In time-delay neural networks (Waibel, Hanazawa, Hinton, Shikano & Lang 1989), activations of hidden units are calculated from the current input and multiple delayed outputs of the preceding hidden layer. An alternative approach used in linked predictive neural networks (Tebelskis & Waibel 1990) and neural prediction models (Iso & Watanabe 1990) is to use the feed-forward network as a pattern predictor to predict the sequence of feature vectors within the current class.

The context of previous class decisions can be incorporated by inclusion of recurrent connections in the neural network architecture. Several recurrent architectures have been proposed for speech recognition applications and include: recurrent connections from output to input (Bourlard & Wellekens 1990, Jordan 1986); self-loops around hidden and output units (Watrous & Shastri 1987); fully recurrent architectures with recurrent connections between hidden units of the same layer (Elman 1990, Robinson & Fallside 1991). The fully recurrent architecture can be interpreted in terms of state-space equations from control theory (Robinson & Fallside 1987).

The outputs of fully recurrent neural networks have been shown to approximate the posterior probability of class membership conditional on the entire sequence of input feature vectors up to the current time (Santini & Del Bimbo 1995). This interpretation follows from the probability estimation proofs for feed-forward networks by considering the recurrent architecture as a deep feed-forward network obtained by unfolding the recurrent architecture in time to form a multilayer feed-forward network with a hidden layer for each time step (Renals, Morgan, Bourlard, Cohen & Franco 1994).

For speech recognition purposes, feed-forward and recurrent neural networks provide no means of encoding the time-variability of speech pronunciation and speaker rate. Some form of dynamic time warping (Ney 1984) is needed to derive the optimal segmentation (class sequence) for an utterance, given the class probabilities estimated by the neural networks (Robinson & Fallside 1991). The dynamic time warping algorithm can be implemented as a neural network, a Viterbi net (Lippmann & Gold 1987), but there is no particular advantage to this implementation.

State-of-the-art speech recognition systems based on Hidden Markov Models (HMM)

the error criterion.

implicitly incorporate the optimal segmentation of an utterance in the recognition stage by selecting a model sequence which maximises the likelihood of generating the observed sequence of feature vectors from those models. The disadvantages of using HMMs are the need to make assumptions about statistical distributions and the fact that training is not discriminative. In addition, it is assumed that feature vectors are independent. Several researchers, including Bridle (1990),Niles (1991) and Bourlard & Wellekens (1990), have outlined direct equivalences between HMMs and recurrent network architectures and used these to train HMMs discriminatively. To exploit the discriminative training of neural networks while retaining the segmentation capabilities of HMMs, hybrid HMM-neural net approaches have been adopted. In these systems, the posterior probabilities estimated by network outputs are scaled to form likelihoods which are used as emission probabilities for HMM states. Both feed-forward (Bourlard & Wellekens 1990, Bourlard 1991, Renals, Hochberg & Robinson 1994) and fully recurrent architectures (Robinson 1994) have been used successfully. Bourlard (1991) has also shown how feed-forward networks can be used to estimate the transition probabilities for tri-phone HMMs. More recent research is concentrating on the development of context-dependent recurrent neural networks which are trained to estimate context-dependent class probabilities directly (Kershaw, Hochberg & Robinson 1995).

A disadvantage of using neural networks for pattern classification tasks is that target functions for each class are required in training. The choice of target function can influence the training and generalization of the network (Etemad 1993). For classification of continuous speech patterns, the need to specify class targets is a disadvantage because the segmentation of the data may not be known, and it is difficult to determine a suitable target. Typically, constant class targets are used. In speech recognition, it is intuitive to assume that confidence in a class decision accumulates over the duration of a class until a point is reached when a decision can be made. A suitable target function might reflect the increasing confidence in a decision. Simple ramp targets (Watrous & Shastri 1987), and target functions based on normalised dissimilarity functions (Hanes, Ahalt & Krishnamurthy 1994, Watrous, Ladendorf & Kuhn 1990) have already been used to train recurrent networks for phone discrimination tasks.

1.3 Outline of Thesis

1.3.1 Part I - Vocal Tract Modelling

Part I considers the problem of modelling an acoustic speech signal using short-time models of the vocal tract system. The aim is to generate natural sounding synthetic speech to which pitch modifications can be applied without appreciable distortion.

In chapter 2, different approaches to modelling the speech signal are outlined and the conventional linear prediction framework for speech synthesis described. Some of the limitations of such a system are highlighted and several approaches to overcoming these limitations are reviewed. The linear prediction framework is used as a baseline for comparing the performance of the alternative frameworks developed in chapters 3 and 4.

In chapter 3, a linear vocal tract modelling framework is set up, using two black-box parametrizations for the vocal tract filter, the ARX (autoregressive with external input) and the OE (output error) model. Using synthetic speech data, their performance in estimating the vocal tract transfer function is compared, and explained using a spectral interpretation of the prediction error criterion used in calculating model parameters. In the processing and synthesis of real speech utterances, the superior quality and pitch manipulation capabilities of ARX models are demonstrated.

Chapter 4 considers nonlinear models of the vocal tract system, which are implemented as single hidden layer neural networks, of either feed-forward or recurrent architecture. Issues of specific relevance to the short-time processing of a signal using neural network models are discussed and methods of initialising the weights of networks from linear black-box and state-space models are introduced. A modified recurrent architecture is developed to facilitate such linear initialisation. The use of regularization and learning rate adaptation for improving the performance of individual models is also considered.

Evaluation of the performance of different types of model is dependent on a subjective assessment of the perceptual quality of the synthetic speech produced. To enable the reader to verify the improved quality and pitch manipulation capabilities reported in this work, an audio tape of the syntheses of several utterances by different model types is included with this thesis. The format of the demonstration is described in Appendix B.

1.3.2 Part II - Classification of Speech Patterns

Chapter 5 presents a study of the operation of single hidden layer recurrent networks trained to classify time-varying patterns of feature vectors, such as those typically generated from acoustic speech data. Several characteristics of the operation are outlined; context-insensitive regions of the input feature space, delay in switching between classes, smoothing of output decisions, limited sensitivity to ordering in sequences of feature vectors from the same class (within-class context). These characteristics originate from saturation of hidden units, which is a result of training with fixed class targets. Experiments using synthetic data generated by vector AR processes, and speech data obtained from the TIMIT database (Garofolo 1988), are presented to illustrate these observations, and to assess their relevance for speech processing with recurrent neural networks. In addition, they are used to explore the potential that training recurrent networks on non-constant class targets has for improving the sensitivity of these networks to within-class context.

Chapter 6 presents the conclusions of this dissertation and some suggestions for future study.

1.4 Publications

Some preliminary work of relevance to chapter 4 can be found in (Burrows & Niranjan 1993). Initial work on vocal tract modelling with recurrent neural networks was presented at the 1995 International Conference on Acoustics, Speech and Signal Processing (Burrows & Niranjan 1995). Some aspects of chapter 5 were presented at the 1994 Conference on Neural Networks for Signal Processing (Burrows & Niranjan 1994).

Part I

Vocal Tract Modelling

Chapter 2

Modelling the Speech Signal

2.1 Introduction

Speech waveforms are rich in information but are highly redundant in structure. Storage of acoustic data is therefore inefficient and a more compact parametric representation of the information conveyed by the signal is desirable. An ideal model should exploit the redundancy in the speech signal to give data compression while capturing the distinguishing features of the waveform and the underlying dynamics of the production mechanism. For speech coding and synthesis applications, the ability to regenerate the original speech waveform from the model is also necessary.

The acoustic speech waveform varies slowly with time as different sounds are produced so the frequency properties of the signal are constantly changing. A time-varying model of the waveforms is needed for which the model parameters are continuously updated at a suitable rate. Typically, a short-time analysis is used, in which the speech waveform is divided into a sequence of overlapping segments of about 20ms in duration, and a new set of model parameters calculated for each segment. Since the articulators move relatively slowly, the vocal tract resonances remain fairly constant for durations of about 10ms (Linggard 1985) which permits an update rate (frame rate) of 10ms. Even the fastest transitions in plosives can be captured relatively well by an update rate of 5ms.

Two approaches to developing a model are *articulatory modelling* and *acoustic modelling*. The articulatory modelling approach aims to represent the vocal tract and movement of the articulators in as much physiological detail as possible and assumes that a similar underlying system will generate a similar output. Articulatory models have the potential for good reproduction from simple control signals and can reproduce all the perceptually relevant effects of real speech, such as co-articulation (Rubin & Baer 1981). However, the dimensions of the vocal tract and a detailed analysis of the movement of the articulators are needed. Such information is difficult to obtain and often requires intrusive measurement techniques. The acoustic modelling approach models the speech waveform directly in either the time or frequency domain. The models are easy to construct because only the speech waveform is required, which is easily obtained using a microphone. An exact match of the waveform or spectrum is not needed for perceptually good synthesis and events which are not perceptually relevant need not be modelled.

The most popular technique for speech modelling applications, such as speech coding and speech synthesis, is the time-domain acoustic modelling method known as linear prediction (LP). After briefly describing the frequency and time domain approaches to acoustic modelling, the remainder of this chapter reviews the LP approach to speech analysis and some of the improvements to the source-filter framework used to synthesise speech from LP models. LP synthesis of speech is used in chapters 3 and 4 as a baseline for comparison of the quality of synthetic speech from black-box linear models and neural network models of the vocal tract.

2.2 Acoustic Modelling

The acoustic modelling approach relies on the acoustic theory of speech production (Fant 1960). This theory regards the speech waveform as the output of a resonant network (or vocal tract filter) which is excited by one or more sound sources at the glottis. It assumes that the voice source, vocal tract and radiation effects can be modelled linearly and non-interactively in a source-filter arrangement, Fig. 2.1. The assumption that the source and filter can be modelled separately is due to the fact that the frequency of vibration of the vocal cords is determined by their mass and tension, and is relatively independent of the resonant frequencies of the vocal tract. The assumption is less valid at low frequencies because nonlinear coupling may cause damping of the first formant. For unvoiced speech, the assumption that excitation occurs at the glottis is rarely valid, since excitation is by turbulence generated at constrictions, usually within the vocal tract itself.

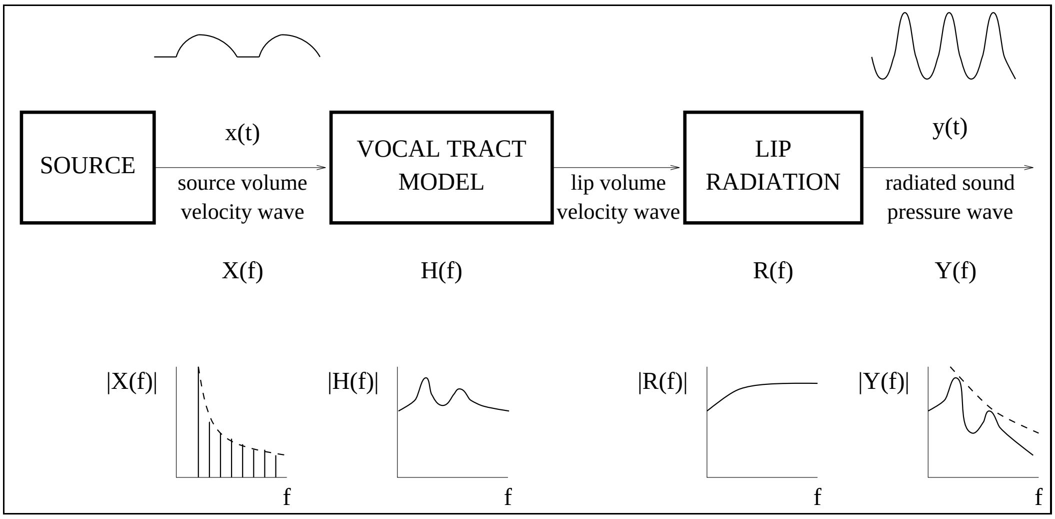


Figure 2.1: The acoustic theory of speech production.

The vocal tract system is characterised in the frequency domain by a *transfer function*

$$\frac{Y(f)}{X(f)} = H(f)R(f)$$

The transfer function gives the ratio of the spectrum of the pressure wave in the sound field, $Y(f)$, at some fixed distance from the lips, to that of the volume velocity wave (or supra-glottal pressure wave), $X(f)$, at the source. The transfer function is composed of the vocal tract transfer function, $H(f)$ and the lip radiation characteristic, $R(f)$. $H(f)$ relates the combined volume velocity wave at the lips and nostrils to the volume velocity wave at the source. $R(f)$ represents the radiation effect at the lips and nostrils. The spectrum of the resultant acoustic speech waveform, $Y(f)$, is given by

$$Y(f) = X(f)H(f)R(f)$$

Acoustic modelling techniques should aim to match the spectrum of the sound wave because the human ear detects the frequency spectra of sounds, rather than the actual time waveforms. The basilar membrane performs a short-time frequency transform of the sound pressure wave and presents this information to the brain via the auditory nerve. Perceptually relevant features are clearly detected in the short-time spectrum of a speech waveform. As the phase of the spectrum carries little information, models which generate perceptually good synthetic speech can be achieved by modelling the amplitude spectrum only. There are two techniques for acoustic modelling of speech waveforms, time-domain and frequency domain techniques. Time-domain techniques rely on the assumption that a good match to the speech waveform in the time domain will give a good spectral match while frequency domain techniques aim to model the salient features of the speech amplitude spectrum directly.

2.2.1 Frequency Domain Acoustic Modelling

Frequency domain techniques use a resonant network with a suitable transfer function to represent the vocal tract system. The resonant network is implemented as a transmission line or by channel or formant synthesisers. The transmission line approach exploits the analogy between the propagation of plane sound waves in an acoustic tube and the propagation of current and voltage waves along a transmission line. The source, lip radiation and nasal cavity are represented as lumped impedances which form a transmission line representation of the vocal tract system with the required resonant properties. This method is almost a hybrid between articulatory and acoustic modelling because the vocal tract is represented by a sequence of area functions and therefore has a physical interpretation.

In channel synthesisers, the resonant network is formed by 10-20 bandpass filters with fixed bandwidths and resonant frequencies which are connected in parallel and excited by an impulse train or white noise. Individual amplitude controls are adjusted so that the resultant sum of contributions from each channel gives a spectrum with 3-5 resonant peaks.

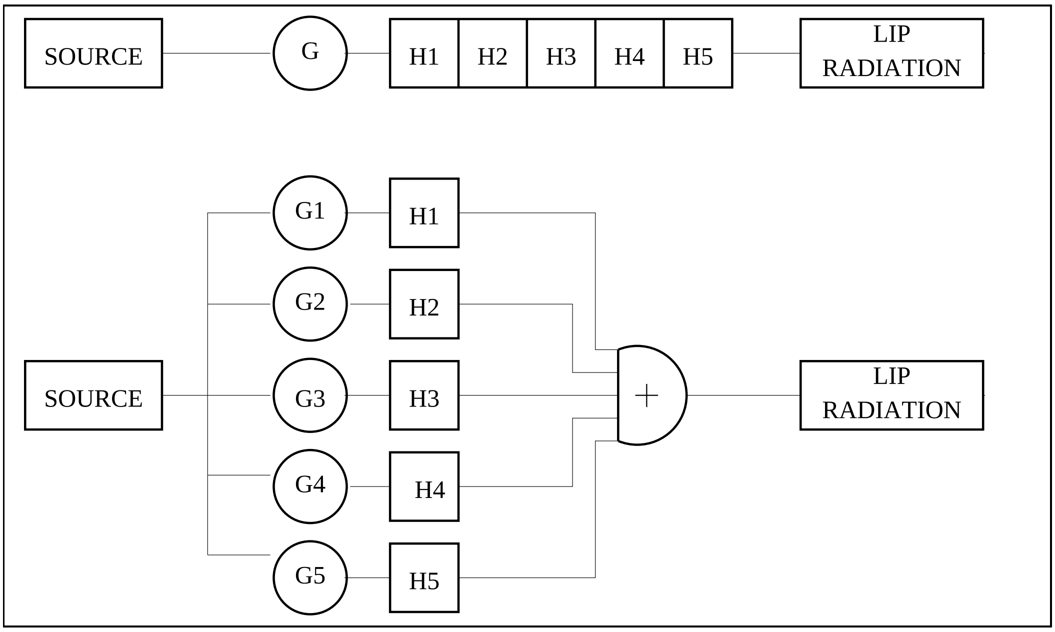


Figure 2.2: Cascade and Parallel Formant Synthesisers.

In formant synthesisers, each formant of the spectrum is modelled directly by a low-pass or bandpass resonant filter of a suitable bandwidth and resonant frequency equal to that of the required formant. An additional correction factor may be included to account for the effect of formants at higher frequencies which are not modelled directly. Formant synthesisers can be realised as either a serial (cascade) or parallel combination of resonators, as shown in Fig. 2.2. The serial formant synthesiser has a transfer function which resembles that of the vocal tract and requires fewer control signals. It performs better on non-nasal voiced sounds. However, it has the disadvantage that zeros in the transfer function have to be included as a separate resonant factor and the number of zeros is thus fixed. In addition, since a single amplitude control is used, careful adjustment of the individual bandwidths is necessary. In contrast, the parallel architecture is good for fricatives, nasals and stops since the individual amplitude controls allow zeros to be introduced into the transfer function by incomplete cancellation of the residues of the parallel factors. This has the disadvantage that the positions of anti-resonances are not immediately obvious from the transfer function. Phase cancellation between the resonant frequencies can also introduce unwanted anti-resonances into the spectrum. Alternating the sign of the amplitude controls is a possible solution. The individual amplitude controls give a higher control data rate than the serial implementation, but this can be reduced by using variable frequency resonators of fixed bandwidth. The advantages and disadvantages of cascade and parallel implementations are discussed in detail in Holmes (1983). Different analogue and digital implementations of formant synthesisers are described in Flanagan (1972) and Linggard (1985). Holmes (1973) implemented a high quality parallel formant synthesiser which used a stylised glottal pulse train as input. Klatt (1980) produced good quality synthesis from a formant synthesiser which aimed to combine the benefits of both architectures. A serial synthesiser was used for non-nasalised vowels and a parallel combination, which included a separate nasal pole-zero branch, was used for nasals, stops and fricatives. The parallel configuration was also used to synthesise vowels when independent control of the formant amplitudes was required, but did not give as good a quality of synthesis as the cascade configuration.

2.2.2 Time Domain Acoustic Modelling

Linear time domain acoustic modelling forms the basis for most commercial low bit rate coders. Despite the inherently limited performance of linear models on nonlinear data such as speech, they are popular due to their easy analysis and implementation. The analysis technique most commonly used is linear prediction analysis, which is reviewed in the following section.

2.3 Linear Prediction Analysis

2.3.1 Linear Prediction for Speech Analysis and Synthesis

Linear prediction techniques use a source-filter arrangement to model the vocal tract system, Fig. 2.1, which assumes that the source is located at the glottis and that a linear filter is adequate to model the frequency properties of the vocal tract. At the analysis stage, it is assumed that no information about the excitation of the vocal tract is known and that the speech waveform can only be modelled from its previous values. The linear vocal tract filter defines an autoregressive (AR) model of the speech, in which the current speech sample, $y(t)$, is predicted from a linear combination of a finite number of past samples

$$\hat{y}_p(t) = -\sum_{k=1}^{n_a} a_k y(t-k) \tag{2.1}$$

where $\hat{y}_p(t)$ is the predicted speech sample. The prediction error or residual, $e(t) = y(t) - \hat{y}_p(t)$, represents structure in the speech which is not captured by the model. For a good model, the residual has no predictable structure and appears as white noise. For voiced speech, the residual has significant peaks at the pitch period which coincide with the instants of excitation of the vocal tract, which coincide with rapid closure of the vocal cords (Baer, Löfqvist & McGarr 1983).

In synthesis (decoding), the actual speech samples, $y(t)$, in Eqn. (2.1) are replaced by $\hat{y}_s(t)$, the predicted values from the filter output. The vocal tract transfer function is expressed in the z-domain as :

$$\hat{H}(z) = \frac{G}{A(z)} \tag{2.2}$$

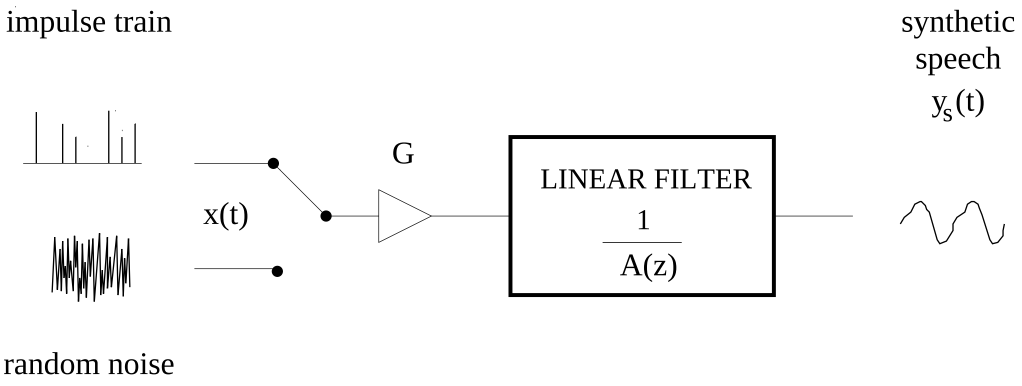


Figure 2.3: Source-filter arrangement for speech synthesis using a linear prediction model.

$$\hat{H}(z) = \frac{G}{1 + \sum_{k=1}^{n_a} a_k z^{-k}} \tag{2.3}$$

$$G = \sqrt{\frac{\sum_{t=1}^{N_a} N_s e^2(t)}{\sum_{t=1}^{N_s} N_a x^2(t)}} \tag{2.4}$$

where the gain term, G, is typically calculated by Eqn. (2.4), so that the energy per sample in the residual and excitation are equal. N_a and N_s are respectively the analysis and synthesis frame lengths, $e(t)$ is the residual, and $x(t)$ is the excitation in synthesis.

When the LP model is excited by the residual signal, the speech waveform is reproduced exactly. This is not practical for most applications, and one approach is to use a model of the residual signal. A source-filter arrangement is used, Fig. 2.3, in which the residual is represented by an impulse train at the pitch frequency for voiced sounds or a random, white noise generator for unvoiced sounds.

The estimated vocal tract transfer function, $\hat{H}(z)$, represents the combined effects of the glottal wave shape, vocal tract response and lip radiation. The spectrum of the output of the source-filter arrangement, $\hat{Y}(e^{j\omega})$, is given by

$$\hat{Y}(e^{j\omega}) = \hat{H}(e^{j\omega})X(e^{j\omega}) \tag{2.5}$$

where $X(e^{j\omega})$ is the spectrum of the excitation and $\hat{H}(e^{j\omega})$ is the frequency response of the estimated vocal tract model. The source spectrum is constant in amplitude because the source is modelled by white noise (unvoiced speech) or an impulse train at the pitch frequency (voiced speech) and is continuous for white noise or has discrete components at the pitch frequency for the impulse train. Since the source spectrum is flat, the parameters of $A(z)$ should be derived so that $\hat{H}(e^{j\omega})$ gives a good match to the spectrum of the speech. A match between the spectral envelope (magnitude spectrum) of the speech and the frequency response of $\hat{H}(z)$ is obtained if the parameters, a_k, are derived to minimise the mean-squared prediction error, E, over the analysis frame length, giving maximum likelihood estimates of parameters assuming prediction errors have Gaussian distributions. The resultant set of simultaneous equations has a closed-form solution for which there are well established algorithms. The methods of Durbin and Burg (see (Rabiner & Schafer 1978)) are both autocorrelation methods, in which it is assumed that samples outside the current analysis frame are zero. The covariance method uses samples outside the current analysis frame and results in a system of equations with different properties. A detailed comparison of the autocorrelation and covariance methods of linear prediction is given in Chandra & Lin (1974), in which the effects of variation of predictive order and analysis window size are also discussed.

2.3.2 Spectral Matching

The spectral match between the estimated transfer function, $\hat{H}(z)$ and the spectral envelope of the speech is shown by applying Parseval's Theorem to Eqn. (2.6). Eqn. (2.9) shows that minimising E is equivalent to minimising the integral of the ratio of the energy spectrum of the speech segment to the magnitude squared of the frequency response of the system model.[1]

$$\begin{aligned} E &= \sum_{t=0}^{N+n_a-1} e^2(t) && (2.6) \\ &= \frac{1}{2\pi}\int_{-\pi}^{\pi} \|Y(e^{j\omega})\|^2 \|A(e^{j\omega})\|^2 dw && (2.7) \\ \hat{H}(e^{j\omega}) &= \frac{G}{A(e^{j\omega})} && (2.8) \\ E(e^{j\omega}) &= \frac{G^2}{2\pi}\int_{-\pi}^{\pi} \frac{\|Y(e^{j\omega})\|^2}{\|\hat{H}(e^{j\omega})\|^2} dw && (2.9) \end{aligned}$$

For a predictor of order n_a, the first $n_a + 1$ values of the autocorrelation of the speech segment and autocorrelation function of the system impulse response are equal. Thus as the predictor order tends to infinity, the magnitude spectra $\|\hat{H}(e^{j\omega})\|$ and $\|Y(e^{j\omega})\|$ will match

$$\lim_{n_a \to \infty} \|\hat{H}(e^{j\omega})\|^2 = \|Y(e^{j\omega})\|^2 \qquad (2.10)$$

[1] Strictly this applies to the autocorrelation method, hence the limits in the summation of Eqn. (2.6). Only in this case do the autocorrelation function of the speech segment and the autocorrelation function of the system impulse response become equal. However, the covariance method still yields a transfer function which may be interpreted as an estimate of the speech spectrum (Rabiner & Schafer 1978).

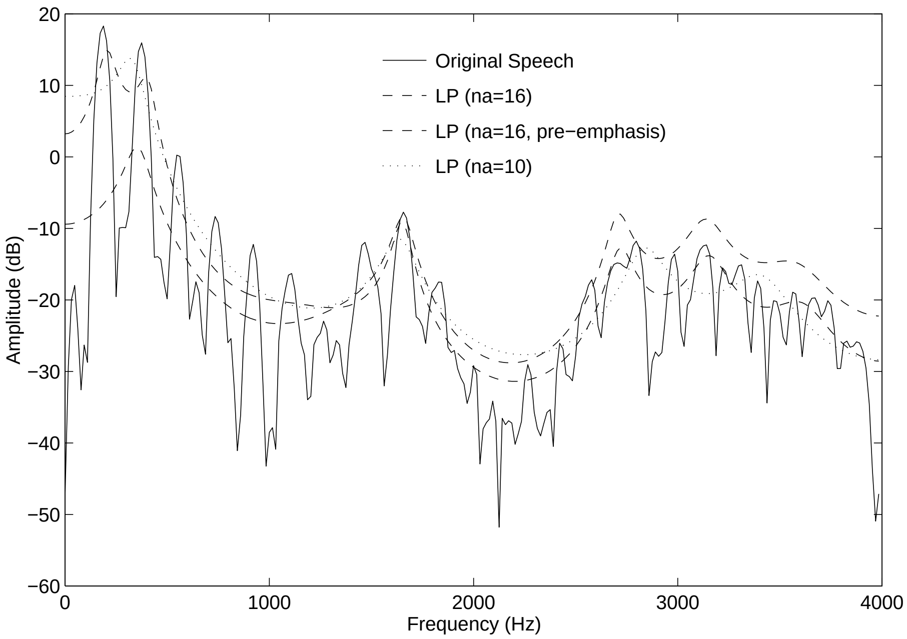


Figure 2.4: Examples of linear prediction spectra for the phoneme 'n' in 'in'.

However, the spectra $\hat{H}(e^{j\omega})$ and $Y(e^{j\omega})$ may not be equivalent because $\hat{H}(e^{j\omega})$ is constrained to be minimum-phase (all zeros inside the unit circle). In general, the speech spectrum is not minimum-phase when radiation occurs from more than one point and there are multiple sound pathways. Due to the spectral matching property of the mean-squared error criterion, linear prediction analysis can be used to obtain a smoothed estimate of the short-time spectral envelope of the speech, as illustrated in Fig. 2.4. Since E depends on the ratio of $||Y(e^{j\omega})||$ and $||\hat{H}(e^{j\omega})||$, the matching process performs uniformly over the frequency range of interest, regardless of the shape of the spectral envelope of the speech. However, the formants of the spectrum are more closely modelled because regions where $||Y(e^{j\omega})|| > ||\hat{H}(e^{j\omega})||$ contribute more to E than regions where $||Y(e^{j\omega})|| < ||\hat{H}(e^{j\omega})||$. Estimates of the speech formants can be obtained by locating peaks in the smoothed spectral envelope or by factorizing $A(z)$ into its constituent poles. Each formant is approximated by a complex-conjugate pole pair which forms a second order filter with transfer function, $A_i(z)$, given by

$$A_i(z) = 1 + a_1 z^{-1} + a_2 z^{-2} \tag{2.11}$$

The frequency of the formant is determined from the pole angle and the bandwidth from the radius. The spectrum is unique in the range $-w_s/2 < w < w_s/2$ and repeats at multiples of the sampling frequency, w_s. The transfer function, $\hat{H}(z)$, is stable when all the poles lie inside the unit circle. This holds if all $A_i(z)$ are stable. Analogous to the cascade and parallel realisation of the resonant network used in formant synthesisers, $\hat{H}(z)$ can be implemented in cascade or parallel form. In cascade form, $\hat{H}(z)$ is expanded as a product of formant factors

$$\hat{H}(z) = \frac{G}{\prod_{i=1}^{\frac{n_a}{2}} A_i(z)} \tag{2.12}$$

In parallel form $\hat{H}(z)$ is expanded as a sum of formant factors

$$\hat{H}(z) = \sum_{i=1}^{\frac{n_a}{2}} B_i(z)/A_i(z) \tag{2.13}$$

where $B_i(z)$ are the residues of the partial fraction expansion of $\hat{H}(z)$.

2.3.3 Predictor Order

A minimum predictor order of $n_a = 8$ is required to model the formants of speech sampled at 8kHz, since 2 poles are required for each formant and there are typically 4 formants in the range 0-4kHz. However, additional poles are needed to model the effects of glottal shaping and lip radiation, as these factors are also represented by $A(z)$. The effect of lip radiation can be approximated by $R(z)$, Eqn. (2.14), with $\mu \approx 1$. Thus the effect of lip radiation is to contribute $+6dB/$octave to the amplitude spectrum of speech.

$$R(z) = 1 - \mu z^{-1} \tag{2.14}$$

The effect of the glottal volume velocity pulse can be approximated by 2 real poles close to $z = 1$ (Deller, Proakis & Hansen 1993). Although the spectrum of a 2-pole model has the correct $-12dB/$octave slope to the magnitude spectrum (see section 3.5.2), the minimum-phase model cannot represent a glottal volume velocity pulse for which the opening time is longer than the closing time. As a result, the shape of the pulse in the time domain is a poor match to the shape of pulses of a real glottal volume velocity waveform.

Nasal coupling may introduce another formant and an anti-resonance into the spectrum of the vocal tract. The formant is modelled by an additional complex-conjugate pair of poles and another 4 poles (2 complex-conjugate pole pairs) are needed to approximate the anti-resonance. Typical analysis orders which give good synthesis lie in the range $n_a = 10 - 16$.

2.3.4 Pre-emphasis of Speech

Assuming that the glottal volume velocity pulses can be approximated by a two-pole model and the lip radiation by Eqn. (2.14), the combined effect of these two factors is to introduce a $-6dB$ per octave shift to the magnitude spectrum of synthetic speech. One method to obtain a linear prediction filter from which this trend is removed is to apply a pre-emphasis filter, $P(z)$, to voiced speech prior to analysis. The filter parameters are calculated to minimise the prediction error for the pre-emphasised speech and a post-filter, $P^{-1}(z)$, is applied to the resulting LP synthesis to generate the required synthetic speech. Typically, the pre-emphasis filter used is

$$P(z) = 1 - \mu z^{-1} \tag{2.15}$$

where μ is in the range $0.9 - 0.95$. Markel & Gray (1974) proposed an optimal value of μ which takes a value close to 1 for voiced speech and close to zero for unvoiced speech. The pre-emphasis filter is identical in form to the lip radiation factor, Eqn. (2.14), and its effect can be interpreted as 'whitening' the speech spectrum because the action of the filter is to approximately cancel out the $-6dB/$octave component of voiced speech due to the combined effect of lip radiation and source characteristics. The linear prediction order can thus be reduced by 1, with little loss of quality. Pre-emphasis does not, however, subtract out the effect of glottal shaping, and 2 real poles (one positive and one negative) are still needed to account for this shaping. As illustrated in Fig. 2.4, the frequency response of the resulting linear prediction model approximates that of the vocal tract only (plus glottal shaping).

A further benefit of pre-emphasis is to give improved estimation of higher formants. Due to the natural roll-off of the speech spectrum (or that introduced by a poor anti-aliasing filter), the speech spectrum tends to be dominated by lower frequencies and the small singular values of the autocorrelation (or covariance) matrix correspond to higher formants (Milenkovic 1986). When a large model order is used, the autocorrelation (co-variance) matrix becomes ill-conditioned and the higher formants are poorly estimated. Pre-emphasis accentuates the higher frequency region of the speech spectrum thus reducing the dynamic range and improving the numerical stability of the parameter estimation procedure.

2.3.5 Limitations of Linear Prediction for Analysis and Synthesis

The advantages of linear prediction for speech analysis are ease of implementation, a closed-form solution, complete separation of the source and vocal tract filter in synthesis, and a direct interpretation in terms of a loss-less acoustic tube model of the vocal tract (Rabiner & Schafer 1978).

Linear prediction has several disadvantages. Unvoiced sounds are poorly modelled by a minimum phase, all-pole linear prediction model because the vocal tract transfer function for these sounds contains zeros. Although argued that the spectral notches produced by zeros are hard to detect (Klatt 1987), synthesis of unvoiced sounds by a linear prediction model is poor. In linear prediction models, zeros have to be approximated by a collection of poles which requires a higher prediction order. The linear prediction parameters are not optimal for synthesis, since they are developed to minimise the mean-squared prediction error, rather than the actual error obtained at the output of the model when used for synthesis (the synthesis error). The main disadvantage of linear prediction is that the source and vocal tract filter are not decoupled in analysis and the linear prediction filter thus models the combined effect of source, vocal tract and lip radiation. As a result, the quality of synthetic speech generated from linear prediction models degrades rapidly as the pitch of the excitation is altered from that of the original speech.

2.4 Improvements to Linear Prediction Analysis and Synthesis

2.4.1 Analysis-by-Synthesis Techniques

The unnatural quality of linear prediction synthesis is due to the simplified model of the residual signal which is used to excite the linear prediction filter in synthesis. The limitations of this simplified source can be compensated for using analysis-by-synthesis techniques. Analysis-by-synthesis techniques aim to improve the quality of the synthesis from a pre-determined linear prediction filter by reducing the mean-squared *synthesis error* (or *output error*) between the original speech and the output of the filter when used in synthesis, Fig. 2.3. This can be done by improving the model of the residual signal which is used to excite the linear prediction filter and is the approach used in multi-pulse and code excited linear prediction (CELP) speech coders.

In multi-pulse coders (Atal & Remde 1982), the representation of the voiced residual during a pitch period is improved by using a series of impulses with different amplitudes and positions, rather than a single impulse. The positions and amplitudes of the pulses are optimised to minimise the mean-squared synthesis error at the coder output.

In CELP coders (Schroeder & Atal 1985), the excitation is generated by a codebook of random sequences which are fed through a long-term prediction filter. The long-term prediction filter accounts for the long-term correlations, or pitch structure, in the residual signal. In synthesis, a gain term and the parameters of the long-term prediction filter are calculated for each entry in the codebook and those values which minimise the mean-squared synthesis error are retained and transmitted to the decoder.

2.4.2 Perceptual Weighting Filters

Analysis-by-synthesis techniques minimise the mean-squared error between the original speech waveform and the waveform of the synthetic speech generated at the output of the vocal tract model or speech coder. This gives a much better fit between the waveforms in the time domain, but does not necessarily lead to a perceptually better synthesis. The perceived loudness of the signal distortion (synthesis error or coding noise) does not depend on its magnitude alone, but is also determined by the spectral shape of the noise with respect to that of the synthetic speech (Atal & Schroeder 1979).

In speech synthesis, the original speech signal is modelled by the sum of the synthetic speech and the noise signal. The theory of auditory masking suggests that noise in the region of the formants of the synthetic signal can be partially obscured by this signal (Atal & Schroeder 1979). The perceived loudness of the noise signal can therefore be reduced by shaping its spectrum to resemble that of the synthetic speech. In analysis-by-synthesis techniques, minimisation of the mean-squared synthesis error has the effect of flattening the magnitude spectrum. The appropriate spectral envelope can be achieved by filtering the synthesis error by a *perceptual weighting filter*, prior to the minimisation stage. The perceptual weighting filter concentrates the synthesis error into regions of the synthetic speech spectrum where it is less audible, so improving the subjective quality of the synthetic speech. Detailed studies of the relative loudness of noise in the presence of a speech signal, which are based on a mathematical formulation of the physiology of hearing (Schroeder, Atal & Hall 1979*a*) have shown that the noise can be made completely inaudible using this technique, if the perceptual weighting filter is complex enough.

For the linear prediction model, the spectrum of the synthetic speech is given, to within a scale factor, by the inverse filter, $1/A(z)$. Using no weighting filter results in a signal-to-noise ratio which is approximately constant at all frequencies and thus assumes that the ear is equally sensitive to distortion at all frequencies. However, the ear is more sensitive at lower frequencies. Studies by Atal & Schroeder (1979) on the effects of quantisation noise have shown that a synthesis error with spectrum $A(z/\alpha)/A(z)$ gives better subjective performance than a synthesis error with a flat spectrum, even when the signal-to-noise ratio (SNR) for the flat spectrum error was higher (21dB compared to 23dB over all frequencies). This spectral shaping gave improved SNR between formants when compared to that of a synthesis error with a flat spectrum and gave much better SNR at low frequencies when compared to that of a synthesis error shaped to match $1/A(z)$. A synthesis error with spectrum shaped to $A(z/\alpha)/A(z)$ can be obtained using a weighting filter, $P(z)$

$$P(z) = \frac{A(z)}{A(z/\alpha)} \tag{2.16}$$

where $A(z)$ is the linear prediction filter and α is a bandwidth expansion factor. Most CELP coding schemes use a perceptual weighting filter of this form (Schroeder & Atal 1985, Kroon & Atal 1988).

For a range of values of α, typical frequency responses for $P(z)$ are shown in Fig. 2.5. Compared with the frequency response of $A(z)$, the frequency response of $P(z)$ has broader valleys than $A(z)$ and is relatively flat inbetween.

Harmonic weighting filters (Gerson & Jasiuk 1992) rely on the same principle as perceptual weighting filters and aim to concentrate noise into the pitch harmonics of voiced speech. Similar filtering techniques have been used to reduce the effects of noise introduced by quantisation of the linear prediction parameters (Atal & Schroeder 1979, Drogo De Iacovo & Montagna 1991).

2.4.3 Decoupling the Source and Vocal Tract Filter

One of the main disadvantages of linear prediction is that no information about the excitation of the vocal tract is used at the analysis stage so the frequency response of the LP model exhibits features which are attributable to the spectrum of this excitation. This is a problem for voiced sounds because the excitation is periodic and the speech spectrum

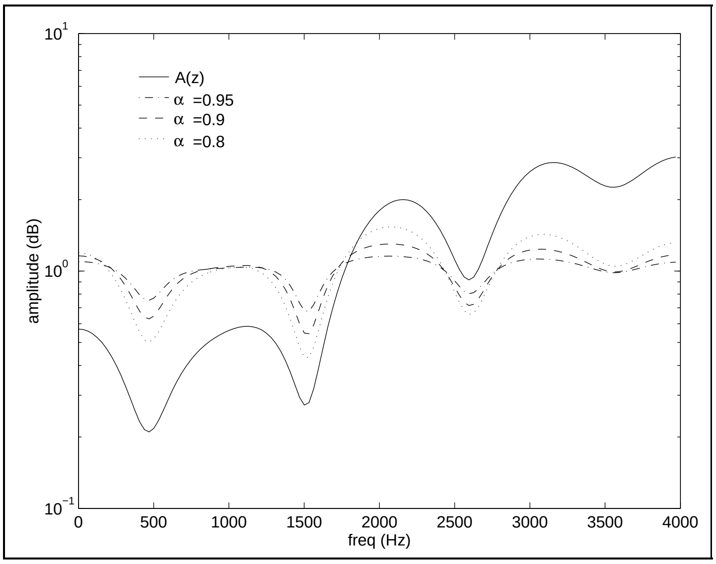


Figure 2.5: Amplitude spectra for typical weighting filters.

contains significant energy at harmonics of the pitch frequency. In such cases, the linear prediction equations in the autocorrelation form do not give accurate estimates of the formant frequencies and bandwidths, and errors in the first formant of up to ±8% have been reported (Klatt 1987). These errors are not a problem when the speech is synthesised from an excitation waveform at the original pitch frequency, but if the pitch of the source waveform is altered, significant distortion of the synthetic speech occurs. The source and filter can only be truly decoupled if the spectral effects of the true excitation of the vocal tract system are accounted for at the analysis stage. In theory, this should give vocal tract models for which the pitch of the excitation can be altered, without significant distortion of the synthetic speech generated by the model. Several approaches to developing a linear prediction filter from which the effect of the excitation has been removed are

- use a pre-emphasis filter to filter data prior to analysis
- pitch-synchronous analysis
- robust pitch estimation
- use the true excitation of the vocal tract at the analysis stage

The use of a pre-emphasis filter was discussed in section 2.3. In pitch-synchronous analysis, the analysis interval is reduced to a length of one pitch period or less so that the harmonic structure of the excitation does not appear in the short time spectrum. Pitch-synchronous and closed-phase linear prediction rely on this principle. Both approaches require the covariance method of linear prediction analysis because the analysis intervals are short, especially for high pitch speakers. This is a disadvantage because the stability of the filter is not ensured and reflection of unstable poles inside the unit circle is required. In addition, accurate location of the pitch period is needed. Closed-phase linear prediction uses an analysis interval which spans the duration of the pitch period for which the vocal cords are closed and there is no excitation of the vocal tract and relies on the assumption that the glottis is indeed closed for some duration of the pitch period. Despite these limitations, closed-phase covariance linear prediction has been shown to give reliable estimates of formant frequencies and bandwidths (Krishnamurthy & Childers 1986, Pearce & Whitaker 1986).

Pitch-synchronous linear prediction derives parameters from an analysis interval which spans a whole pitch period. Although more samples are available for the analysis of high pitched speech and the analysis is less sensitive to incomplete glottal closure, errors of up to 10% in location of the first formant still occur because the analysis interval contains the excitation instant (Lee 1988).

Robust estimation aims to reduce the errors in estimation of lower formants and bandwidths which arise due to inclusion of the instant of glottal closure within the analysis interval. At the instant of glottal closure, large residual errors occur which can greatly increase the variance of the estimated LP parameters. The effect of these large errors is reduced by weighting large residual errors to contribute only linearly to the error criterion. The system of linear prediction equations is no longer linear and must be solved by an iterative procedure. Lee (1988) has shown that better decoupling of source and vocal tract filter is achieved using robust analysis because the influence of the periodicity of the excitation on estimation accuracy of the first formant frequency and bandwidth is reduced. Robust estimation is therefore less sensitive to the selection of the analysis window length and to the alignment of the excitation within that window.

Complete decoupling of the source characteristics from the frequency response of the estimated vocal tract model is achieved by using the true excitation of the vocal tract in the analysis stage. Since real measurements of this excitation are not usually available, an approximation or model of the excitation can be used. Provided the approximation to the true excitation of the vocal tract system is fairly good, the transfer function of the vocal tract model expresses the frequency properties of the vocal tract alone (and possibly the lip radiation factor) and decoupling of the source and vocal tract is achieved.

2.5 System Identification Approach to Vocal Tract Modelling

When an explicit excitation waveform is used in the analysis stage, the vocal tract model no longer describes an AR model of the speech waveform, but gives a functional mapping between the excitation signal, $x(t)$, and the output speech waveform, $y(t)$. The vocal tract modelling task is thus one of system identification, in which the aim is to model the underlying dynamics of the vocal tract. It is common to assume that the next output, $y(t)$, can be computed from the observed data up to and including time t-1, and that the observation is subject to a stochastic disturbance, $\nu(t)$. The input and output of the vocal tract system are related by the equation

$$y(t) = h(x(1), \ldots, x(t-1), y(1), \ldots, y(t-1), \boldsymbol{\theta}) + \nu(t) \tag{2.17}$$

The functional mapping, $h(.)$, is realised by a particular choice of model structure which is parametrized by a finite dimensional vector, $\boldsymbol{\theta}$. The additive disturbance, $\nu(t)$ accounts for noisy measurements and signals which affect the system in some way but are ignored by the model.

The function, $h(.)$ may represent a linear or nonlinear mapping. The following two chapters consider linear black box models and nonlinear neural network models as candidate model structures, and compare their performance for speech synthesis to that of models calculated by the conventional linear prediction analysis approach described in this chapter.

Chapter 3

Linear Models of the Vocal Tract

In this chapter, a linear vocal tract modelling framework is developed for speech synthesis. The framework aims to overcome one of the main problems of existing linear prediction based techniques: unnatural sounding synthetic speech which degrades rapidly in quality as the pitch of the excitation is altered from that of the original speech. For voiced speech, a system identification approach is adopted, in which an explicit representation of the vocal tract excitation is used at the analysis stage and the vocal tract model approximates the transfer function of the vocal tract system. For unvoiced speech, excitation is provided by a codebook of Gaussian sequences. In linear system identification, candidate model structures are typically chosen from a selection of black-box linear models, for which the behaviour, stability and parameter estimation algorithms have been extensively studied (Ljung 1987). Two specific forms of black-box linear model are considered in this chapter, the ARX (autoregressive with external input) model and the OE (output error) model, for which the parameters are estimated by equation error and output error minimisation respectively. Synthetic speech data is used to illustrate the differences between the two parameter estimation methods. The performance of the models for the synthesis of real speech data is compared with that of linear prediction synthesis and the perceptual significance of the different model types is explored. The superior pitch manipulation capabilities of black-box models is demonstrated and an audio tape demonstration is included with this thesis to enable the reader to verify the reported improvements.

3.1 Linear Black-Box Models

The system identification approach to vocal tract modelling is shown in Fig.3.1. In this approach, it is assumed that the vocal tract system can be represented by $H(q)$, an ideal transfer function[1] which describes the relationship between the glottal excitation signal,

[1] The forward shift operator, q, which is commonly adopted in system identification literature, is used in this chapter to express time-domain relationships. q is such that $qx(t) = x(t+1)$ and the corresponding delay operator is $q^{-1}x(t) = x(t-1)$. $H(q)$ will be referred to as the transfer function of a system, although strictly speaking, this term should be reserved for the z-transform notation, $H(z)$. The corresponding frequency response is given by $H(e^{j\omega})$.

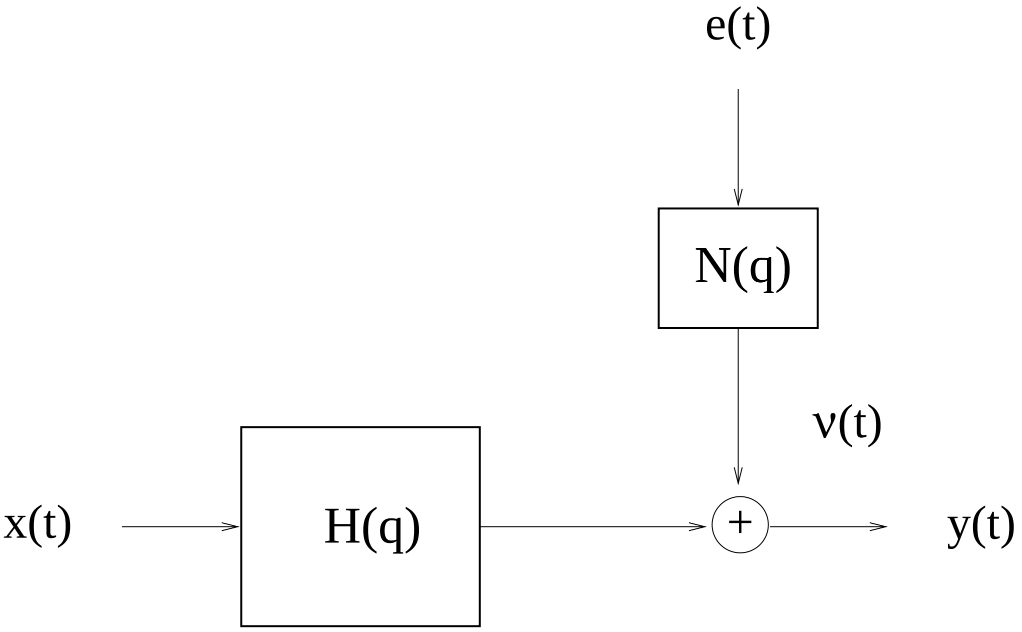


Figure 3.1: System identification approach to vocal tract modelling.

$x(t)$, and the speech signal, $y(t)$. It is further assumed that the output of the system is subject to a stochastic disturbance, $\nu(t)$. It is further assumed that $\Phi_v(\omega)$, the spectral density of the disturbance, can be represented by a noise model, $N(q)$, such that

$$\Phi_v(\omega) = \lambda \| N(e^{j\omega}) \|^2 \tag{3.1}$$

The input and output of the vocal tract system are thus related by the equation

$$y(t) = H(q)x(t) + N(q)e(t) \tag{3.2}$$

where $e(t)$ is a sequence of independent, identically distributed random variables with zero means and variances λ.

Linear black-box models assume that the functional mapping between the output and input of a dynamical system can be expressed as a set of vector difference equations such that $H(q)$ and $N(q)$ are parametrized as a ratio of polynomials in the backward shift operator q^{-1}. $H(q)$ and $N(q)$ are differentiable functions of the parameter vector, $\boldsymbol{\theta}$, which contains the coefficients of these polynomials. Several different parametrizations of black-box models have been developed which represent different families of models. The ARX (autoregressive with external input) and the OE (output error) families of models are considered in more detail in the following sections. Details of other parametrizations can be found, for example, in the book by Ljung (1987).

3.1.1 ARX Models

ARX models define the functional relationship between input and output as

$$\begin{aligned} y(t) &= -\sum_{j=1}^{n_a} a_j y(t-j) + \sum_{k=0}^{n_b-1} b_k x(t-k) + e(t) && (3.3) \\ A(q)y(t) &= B(q)x(t) + e(t) && (3.4) \end{aligned}$$

where $A(q)$ and $B(q)$ are polynomials in q^{-1}, with coefficients a_j and b_k respectively

$$A(q) = 1 + a_1 q^{-1} + a_2 q^{-2} + \cdots + a_{n_a} q^{-n_a} \qquad (3.5)$$

$$B(q) = b_0 + b_1 q^{-1} + \cdots + b_{n_b-1} q^{-n_b+1} \qquad (3.6)$$

The transfer function and model noise are given by

$$\begin{aligned} H(q) &= B(q)/A(q) && (3.7) \\ N(q) &= 1/A(q) && (3.8) \end{aligned}$$

ARX models make assumptions about the spectral density of the system noise. For a sufficiently high order model, ARX models are able to capture the dynamics of any system. As the order of the model is reduced, the transfer function may give a poor estimate of the true transfer function of the system if too many poles of $A(q)$ are needed to model the system noise. Eqn. (3.3) can be written as a linear regression

$$y(t) = \boldsymbol{\theta}_{arx}^{\mathrm{T}} \phi_{arx}(t) + e(t) \qquad (3.9)$$

$$\boldsymbol{\theta}_{arx} = [a_1 \ \ldots \ a_{n_a} \ b_0 \ \ldots \ b_{n_b-1}]^{\mathrm{T}} \qquad (3.10)$$

$$\phi_{arx}(t) = [-y(t-1) \ \ldots \ -y(t-n_a) \ x(t) \ \ldots \ x(t-n_b+1)]^{\mathrm{T}} \qquad (3.11)$$

ARX models are known as *equation error* models because $e(t)$ represents that part of the data which is not accounted for by the model parameters. Linear prediction models, which were considered earlier, and are widely used in speech processing, are also equation error models. They define an AR (autoregressive) model of the vocal tract system, which is given by Eqn. (3.3) with $x(t)$ and $B(q)$ set to zero.

3.1.2 OE Models

OE models describe the undisturbed output (or synthesis), $\hat{y}_s(t)$, as a linear difference equation

$$\begin{aligned} \hat{y}_s(t) &= -\sum_{j=1}^{n_f} f_j \hat{y}_s(t-j) + \sum_{k=0}^{n_b-1} b_k x(t-k) && (3.12) \\ F(q)\hat{y}_s(t) &= B(q)x(t) && (3.13) \end{aligned}$$

OE models assume that the parameters define the system output exactly and that any error between the observed output and the true system output is due to additive disturbance noise, $e(t)$. The observed output is given by

$$\begin{aligned} y(t) &= \hat{y}_s(t) + e(t) && (3.14) \\ y(t) &= \frac{B(q)}{F(q)} x(t) + e(t) && (3.15) \end{aligned}$$

where $F(q)$ and $B(q)$ are polynomials in q^{-1} with coefficients f_j and b_k respectively. The transfer function and noise model are given by

$$\begin{aligned} H(q) = B(q)/F(q) && (3.16) \\ N(q) = 1 && (3.17) \end{aligned}$$

The output can be expressed as a pseudo-linear regression

$$\begin{aligned} y(t) &= \boldsymbol{\theta}_{oe}^{\mathrm{T}} \phi_{oe}(t) + e(t) && (3.18) \\ \boldsymbol{\theta}_{oe} &= [f_1 \ \dots \ f_{n_f} \ b_0 \ \dots \ b_{n_b-1}]^{\mathrm{T}} && (3.19) \\ \phi_{oe}(t) &= [-\hat{y}_s(t-1) \ \dots \ -\hat{y}_s(t-n_a) \ x(t) \ \dots \ x(t-n_b+1)]^{\mathrm{T}} && (3.20) \end{aligned}$$

3.2 Prediction versus Synthesis

The distinction between operation of models in prediction and synthesis[2] is illustrated in Fig. 3.2. The one-step ahead prediction, $\hat{y}_p(t)$, and the synthesis of the system output, $\hat{y}_s(t)$, are given by the equations

$$\hat{y}_p(t) = N^{-1}(q)H(q)x(t) + [1 - N^{-1}(q)]y(t) \tag{3.21}$$

$$\hat{y}_s(t) = H(q)x(t) \tag{3.22}$$

Thus there are two errors on which the performance of the different model families can be based. The one-step ahead prediction error, $e_p(t)$, is given by

$$\begin{aligned} e_p(t) &= y(t) - \hat{y}_p(t) \qquad (3.23) \\ &= N^{-1}(q)[y(t) - H(q)x(t)] \qquad (3.24) \end{aligned}$$

The synthesis error, $e_s(t)$, is given by

$$\begin{aligned} e_s(t) &= y(t) - \hat{y}_s(t) \qquad (3.25) \\ &= y(t) - H(q)x(t) \qquad (3.26) \end{aligned}$$

For ARX models, the prediction error, $e_p(t)$, is exactly the *equation error* of Eqn.3.3. For OE models, the synthesis error, $e_s(t)$, is exactly the *output error* of Eqn. 3.12. For a given model family, the prediction and synthesis errors are related by the expression

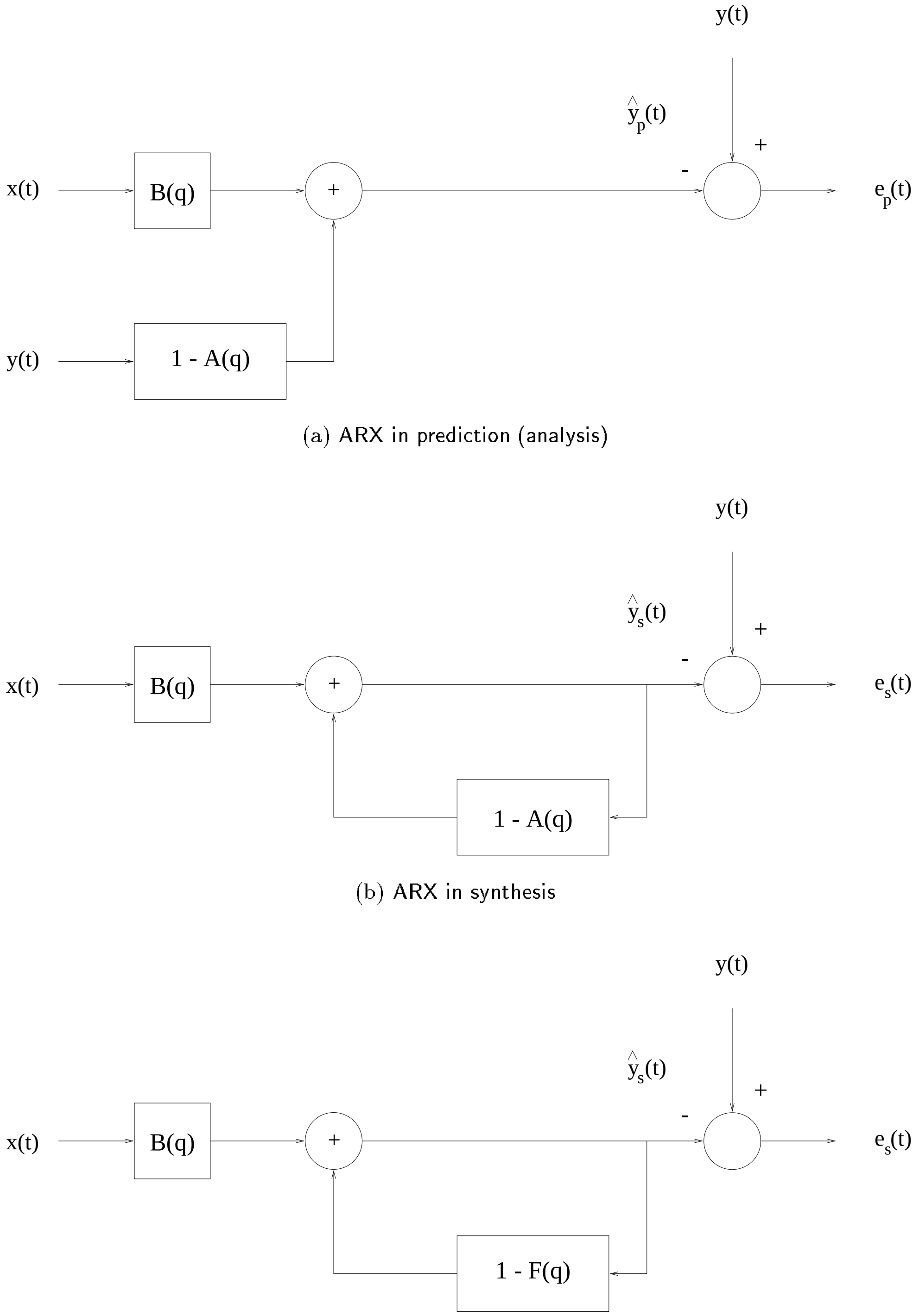
$$e_p(t) = N^{-1}(q)e_s(t) \tag{3.27}$$

OE models are termed 'natural' predictors because $e_p(t) = e_s(t)$. In this and the following chapter, model performance is evaluated objectively by the prediction and synthesis signal-to-noise ratios (SNR) which are defined as

$$\text{prediction SNR} = 10\log_{10}\left(\frac{\sum\limits_{t=0}^{N-1} y^2(t)}{\sum\limits_{t=0}^{N-1} e_p^2(t)}\right) \tag{3.28}$$

$$\text{synthesis SNR} = 10\log_{10}\left(\frac{\sum\limits_{t=0}^{N-1} y^2(t)}{\sum\limits_{t=0}^{N-1} e_s^2(t)}\right) \tag{3.29}$$

[2] In system identification nomenclature, the model generated output, $\hat{y}_s(t)$, is normally referred to as the noise-free simulation of a system. Here, the term synthesis is used because the models are of the vocal tract and the output generated in this manner represents synthesised speech.



(a) ARX in prediction (analysis)

(b) ARX in synthesis

(c) OE in synthesis (analysis)

Figure 3.2: Operation of black-box models in prediction and synthesis.

3.3 Parameter Estimation

One method for finding the model parameters that best describe an N-length data record is the prediction-error method. The sum of squared prediction errors, $V_N(\boldsymbol{\theta})$, is minimised with respect to the parameters, $\boldsymbol{\theta}$.

$$V_N(\boldsymbol{\theta}) = \frac{1}{2}\sum_{t=0}^{N-1} e_p^2(t) \tag{3.30}$$

$$\hat{\boldsymbol{\theta}} = \underset{\boldsymbol{\theta}}{\text{argmin}} \ V_N(\boldsymbol{\theta}) \tag{3.31}$$

For ARX models, the prediction error is an equation error, and the parameter estimation is known as *equation error minimisation.* The equations are linear in the parameter and can be found in closed-form by the least-squares method (Ljung 1987).

For OE models, the prediction-error method is actually an *output error minimisation* procedure, because $e_p(t) = e_s(t)$. The parameters are defined by a system of nonlinear equations ($e_s(t)$ depends on previous model outputs) and must be solved by iterative techniques such as the Gauss-Newton method (Ljung 1987). Adaptive filtering techniques, such as the SHARF algorithm, have also been proposed (Niranjan 1990).

3.3.1 Frequency Domain Interpretation of Prediction-Error Method

In modelling the vocal tract system, the aim is to derive models from which high quality synthetic speech can be generated. When evaluating the performance of a model, it is not sufficient to compare mean squared synthesis error alone, since this does not account for perceptually significant factors such as auditory masking and the varying sensitivity of the ear across frequencies. As discussed in section 2.4.2, the shape of the synthesis error spectrum influences the perceived loudness of this signal. Although high quality synthesis requires an accurate estimate of the true system transfer function, the ear is less sensitive to distortion at higher frequencies. Thus, the frequency bias of the transfer function estimate and the spectrum of the synthesis error need to be considered when comparing different families of models. This requires a spectral interpretation of the prediction-error method[3] which is used to calculate the model parameters. The prediction error criterion, Eqn. (3.30), can be transformed to the frequency domain using Parseval's Theorem. Following Ljung (1987, pp.173–175), Eqn. (3.30) can be written as

$$V_N(\boldsymbol{\theta}) \approx \frac{1}{4\pi}\int_{-\pi}^{\pi} \|N(e^{j\omega},\boldsymbol{\theta})\|^{-2}\|Y(e^{j\omega}) - H(e^{j\omega},\boldsymbol{\theta})X(e^{j\omega})\|^2 \, dw \tag{3.32}$$

[3] A more detailed discussion on the frequency domain interpretation of the prediction error method is given by Ljung (1985) and Wahlberg & Ljung (1986).

$$= \frac{1}{4\pi}\int_{-\pi}^{\pi} \frac{||\Phi_{\mathrm{ER}}(\omega,\boldsymbol{\theta})||^2}{||N(e^{j\omega},\boldsymbol{\theta})||^2}\, dw \tag{3.33}$$

$$V_N(\boldsymbol{\theta}) \approx \frac{1}{4\pi}\int_{-\pi}^{\pi} ||\frac{Y(e^{j\omega})}{X(e^{j\omega})} - H(e^{j\omega},\boldsymbol{\theta})||^2 \frac{||X(e^{j\omega})||^2}{||N(e^{j\omega},\boldsymbol{\theta})||^2}\, dw \tag{3.34}$$

$$= \frac{1}{4\pi}\int_{-\pi}^{\pi} ||\hat{H}(e^{j\omega}) - H(e^{j\omega},\boldsymbol{\theta})||^2 Q(\omega,\boldsymbol{\theta})\, dw \tag{3.35}$$

where the additional term due to the additive disturbance, $\nu(t)$, is ignored. Eqn. (3.33) can be interpreted as a fit between the spectrum of the model noise, $N(e^{j\omega},\boldsymbol{\theta})$, and the spectrum of the synthesis error (output error) of the system, $\Phi_{\mathrm{ER}}(\omega,\boldsymbol{\theta})$, which is given by

$$\Phi_{\mathrm{ER}}(\omega,\boldsymbol{\theta}) = Y(e^{j\omega}) - H(e^{j\omega},\boldsymbol{\theta})X(e^{j\omega}) \tag{3.36}$$

$\hat{H}(e^{j\omega})$ is the *empirical transfer function estimate* (ETFE) for the system, which is an estimate of the transfer function calculated from the ratio of the spectra of the output and input of the system at each frequency of interest[4]

$$\hat{H}(e^{j\omega}) = \frac{Y(e^{j\omega})}{X(e^{j\omega})} \tag{3.37}$$

Eqn. (3.35) can be interpreted as a weighted least-squares fit between the model transfer function, $H(e^{j\omega},\boldsymbol{\theta})$, and the ETFE, $\hat{H}(e^{j\omega})$. The weighting function, $Q(\omega,\boldsymbol{\theta})$, represents the model signal-to-noise ratio

$$Q(\omega,\boldsymbol{\theta}) = \frac{||X(e^{j\omega})||^2}{||N(e^{j\omega},\boldsymbol{\theta})||^2} \tag{3.38}$$

$Q(w,\boldsymbol{\theta})$ determines the frequency distribution of bias in the estimate of the transfer function, in the case when the true system is not in the model set defined by $H(e^{j\omega},\boldsymbol{\theta})$ and $N(e^{j\omega},\boldsymbol{\theta})$. Higher values of $Q(\omega,\boldsymbol{\theta})$ enforce a better fit between $H(e^{j\omega},\boldsymbol{\theta})$ and $\hat{H}(e^{j\omega})$.

For models with $N(e^{j\omega},\boldsymbol{\theta})$ independent of $\boldsymbol{\theta}$, OE models for example, the prediction-error method can be interpreted as finding the parameters, $\hat{\boldsymbol{\theta}}$, for which the model transfer function, $H(e^{j\omega},\hat{\boldsymbol{\theta}})$, gives the best mean-square approximation to the ETFE, with frequency weighting $Q(\omega,\hat{\boldsymbol{\theta}})$. When the model noise depends on $\boldsymbol{\theta}$, ARX models for example, Eqn. (3.32) must also be satisfied, and the parameter estimate, $\hat{\boldsymbol{\theta}}$, can be regarded as a compromise between fitting $H(e^{j\omega},\hat{\boldsymbol{\theta}})$ to $\hat{H}(e^{j\omega})$ and fitting $N(e^{j\omega},\hat{\boldsymbol{\theta}})$ to $\Phi_{\mathrm{ER}}(\omega,\hat{\boldsymbol{\theta}})$.

[4] For frequencies where $X(e^{j\omega}) = 0$, the ETFE is regarded as undefined. The ETFE can be calculated, for example, from the ratio of the discrete Fourier transforms of the output and input sequences, at each discrete frequency. The ETFE is discussed in detail in Ljung (1987, pp. 146–151), and its relation to other estimates of the transfer function is described in (Ljung 1985).

3.3.2 Perceptual Significance of the Model Noise and Transfer Function Bias

A suitable frequency bias for the transfer function estimates should reflect the varying sensitivity of the ear across the frequency range. Although the ear is more sensitive at low frequencies, reliable estimation of high frequency formants is still necessary for perceptually good quality synthesis. The frequency bias is dependent on the excitation and the model parameters, $A(q)$ and $F(q)$. For speech applications, the spectrum of the voiced excitation, $X(e^{j\omega})$, tends to have a trend of -12dB/octave, and the frequency responses $A(e^{j\omega}, \boldsymbol{\theta})$ and $F(e^{j\omega}, \boldsymbol{\theta})$ are generally high-pass. For OE models, the frequency bias of the transfer function estimate is given by

$$Q_{oe}(\omega, \boldsymbol{\theta}) = \|X(e^{j\omega})\|^2 \tag{3.39}$$

Thus much poorer estimates are obtained at higher frequencies. For ARX models

$$Q_{arx}(\omega, \boldsymbol{\theta}) = \|X(e^{j\omega})\|^2 \|A(e^{j\omega}, \boldsymbol{\theta})\|^2 \tag{3.40}$$

The high-pass characteristic of $A(e^{j\omega}, \boldsymbol{\theta})$ cancels out, to some extent, the spectral trend of $X(e^{j\omega})$ and the dependence of the frequency bias on $A(e^{j\omega}, \boldsymbol{\theta})$ gives de-emphasis in the region of the formants of speech.

Due to the auditory masking properties of the human ear, the synthesis error can be made less audible if it is concentrated into the region of the formants of the synthetic speech. For ARX models, the parameter estimation procedure requires the spectrum of the synthesis error to match the model noise spectrum. The noise spectrum is given by

$$N_{arx}(e^{j\omega}, \boldsymbol{\theta}) = 1/A(e^{j\omega}, \boldsymbol{\theta}) \tag{3.41}$$

The frequency response, $1/A(e^{j\omega}, \boldsymbol{\theta})$, gives the formants of the synthetic speech, thus the synthesis error is concentrated into these regions where its perceived loudness is reduced by the effect of auditory masking. Thus the ARX model implicitly applies a degree of perceptual weighting to the synthesis error. Although this synthesis error spectrum matches the speech formants exactly, it results in a constant LP signal-to-noise ratio across all frequencies which does not reflect the varying sensitivity of the ear to distortion at different frequencies. OE models enforce no constraints on the spectrum of the synthesis error, which will tend to that of the true noise in the system. It is therefore difficult to predict the significance of the model noise on the perceived loudness of the synthesis error from such models.

3.3.3 Changing the Noise Model and Transfer Function Bias

A pre-emphasis filter can be used to adjust the noise model and frequency bias of the vocal tract transfer function estimate so that they have are more perceptually relevant for speech synthesis applications. For linear models, pre-filtering the input and output sequences by a filter, $P(q)$, prior to identification, is equivalent to filtering the prediction error and leads to a modified error criterion

$$\tilde{V}_N(\boldsymbol{\theta}) = \frac{1}{2}\sum_{t=0}^{N-1} e_{\mathrm{F}}^2(t) \tag{3.42}$$

where $e_{\mathrm{F}} = P(q)e_p(t)$. Pre-filtering alters the model noise and frequency bias functions to

$$\tilde{N}(e^{j\omega}, \boldsymbol{\theta}) = N(e^{j\omega}, \boldsymbol{\theta})/P(e^{j\omega}) \tag{3.43}$$

$$\tilde{Q}(\omega, \boldsymbol{\theta}) = Q(\omega, \boldsymbol{\theta})\|P(e^{j\omega})\|^2 \tag{3.44}$$

For ARX models, both $Q(\omega, \boldsymbol{\theta})$ and $N(e^{j\omega}, \boldsymbol{\theta})$ are dependent on the model parameters, $\boldsymbol{\theta}$, which are unknown before analysis. The effect of pre-emphasis on the frequency distribution of bias and spectrum of the synthesis error is only seen once analysis has been performed, and some trial and error is needed to find a suitable pre-emphasis filter to achieve the required frequency distribution, or a synthesis error spectrum with a particular shape[5].

For OE models, $Q(\omega, \boldsymbol{\theta})$ and $N(e^{j\omega}, \boldsymbol{\theta})$ are independent of the model parameters and the change in distribution of bias and synthesis error spectrum, caused by pre-emphasis with a particular filter, can be predicted (assuming the filter parameters are independent of the model parameters). Thus, a pre-emphasis filter for obtaining a desired frequency bias or a particular change in the spectrum of the synthesis error (assuming convergence to the global minimum) can be determined prior to analysis. Note that the model noise is independent of $\boldsymbol{\theta}$ and pre-filtering changes the spectrum of the synthesis error indirectly, by changing $Q(\omega, \boldsymbol{\theta})$. Pre-filtering changes both the global minimum and the local minima of the error criterion.

A full discussion on the effect of altering the frequency bias by pre-filtering and the effect of other design variables, such as the sampling interval and prediction horizon, is given by Wahlberg & Ljung (1986).

3.4 Model Order Selection

The order of the polynomials in the black-box model (n_a, n_b, n_f) is determined by three factors; the properties of the speech data to be modelled, the specific form of $H(q)$, and the form of the vocal tract excitation.

[5] The shape of the synthesis error spectrum after analysis is known in terms of $A(e^{j\omega}, \boldsymbol{\theta})$, but this is not known prior to analysis.

3.4.1 A(q) and F(q)

The poles of the vocal tract filter give the positions and bandwidths of the formants. Each formant is represented by a complex-conjugate pole pair, $[p_i, \bar{p}_i]$, where $p_i = r_i \exp(j\psi_i)$. The frequency, F_i and bandwidth, B_i, of the formant are given by

$$F_i = \psi_i / 2\pi T \tag{3.45}$$

$$B_i = -(1/\pi T) \log |r_i| \tag{3.46}$$

For male talkers, there is usually 1 formant per kHz (Klatt 1980) For speech sampled at 8 kHz, for which the frequency range of interest is 1-4 kHz, 8 poles are needed to model formants.

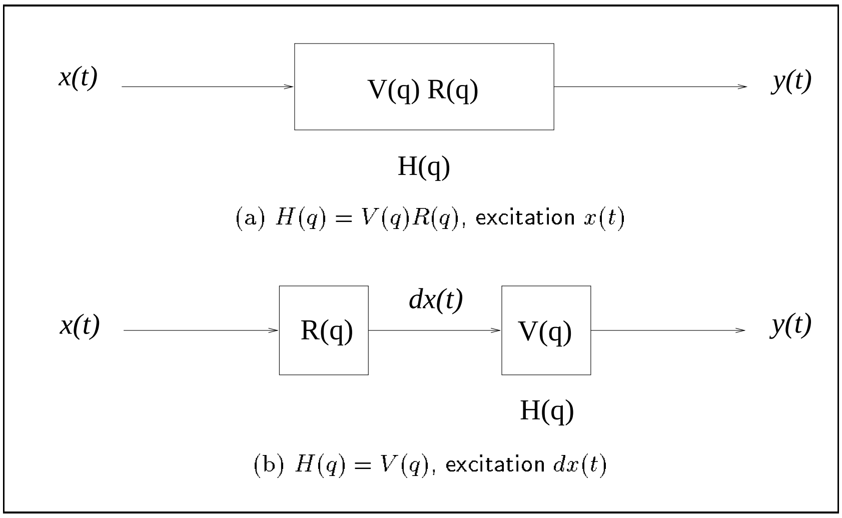
3.4.2 B(q)

The interpretation of $B(q)$ depends on how the lip radiation, $R(q)$, vocal tract transfer function, $V(q)$, and the effects of the shaping of the glottal excitation, $G(q)$, are combined within the vocal tract modelling framework. Several alternative configurations are possible and two are illustrated in Fig. 3.3. These configurations assume that $R(q) = 1 - \mu q^{-1}$ and that $G(q)$ is an all-zero model of the glottal excitation, such that $x(t) = G(q)e(t)$, where $e(t)$ is a zero mean, Gaussian excitation.

In Fig. 3.3(a), the black-box model is excited by a representation of the glottal excitation, $x(t)$, and the effect of lip radiation is lumped into the vocal tract transfer function, such that $H(q) = V(q)R(q)$. If it is assumed that n_z zeros are sufficient to model the zeros of the vocal tract, this configuration requires that $B(q)$ include an additional zero to account for the lip radiation effect, such that $n_b = n_z + 2$.

In Fig. 3.3(b), the effect of lip radiation is combined with the excitation before analysis, so that the black-box model is excitation by a signal, $dx(t)$, which is approximately the first difference of the glottal excitation. In this configuration, $H(q) = V(q)$ and $B(q)$ models the zeros of the vocal tract transfer function only, requiring $n_b = n_z + 1$.

Other alternatives are to separate out $R(q)$ from $H(q)$ in Fig. 3.3(a), which requires $n_b = n_z + 1$, or to lump the effect of glottal shaping into $H(q)$ in Fig. 3.3(b), such that $H(q) = G(q)V(q)R(q)$, and the excitation of the black-box model is assumed to be a zero mean, Gaussian excitation. Such a configuration is not beneficial, since it requires higher order n_b, which can lead to ill-conditioning of the parameter equations for ARX models. In addition, it is not possible to separate out those zeros which model the actual shape of the excitation. Additional zeros can also be included in $B(q)$ to allow the black-box model to take account of the propagation delay between glottis and lips and an advance misalignment between the instants of glottal closure in the excitation, $x(t)$, and the true instants of glottal closure for the original speech signal (see section 3.6.4).



(a) $H(q) = V(q)R(q)$, excitation $x(t)$

(b) $H(q) = V(q)$, excitation $dx(t)$

Figure 3.3: Source-filter configurations for black-box models with transfer function $H(q)$. $V(q)$ represents the transfer function of the vocal tract and $R(q)$ represents the effect of lip radiation.

3.5 Generating an Excitation Waveform for Black-Box Models

When modelling the vocal tract system with black-box models, Fig. 3.1, the actual excitation of the vocal tract is needed in the analysis stage. The excitation signal, $x(t)$, is assumed to be the glottal volume velocity waveform. Measurements of the actual glottal volume velocity waveform are not easy to obtain and, for practical purposes, an approximate representation is required. An accurate representation is needed if $H(q)$ is to model the true vocal tract transfer function realistically. The glottal volume velocity waveform can be approximated directly from the speech waveform using inverse-filtering techniques, or for voiced speech it can be modelled by a pitch-synchronous sequence of volume velocity pulse models which are synchronised to the instants of glottal closure of the vocal tract. These approaches are reviewed in the following sections.

3.5.1 Inverse Filtering Techniques

Inverse filtering techniques are based on filtering the speech through the inverse of the vocal tract and lip radiation filters, to obtain an estimate of the excitation signal (Ananthapadmanabha & Yegnanarayana 1979, Krishnamurthy & Childers 1986, Howard & Breen 1989, Hedelin 1984). Although glottal volume velocity waveforms derived from inverse filtering have been shown to give superior synthesis quality than pulse based models (Holmes 1973), inverse-filtering techniques suffer the disadvantage that they rely on the accuracy of linear prediction analysis. Pitch-synchronous or closed-phase covariance analysis is needed to eliminate the source characteristics from the transfer function. This is a disadvantage because it requires accurate location of the pitch period or closed-phase of a pitch period, which requires an external timing signal, such as that from a laryngograph, or an extensive search procedure of all possible locations. The short analysis intervals for high pitch speakers gives poor performance. In an attempt to overcome the limitation of a short analysis interval, Alku (1992) proposed an iterative inverse filtering technique, in which the source contribution is first estimated by a low-order LP analysis, and then removed from the speech by inverse filtering. The vocal tract filter is then iteratively re-estimated from the inverse-filtered speech, using an analysis interval that is no longer constrained to be a single pitch period in duration. However, the performance for female speakers was still poor. Hedelin (1984) has reported the poor performance of inverse-filtering techniques for sounds such as nasals, which require zeros in the transfer function. A further disadvantage of inverse-filtering techniques is that high quality speech recordings are needed because the analysis is sensitive to low frequency phase distortion.

3.5.2 Volume Velocity Pulse Models

Use of a pulse based representation of the vocal tract excitation has been shown to improve the naturalness of synthetic speech from formant synthesisers and vocal tract models (Holmes 1973, Rosenberg 1971, Fujisaki & Ljungqvist 1986). The parametric representation offered by volume velocity pulse models is advantageous for coding and storage purposes and allows easy adaptation of speaker style (for example, stressed, excited, breathy) by changing the underlying pulse shape (Rutledge, Cummings, Lambert & Clements 1995). The disadvantage of a pulse based representation is that it has to be accurately synchronised to the speech utterance, so that the instants of glottal closure of the pulses align with the true instants of glottal closure for a particular utterance. There are a variety of suitable pulse shapes, methods for determining their parameters and methods for correctly synchronising a pulse sequence to the speech waveform.

Choice of Pulse Shape

Examination of typical inverse-filtered waveforms shows that volume velocity pulses generally have an asymmetric pulse shape with faster closing phase than opening phase and an approximate -12 dB/octave spectral roll-off, which suggests a discontinuity in the first derivative. Assuming that the glottal excitation is zero for some duration of the pitch period (Flanagan 1972), it can be represented by an FIR filter with a finite number of zeros. Alternatively, a pulse shape can be defined by piece-wise polynomial or trigonometric functions. The Rosenberg pulse model (Rosenberg 1971), shown in Fig. 3.4, is comprised of two trigonometric sections. The opening and closing phases of the pulse, of duration T_o and T_c respectively, are given by:

$$x(t) \quad = \quad \frac{\alpha}{2}\left[1 - \cos(\frac{\pi t}{T_o})\right] \qquad 0 \leq t \leq T_o \tag{3.47}$$

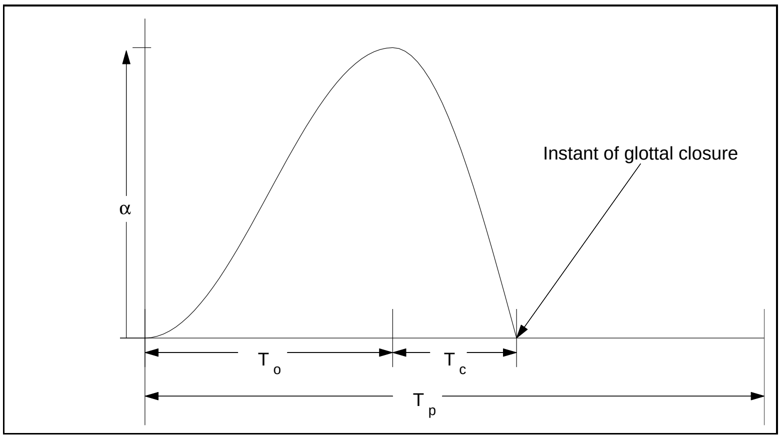


Figure 3.4: The Rosenberg glottal volume velocity wave pulse.

$$x(t) = \alpha\cos(\frac{\pi(t-T_o)}{2T_c}) \qquad T_o \leq t \leq T_o + T_c \qquad (3.48)$$

$$x(t) = 0 \qquad T_o + T_c < t < T_p \qquad (3.49)$$

The single slope discontinuity at closure gives the pulse a spectral envelope which falls off at −12 dB/octave. The asymmetric pulse shape avoids the occurrence of sharp dips in the spectrum which can cause cancellation or flattening of the formants of the vocal tract filter.

The Rosenberg pulse is defined by 3 parameters, the duty-cycle (ratio of open phase to pitch period, $(T_o + T_c)/T_p$), the pulse skew (ratio of the opening time to closing time, T_o/T_c), and the pulse amplitude, α_0. Fant (1979) also proposed a 3 parameter polynomial model allowing independent control of the derivative at the closure discontinuity. The LF model (Fant, Liljencrants & Lin 1985) has an additional parameter which gives a rounding of the volume pulse at closure. This accommodates breathy phonation, in which glottal closure is gradual and results in a small residual flow after the main excitation of the vocal tract. The LF model has a faster spectral roll-off than the −12dB/octave normally observed because the single discontinuity is no longer located at the instant of glottal closure, but at some point during the closing phase. More refined polynomial models using up to 6 parameters have also been proposed (Fujisaki & Ljungqvist 1986). These models allow for secondary excitation at glottal opening, a rounded glottal closure and a period of negative flow after glottal closure caused by lowering of the vocal folds, and were found to give improved prediction gains. Although the prediction gain is improved by using more complex pulse shapes, earlier studies by Rosenberg (1971) have shown that the simple pulse shape with a single slope discontinuity at closure, Fig. 3.4, gives good perceptual quality synthetic speech.

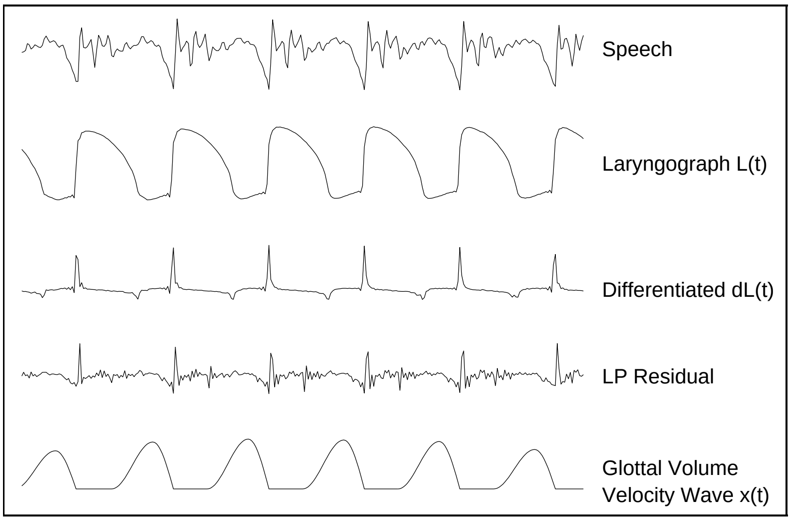


Figure 3.5: Typical speech, laryngograph, residual and glottal volume velocity waveforms (time-aligned).

Determining the Parameters of Pulse Models

A computationally intensive approach to determining the parameters of volume velocity pulses is to use an analysis-by-synthesis search of the entire parameter space. The parameters are systematically altered, a vocal tract model calculated for each new excitation, and the parameters which minimise the mean-squared error criterion over the analysis interval are selected. This method has been widely used in previous work (Fujisaki & Ljungqvist 1986, Fujisaki & Ljungqvist 1987, Wang et al. 1990, Thomson 1992). The computational requirement can be reduced by a coarse quantization of the parameter space. Funaki & Mitome (1990), for example, used a codebook of pulses with different parameters. Another computationally intensive approach, which was used by Hedelin (1984), is to enforce a pulse based parametrization of the inverse-filtered waveform by minimising the mean-squared error between the actual inverse-filtered waveform and a sequence of pulse models.

The studies by Rosenberg (1971) found that acceptable relative opening and closing times for glottal volume velocity pulses are typically 40% and 16% of the pitch period respectively. A simple approach for setting the timing parameters of pulse models is thus to fix the opening and closing times to 40% and 16% of the local pitch period.

Synchronising the Volume Velocity Pulses to the Speech Waveform

Cheng & O'Shaughnessy (1989) have shown that poor estimates of the parameters of the vocal tract model are obtained if the instants of glottal closure in the excitation signal and the true instants of glottal closure of an utterance are misaligned. In theory, the instants of glottal closure can be located from peaks in the linear prediction residual. Sounds such as vowels show clear peaks in the residual signal at the instant of glottal closure, as illustrated in Fig. 3.5. However, multiple peaks of either polarity occur around the instant of glottal closure when the real transfer function contains zeros, or when large errors in the estimates of formants and bandwidths occur (Ananthapadmanabha & Yegnanarayana 1979). This ambiguity results in poor estimates of the instants of glottal closure for high pitched sounds, voiced fricatives and nasals. Improved estimates of the instants of glottal closure from LP residuals can be obtained using robust techniques which are based on epoch filtering and maximum likelihood estimation (Ananthapadmanabha & Yegnanarayana 1979, Cheng & O'Shaughnessy 1989).

Accurate location of the instants of glottal closure requires monitoring of the vocal fold vibrations, for example using high speed filming techniques (Kiritani, Honda, Imagawa & Hirose 1986), laryngoscopy, photoglottography and electroglottograpgy (Borden & Harris 1984). Laryngoscopy and photoglottography require illumination of the vocal folds by a light source, which is then monitored by a mirror or photo-cell. Both are invasive techniques which inhibit normal phonation and are repugnant to many subjects. Electroglottography (using an electroglottograph or laryngograph) is the only practical method for monitoring vocal fold vibration since it is a non-invasive technique. A laryngograph (or electroglottograph) measures the change in conductance (impedance) between two small electrodes placed on either side of the larynx (Borden & Harris 1984, Fourcin & Abberton 1971). As the vocal folds come into contact, the impedance across the larynx decreases causing a peak in the laryngograph signal. In normal voicing, the laryngograph signal shows three distinct phases: a relatively sharp rise associated with rapid closure of the vocal cords, a gradual fall associated with opening of the vocal cords as the sub-glottal pressure increases, and a flatter base region corresponding to the interval for which the vocal cords are out of contact. A typical laryngograph signal for normal voicing, which has been time-aligned to the speech signal to account for the propagation delay between glottis and microphone, is shown in Fig. 3.5 (transconductance is taken as positive). Comparisons of high speed filming of the vocal cords with simultaneously recorded laryngograph signals carried out by Baer et al. (1983), have shown that the laryngograph signal provides a reliable measure of vocal fold contact and that rapid closure of the vocal cords causes a rapid rise in the laryngograph signal. The instants of glottal closure thus coincide with positive peaks in the differentiated laryngograph signal, $dL(t)$.

An advantage of using laryngograph signals to locate instants of glottal closure is that they can also be used to accurately determine period-by-period estimates of pitch and voicing decision (Krishnamurthy & Childers 1986). Voicing decisions are made based on the structure of the laryngograph signal. During unvoiced segments, when the vocal cords are out of contact, the signal from the laryngograph is high frequency noise generated by the internal electronics of the device whereas voiced segments show distinct pulses at the pitch period. The relative opening and closing durations for pulse models can also be measured from the spacing of the negative and positive peaks in the derivative of the laryngograph signal. These values can either be used directly (Krishnamurthy & Childers 1986, Howard & Breen 1989), or form the initial estimates for an analysis-by-synthesis search (Milenkovic 1986, Lobo & Ainsworth 1992).

	VOWEL DATA (Chandra & Lin 1974)	**NASALISED VOWEL DATA** (Fujisaki & Ljungqvist 1987)
Poles	$n_a = 10$, $A(q)$ set from data	$n_a = 12$, $A(q)$ set from data
Zeros Zeros	$n_b = 2$, $B(q) = 1 - q^{-1}$ (fixed), input $x(t)$ $n_b = 1$, $B(q) = 1$ (fixed), input $dx(t)$	$n_b = 4$, $B(q)$ set from data, input $x(t)$ $n_b = 3$, $B(q)$ set from data, input $dx(t)$
Pitch of $x(t)$	120Hz	120Hz
SNR	20dB	20dB

Table 3.1: Summary of source-filter parameters for generation of synthetic data.

Without Pre-emphasis	**Vowels**	**Nasalised Vowel**
LP (covariance)	$n_a = 13$	$n_a = 17$
ARX or OE ARX or OE	$n_a = 10, n_b = 1$ input $dx(t)$ $n_a = 10, n_b = 2$ input $x(t)$	$n_a = 12, n_b = 4$ input $x(t)$ $n_a = 12, n_b = 3$ input $dx(t)$
With Pre-emphasis	**Vowels**	**Nasalised Vowel**
LP (covariance)	$n_a = 12$	$n_a = 16$
ARX or OE ARX or OE	$n_a = 10, n_b = 1$ input $dx(t)$ $n_a = 10, n_b = 2$ input $x(t)$	$n_a = 12, n_b = 4$ input $x(t)$ $n_a = 12, n_b = 3$ input $dx(t)$

Table 3.2: Summary of model orders used in analysis of synthetic speech data.

3.5.3 The Glottal Excitation Model Used in This Work

Due to the difficulties and limitations of inverse-filtering techniques, a pulse based representation of the excitation was used in this work. All voiced excitation waveforms $x(t)$, including those used for the generation of synthetic speech data, were based on the Rosenberg pulse model. This pulse shape was selected for its simplicity and good perceptual performance. Pulse amplitudes, α, were fixed at unity and suitable input scaling determined by the vocal tract models. The opening and closing durations for pulses were fixed at $0.4T_p$, and $0.2T_p$ respectively, where T_p is the required pitch of the excitation. These values fall within the preferred relative durations reported by Rosenberg (1971). For synthetic voiced speech, an excitation was simply generated by a sequence of pulses at the required pitch period, T_p. For real speech, a glottal volume velocity waveform for an utterance was generated as a sequence of pulses, in which the instants of closure of the pulses were aligned with the peaks in $dL(t)$, as illustrated in Fig. 3.5. The local pitch period, T_p, was determined from the spacing between the peaks in $dL(t)$. The first difference of the glottal volume velocity waveform, where required, was calculated by

$$dx(t) = x(t) - x(t-1) \tag{3.50}$$

3.6 Comparison of Different Analysis Methods Using Synthetic Data

In this section, synthetic speech data is used to illustrate the differences in LP, ARX and OE models of the vocal tract, to show the effect of using the different source-filter configurations in Fig. 3.3, and to show the effects of pre-filtering, misalignment of the excitation and a reduction in signal-to-noise ratio on the resulting estimates of the vocal tract transfer function. Segments of synthetic speech data, y_s, were generated using

$$y_s(t) = B(q)/A(q)x(t) + e(t) \tag{3.51}$$

The excitation, $x(t)$, was provided by a fixed length sequence of Rosenberg pulses at a pitch of 120Hz, which was set up as described in section 3.5.3. The variance of the zero mean Gaussian noise term, $e(t)$, was adjusted to give a nominal SNR of 20dB. The poles and zeros for $B(q)$ and $A(q)$ were set from typical formant and bandwidth values for synthetic vowels and a synthetic nasalised vowel, using Eqns. (3.45) and (3.46) to calculate the complex conjugate pair associated with each resonance or anti-resonance. Assuming a density of 1 formant/kHz, the sampling interval for synthetic data with N_{F} formants is given by $T = 1/(2N_{\mathrm{F}})$. This sampling period is used to set the required pitch of the excitation. Where stated, the pre-emphasis filter $P(q) = 1 - 0.95q^{-1}$ was used.

For synthetic vowels, formant and bandwidth values for nine English vowels were taken from Chandra & Lin (1974). This data gives 5 formant locations for each vowel, and is thus assumed to represent speech sampled at 10kHz. Chandra & Lin (1974) used the formant and bandwidth values to synthesise synthetic vowels using a formant synthesiser that was excited by a 2-pole model of the glottal source. They did not include the effect of lip radiation. In this implementation, both configurations for incorporating lip radiation effects, Fig 3.3, were used to generate synthetic vowels. $B(q)$ was set to $B(q) = 1 - q^{-1}$ for models excited by $x(t)$ and $B(q) = 1$ for models excited by $dx(t)$. The spectra of data generated in this manner has a spectral shift of 6dB/octave compared to that generated by Chandra & Lin (1974).

Values for 6 formants (an additional formant is introduced by nasal coupling) and an anti-resonance for the synthetic nasalised vowel /$\tilde{\epsilon}$/, were taken from Fujisaki & Ljungqvist (1987). They used a terminal analog synthesiser excited by the first difference of a glottal volume velocity wave model (different to that used here) to generate synthetic data. To realise a similar synthesis, $B(q)$ and $A(q)$ were set from the given data and the filter was excited by $dx(t)$, an excitation at 120Hz. A summary of parameter settings for generation of synthetic vowel and nasalised vowel, data from the formant and bandwidth data given by Chandra & Lin (1974) and Fujisaki & Ljungqvist (1987), is given in Table 3.1.

The synthetic speech was subjected to analysis by LP, ARX (equation error) and OE (output error) models and the resulting vocal tract transfer functions and noise (output error or synthesis error) spectra examined. A summary of the model orders used for analysis is given in Table 3.2. For synthetic data generated without noise, $e(t) = 0$, both ARX and OE models are able to identify the transfer function exactly, if the correct values of n_a, n_f and n_b are used for the analysis. Comparison of Eqns. (3.15) and (3.51) shows that with the addition of noise, the synthetic data can be considered the noisy simulation from an underlying OE model. Assuming that n_b and n_f are chosen correctly for analysis, the OE model gives an unbiased estimate of the true parameters because the true model can be described by the OE family of models. The performance, however, is dependent on the SNR and deteriorates as the variance of the noise is increased. In contrast, an ARX model with the same number of poles and zeros as the synthetic data system, will give a biased estimate of the vocal tract parameters because the true underlying model does not lie within the set described by an ARX model of this order. The ARX model forces the noise spectrum to be $1/A(e^{j\omega}, \boldsymbol{\theta})$, but the spectrum of the noise that was used to generate the data is flat. Thus a MSE comparison of ARX and OE models on synthetic data would be unfair.

3.6.1 Noise Model and Transfer Function Estimate

In section 3.3.1, the spectral interpretation of the ARX and OE models was given, and the differences between the noise spectrum and frequency bias of the transfer function estimate for each model was discussed. A similar spectral interpretation for linear prediction models was given in section 2.3.2. As shown by Eqn. (2.9), the frequency response of the inverse linear prediction filter provides a smoothed estimate of the magnitude spectrum of the speech. This is illustrated in Fig. 3.6(a) for the synthetic vowel in 'hod', and in Fig. 3.9(a) for the nasalised vowel. The LP spectrum shows a spectral tilt due to the combined effect of the source and lip radiation factors which are lumped into the frequency response of the vocal tract model. For the nasalised vowel, it was found that increasing the linear prediction model order by two additional poles gave adequate modelling of the anti-resonance. For linear prediction, the spectral matching between the model frequency response and the spectral envelope of the speech is performed uniformly across the frequency range of interest[6].

For black-box models excited by $x(t)$, the transfer function estimate, $H(q)$, combines both vocal tract and lip radiation characteristics. For models excited by $dx(t)$, $H(q)$ represent the vocal tract only, and the resulting frequency response has a $-6dB/$octave spectral shift compared to the transfer function estimate obtained using $x(t)$. For the

[6] When considering the spectral matching performance, it is also necessary to take into account the spectral trend of $\|Y(e^{j\omega})\|^2$ (see section 2.3.2). Lower values of $\|Y(e^{j\omega})\|^2$ contribute less to the mean-square error criterion and the spectral match at frequencies where $\|Y(e^{j\omega})\|^2$ is lower are therefore less accurate. A similar argument applies to the match between $\hat{H}(e^{j\omega})$ and $H(e^{j\omega}, \boldsymbol{\theta})$.

synthetic vowel in 'hod', this is illustrated by comparison of Figs. 3.6(b) and 3.6(c). The corresponding illustrations for the nasalised vowel are Figs. 3.9(b) and 3.9(c).

The frequency biases, $Q(\omega, \boldsymbol{\theta})$, for the transfer function estimates of Fig. 3.6(c) are shown in Figs. 3.7(a) and 3.7(b), for ARX and OE models respectively. For ARX models,

$$Q_{arx}(\omega, \boldsymbol{\theta}) = \|X(e^{j\omega})\|^2 \|A(e^{j\omega}, \boldsymbol{\theta})\|^2$$

The dependence of $Q(\omega, \boldsymbol{\theta})$ on $A(e^{j\omega}, \boldsymbol{\theta})$ is seen in Fig. 3.7(a) and results in de-emphasis of the transfer function fit in the region of the formants, especially the first and second formant. De-emphasis can result in poor estimates of formants in these regions, as seen by the lack of a third formant in the ARX transfer function estimate for the nasalised vowel, Fig. 3.9(c). This is due to strong de-emphasis of the second and third formant, shown by the corresponding frequency bias in Fig. 3.10(a).

For OE models, $Q_{oe}(e^{j\omega}, \boldsymbol{\theta}) = \|X(e^{j\omega})\|^2$. The frequency bias of the transfer function estimate depends only on the spectrum of the excitation, and falls off more rapidly than that of ARX models, as illustrated by comparison of Figs. 3.7(a) and 3.7(b) for the synthetic vowel, or Figs. 3.10(a) and 3.10(b) for the nasalised vowel. As a result, the frequency bias for the ARX model penalises high frequency misfit much more than the frequency bias for the OE model, resulting in a better transfer function estimate at higher frequencies.

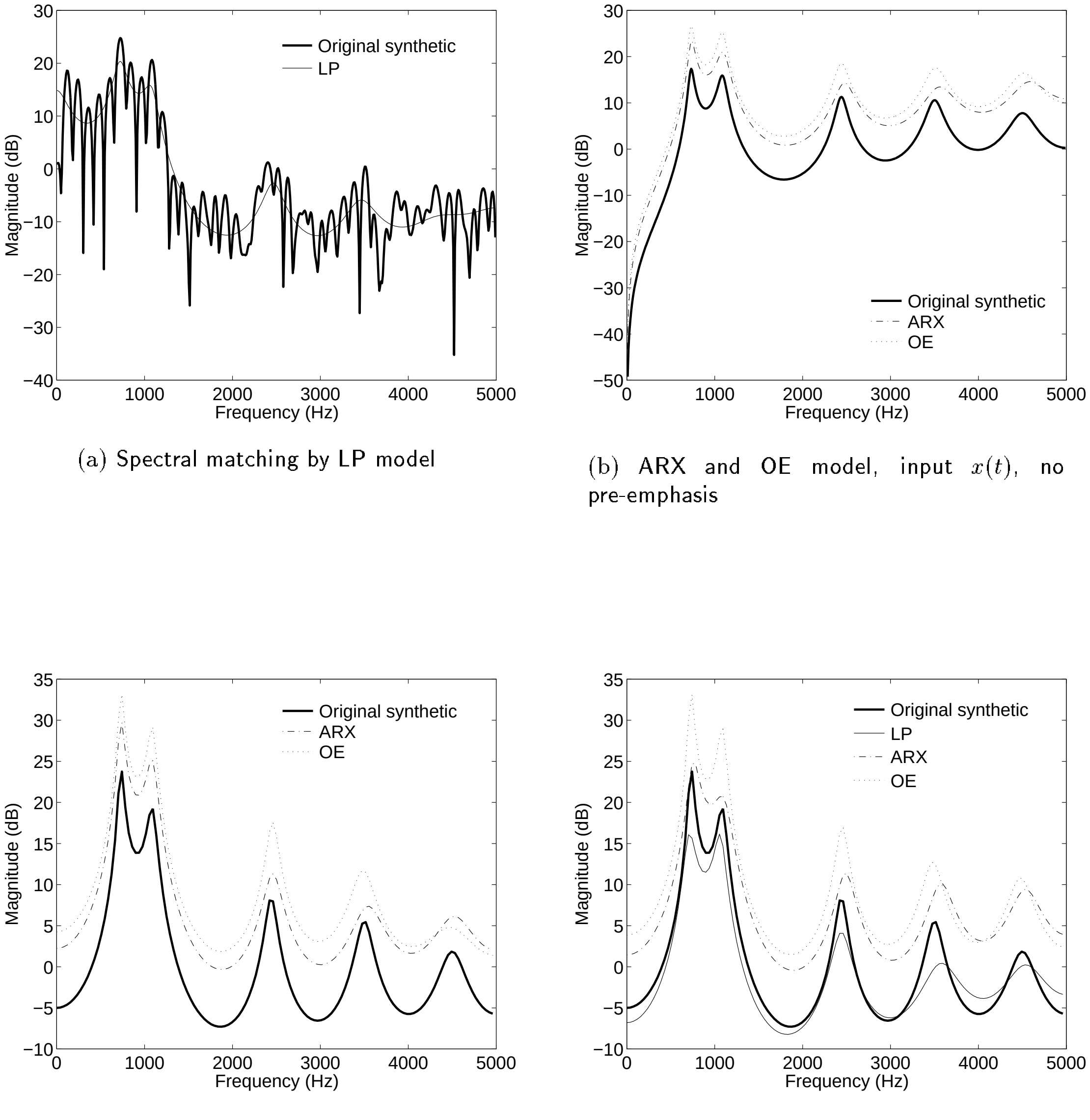
In section 3.3.1, the parameter estimation procedure for ARX models was described as a trade-off between matching the transfer function of the model to the ETFE, with frequency weighting $Q(\omega, \boldsymbol{\theta})$, and matching the envelope of the synthesis error spectrum, $\|\Phi_{\mathrm{ER}}(\omega, \boldsymbol{\theta})\|^2$, to the model noise spectrum, $\|N(e^{j\omega}, \boldsymbol{\theta})\|^2 = \|1/A(e^{j\omega}, \boldsymbol{\theta})\|^2$. This spectral match is shown in Fig. 3.8(a), in which the spectral envelope of the synthesis error matches $\|1/A(e^{j\omega}, \boldsymbol{\theta})\|^2$. For OE models, the parameter estimation procedure places no such constraint on the spectrum of the synthesis error, which will tend toward that of the true system noise. This is seen in Fig. 3.8(b), in which the flat spectrum is due to the fact that the synthetic data was generated with addition of Gaussian noise.

3.6.2 Effect of Pre-emphasis on Estimation of Transfer Function

For the nine synthetic vowels given in Chandra & Lin (1974), Fig. 3.11 shows the effect of pre-emphasis on the accuracy of estimation of frequencies and bandwidths. For each formant (bandwidth), the mean absolute estimation error, averaged over all 9 synthetic vowels, is shown. The absolute estimation error for quantity, $\hat{x}$, with true value x_0, is given by

$$E = \frac{|\hat{x} - x_0|}{x_0} \tag{3.52}$$

A slightly higher SNR (30dB) was used to generate synthetic data for these models, to reduce convergence to local minima by OE models.



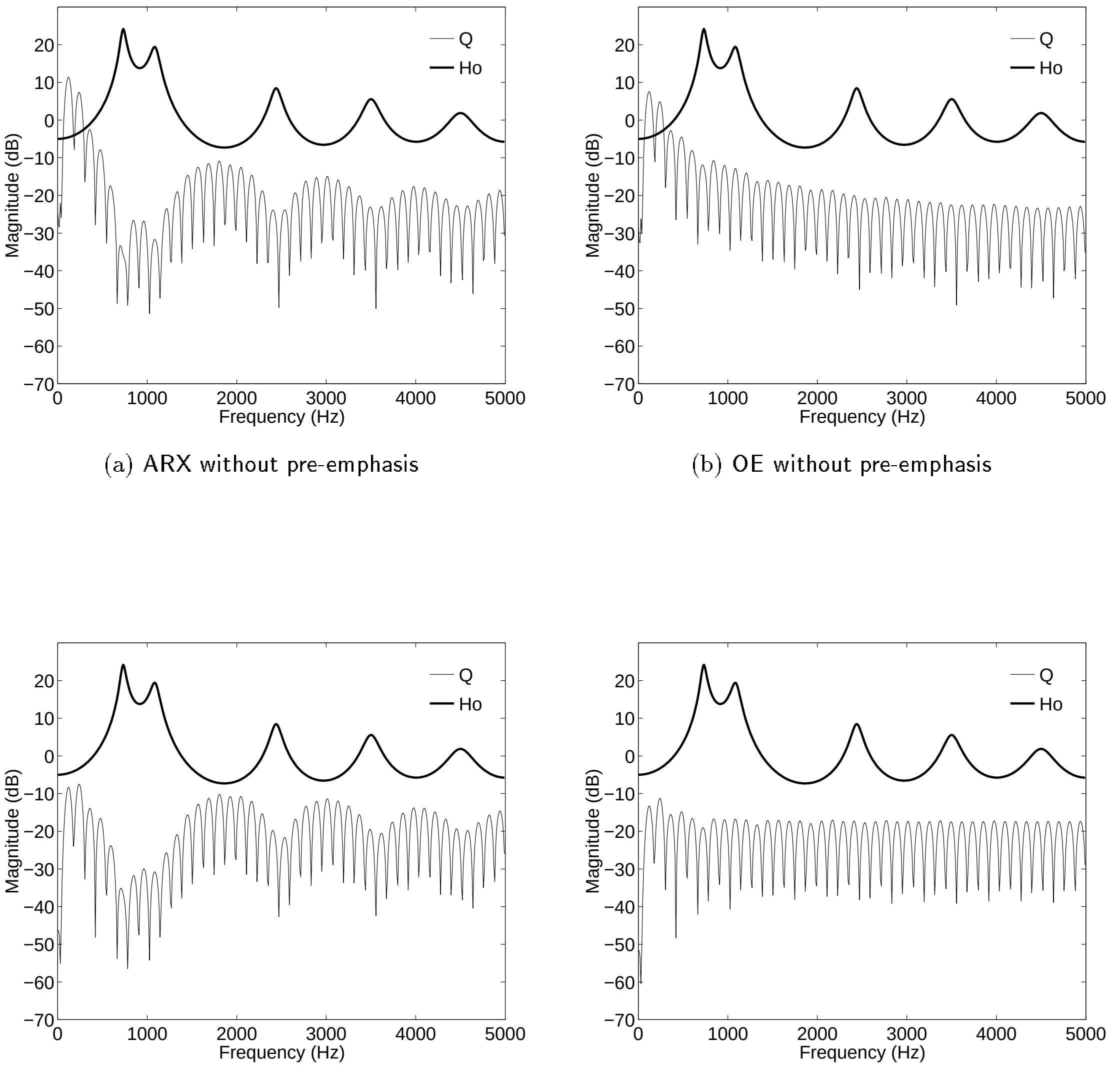
(a) Spectral matching by LP model

(b) ARX and OE model, input $x(t)$, no pre-emphasis

(c) ARX and OE model, input $dx(t)$, no pre-emphasis

(d) ARX and OE model, input $dx(t)$, with pre-emphasis

Figure 3.6: Estimates of the vocal tract transfer function for the synthetic vowel in 'hod'.



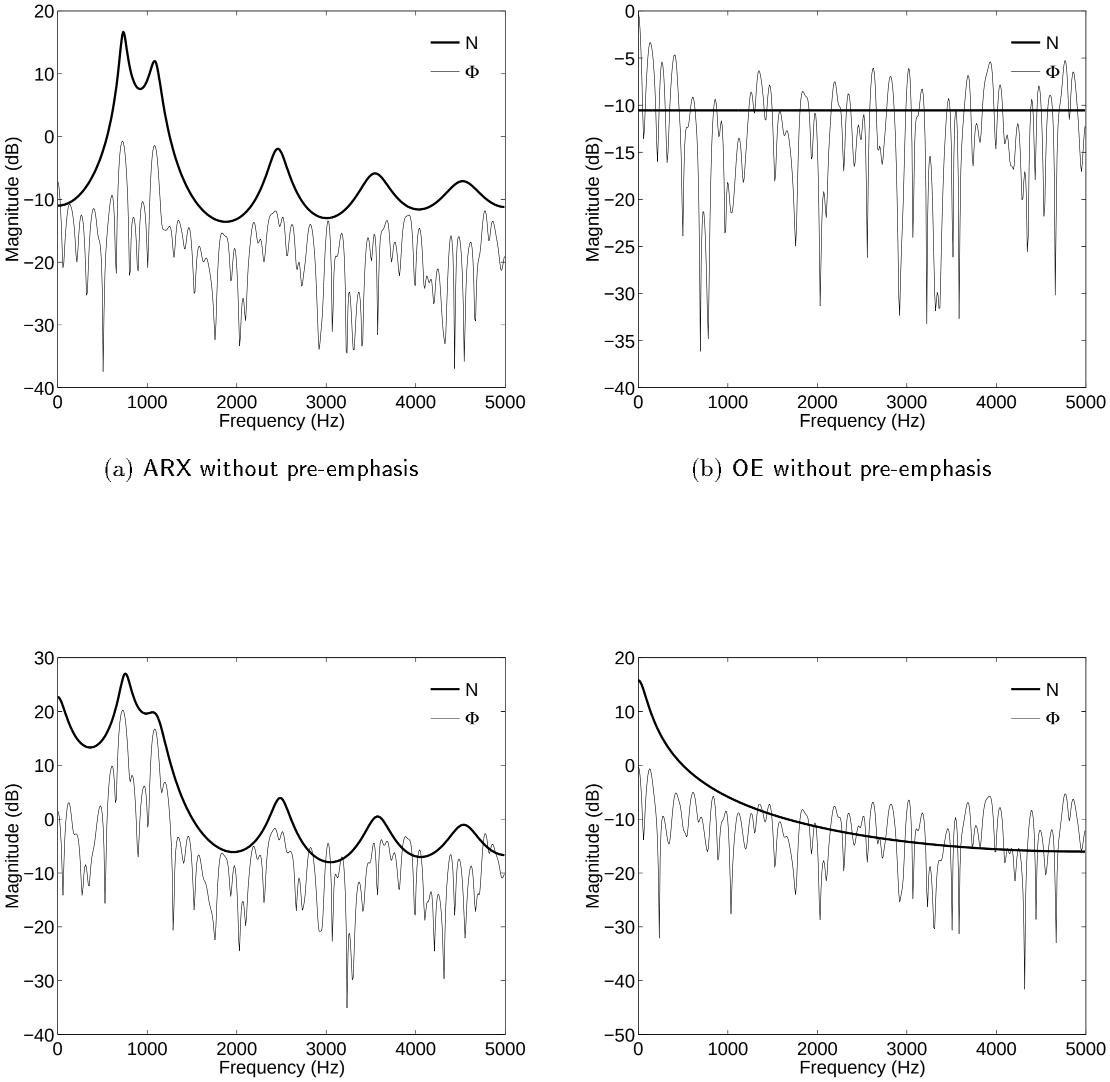
(a) ARX without pre-emphasis

(b) OE without pre-emphasis

(c) ARX with pre-emphasis

(d) OE with pre-emphasis

Figure 3.7: Frequency bias functions (Q) for the transfer function estimates shown in Fig. 3.6(c) (without pre-emphasis) and Fig. 3.6(d) (with pre-emphasis). The transfer function for the original synthetic data (Ho) is also shown.



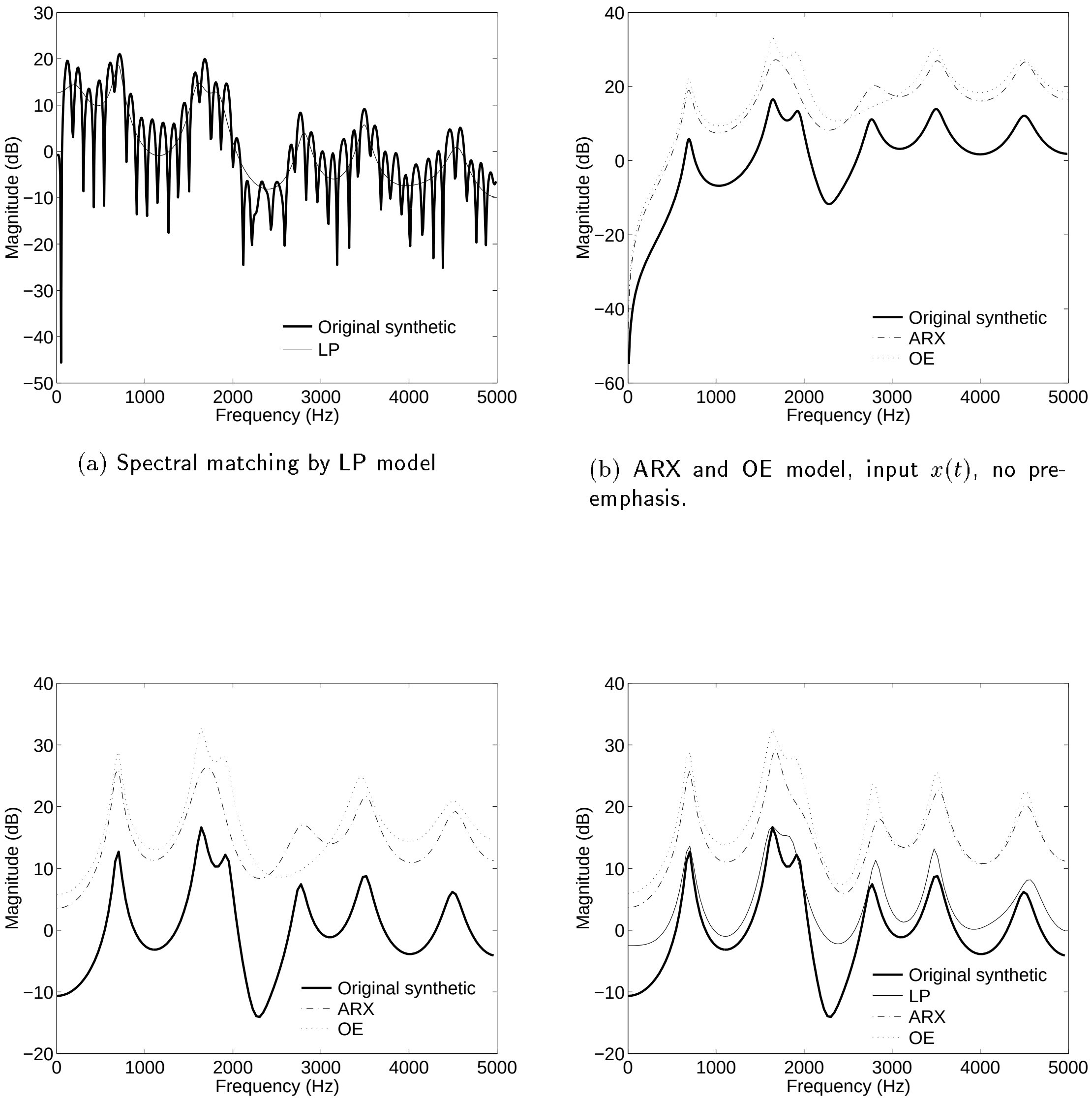
(a) ARX without pre-emphasis

(b) OE without pre-emphasis

(c) ARX with pre-emphasis

(d) OE with pre-emphasis

Figure 3.8: Noise models (N) and spectra of synthesis errors (Φ) for the models shown in Fig. 3.6(c) (without pre-emphasis) and Fig. 3.6(d) (with pre-emphasis).



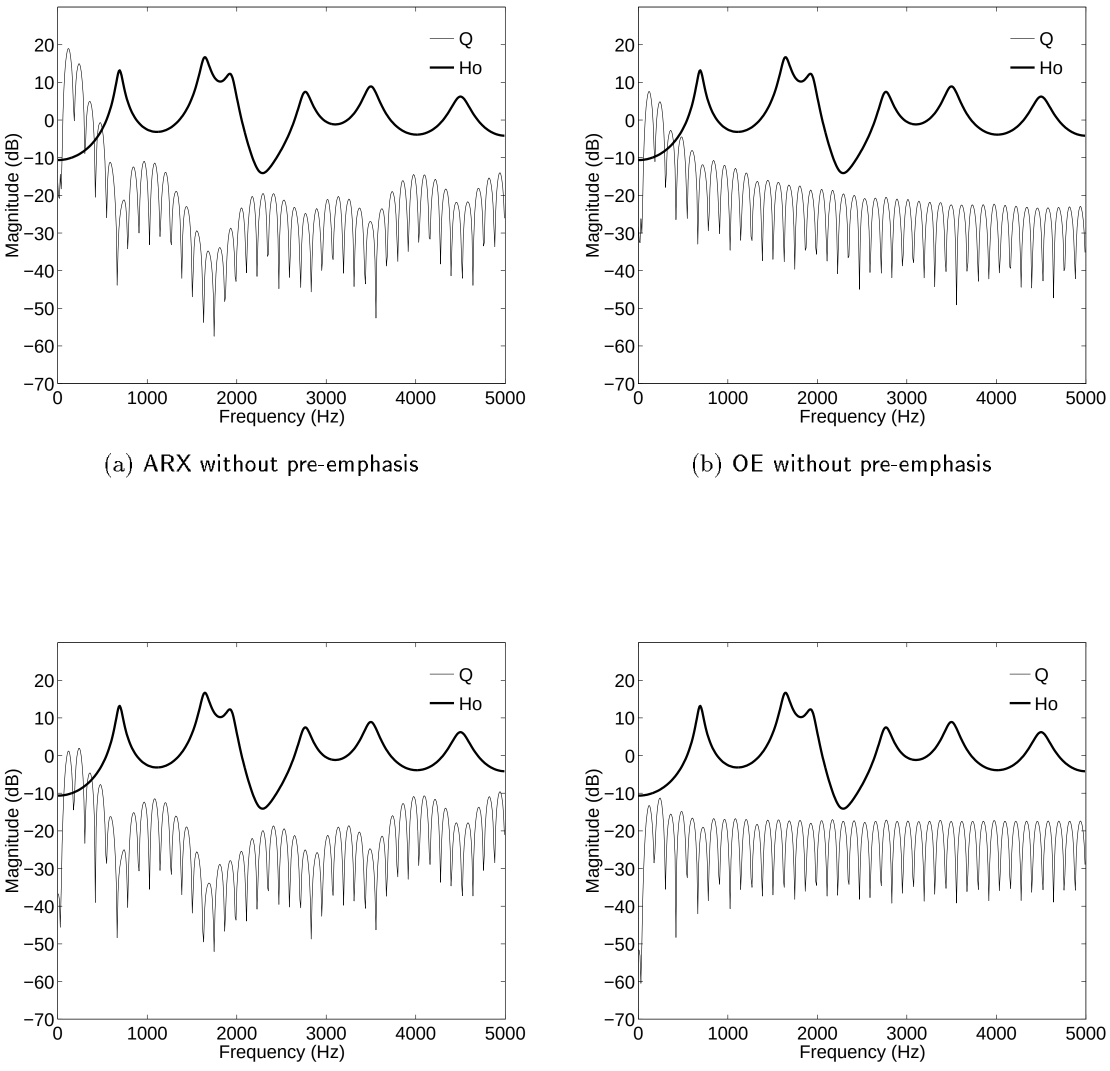
(a) Spectral matching by LP model

(b) ARX and OE model, input $x(t)$, no pre-emphasis.

(c) ARX and OE model, input $dx(t)$, no pre-emphasis

(d) ARX and OE model, input $dx(t)$, with pre-emphasis.

Figure 3.9: Estimates of the vocal tract transfer function for the synthetic nasalised vowel $/\tilde{\epsilon}/$.



(a) ARX without pre-emphasis

(b) OE without pre-emphasis

(c) ARX with pre-emphasis

(d) OE with pre-emphasis

Figure 3.10: Frequency bias functions (Q) for the transfer function estimates shown in Fig. 3.9(c) (without pre-emphasis) and Fig. 3.9(d) (with pre-emphasis). The transfer function for the original synthetic data (Ho) is also shown.

For linear prediction, there is a clear improvement in the estimation of the first formant and bandwidth when pre-emphasis is used. The effect of pre-emphasis is to remove the -6dB/octave spectral contribution resulting from the combined effect of lip radiation and source. This gives improved estimation of the vocal tract transfer function at lower frequencies. The frequency response of the linear prediction model obtained with pre-emphasis can be regarded as representing that of the vocal tract only. This is illustrated by comparison of Figs. 3.6(a) and 3.6(d), for the synthetic vowel in 'hod', and by comparison of Figs. 3.9(a) and 3.9(d), for the synthetic nasalised vowel /$\tilde{\epsilon}$/. With pre-emphasis, the spectral matching between the model frequency response and the spectral envelope of the pre-emphasised speech is still performed uniformly across the frequency range of interest, but the relative magnitudes of the higher frequencies are enhanced by pre-emphasis and therefore have a greater effect on the error criterion.

With black-box models, both the speech and excitation are pre-emphasised. Pre-emphasis improves the numerical stability of the parameter estimation because the dynamic range of the data is reduced (Deller et al. 1993). Pre-emphasis does not alter the nature of the underlying transfer function, but the limiting values of the parameters are not the same because pre-emphasis changes the model noise and the frequency bias of the transfer function estimate, as described in section 3.3.3.

For ARX models, Figs. 3.6(c) and 3.6(d) illustrate the effect of pre-emphasis on the estimation of the vocal tract transfer function, for the synthetic vowel in 'hod'. With pre-emphasis, there is improved estimation of higher frequencies and bandwidths, but poorer estimation of the first and second formants. This was generally the case for the synthetic vowels, and is illustrated in Figs. 3.11(c) and 3.11(d), by the higher absolute error in estimation of formants and bandwidths at lower frequencies, and by the lower absolute error in estimation of higher frequencies, when pre-emphasis is used. As illustrated by comparison of Fig. 3.9(c) and 3.9(d), pre-emphasis improved estimation of the anti-resonance of the nasalised vowel by the ARX model, but the third formant was still not estimated.

For OE models, the accuracy in estimation of the formants and bandwidths is shown in Figs. 3.11(e) and 3.11(f). Pre-emphasis improves the estimation of formants and bandwidths at all frequencies, and at higher frequencies in particular. For the synthetic vowel in 'hod', this is illustrated by comparison Figs. 3.6(c) and 3.6(d), which shows improved estimation of the formant at 4.5 kHz when pre-emphasis is used. Similarly, for the nasalised vowel, comparison of Figs. 3.9(c) and 3.9(d) shows improved estimation of the bandwidths of higher formants using pre-emphasis.

As discussed in section 3.3.3, pre-emphasis alters the model noise and the frequency bias of the transfer function estimate. The noise model becomes

$$\tilde{N}(q) = N(q)/P(q)$$

and the frequency bias becomes

$$\tilde{Q}(\omega, \boldsymbol{\theta}) = Q(\omega)\|P(e^{j\omega})\|^2$$

The pre-emphasis filter, $P(q)$, has a high-pass characteristic which reduces the influence of the lower frequencies on the transfer function fit and results in a frequency bias which is more evenly distributed across the frequency range. This is illustrated in Fig. 3.7, for the synthetic vowel in 'hod', and in Fig. 3.10 for the nasalised vowel. As a result, pre-emphasis improves the estimation of higher formants by both ARX and OE models, as illustrated by comparison of Figs. 3.6(c) and 3.6(d) for the synthetic vowel in 'hod', and by comparison of Figs. 3.9(c) and 3.9(d) for the nasalised vowel.

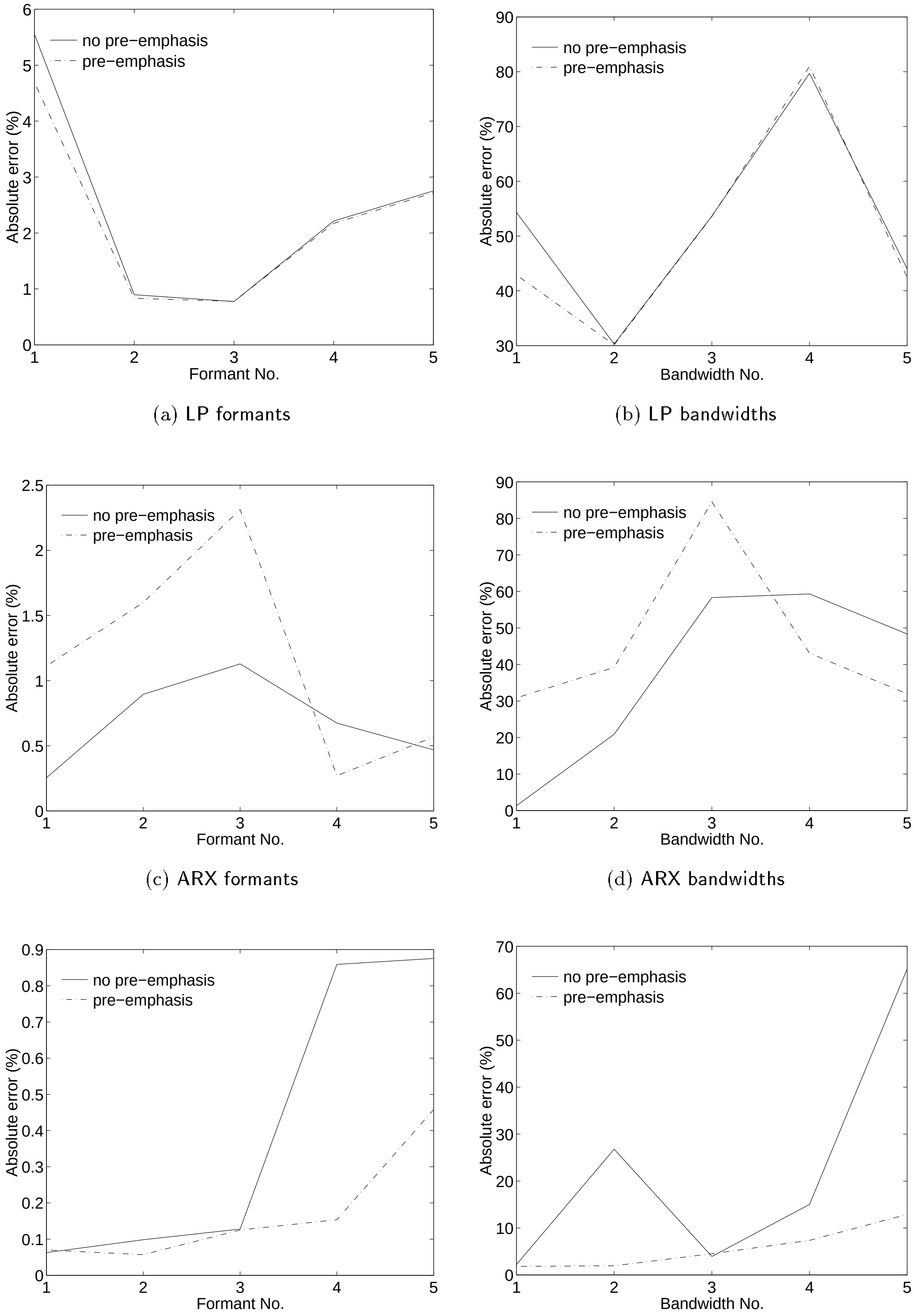
For the ARX model, the parameter estimation procedure also requires that the spectral envelope of the synthesis error match that of the new model noise. This is illustrated in Fig. 3.8(c), for the synthetic vowel in 'hod'. The constraints imposed by the new noise model can lead to poorer estimation of lower frequencies, Fig. 3.6(d), because the fit between the spectrum of the synthesis error and that of the new model noise, shown in Fig. 3.8(c), is achieved at the expense of the transfer function fit in the lower frequency region. For the OE model, the spectrum of the new model noise is $1/P(e^{j\omega})$. As this is independent of the model parameters, the spectrum of the synthesis error still tends to that of the true system noise. This is illustrated in Fig. 3.8(d), in which the spectrum of the synthesis error remains flat, despite the change in shape of the spectrum of the model noise. With pre-emphasis, the OE model is still able to give a good estimate of the transfer function, whereas the ability of the ARX model to provide a solution is limited by the need to also fit the spectrum of the synthesis error to the new noise model.

For the synthetic nasalised vowel, Fig. 3.9(c) shows poor estimates of the vocal tract transfer function by both ARX and OE models without pre-emphasis, especially in the region of the anti-resonance. The low values of $H(e^{j\omega})$ in this region contribute much less to the error criterion. In comparison, Fig. 3.9(d) shows that the transfer function estimates obtained with pre-emphasis are greatly improved for both models.

3.6.3 Effect of Noise on Estimation of Transfer Function

The effect of reducing the SNR on the estimation of the vocal tract transfer function is illustrated in Fig. 3.12, for transfer function estimates for the synthetic vowel in 'hawed' (input $dx(t)$). For OE models, 20 iterations of the Gauss-Newton algorithm were performed. The initial estimate of the parameters was obtained from an ARX model of the data. At lower SNR, this initial estimate may be poor.

As the SNR is reduced, the spectrum of noisy speech becomes flatter due to high frequency energy from the noise. This is seen in the estimates of the spectral envelope made by linear prediction models, and the transfer functions estimated by the ARX model, Figs. 3.12(a), 3.12(c) and 3.12(e), where the estimates become progressively flatter at higher frequencies. Comparison of these results with those using pre-emphasis, Figs. 3.12(b), 3.12(d) and 3.12(f), shows that in increased noise, pre-emphasis may not be beneficial because it enhances the higher frequencies where the noise energy is predominant.



(a) LP formants

(b) LP bandwidths

(c) ARX formants

(d) ARX bandwidths

(e) OE formants

(f) OE bandwidths

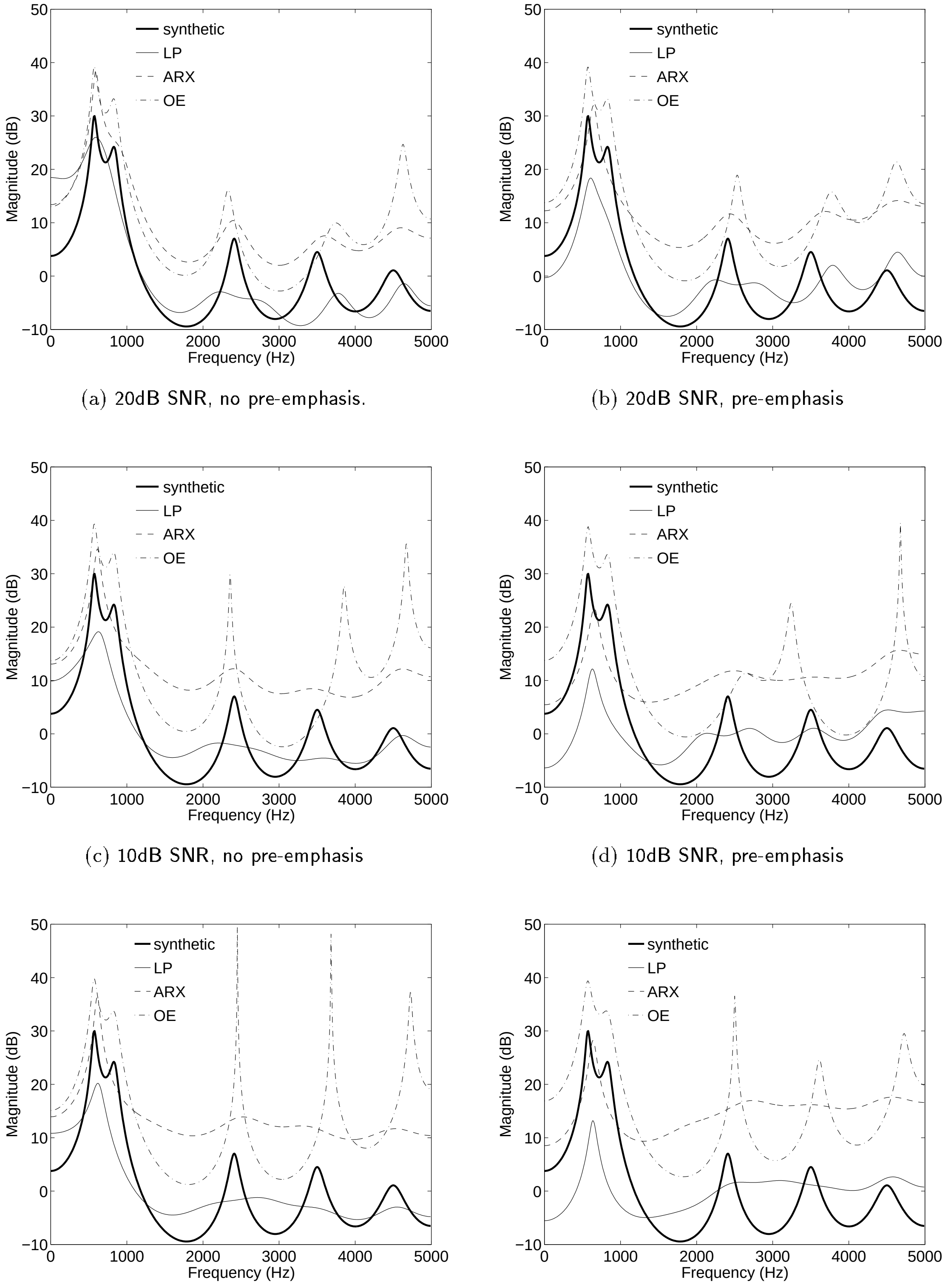
Figure 3.11: Absolute (%) error in estimation of formants and bandwidths of synthetic vowel data.

For OE models, the performance is affected by the presence of local minima in the error criterion. Local minima were more evident at lower SNR because ARX models provided poor initial estimates of parameters for OE models in these cases. As a result, the estimates of the transfer function by OE models were poor, as illustrated in Figs. 3.12(a), 3.12(c) and 3.12(e). With pre-emphasis, convergence to different local minima occurred, which is illustrated by comparing Figs. 3.12(c) and 3.12(d), for example. Poor local minima often exhibit high frequency formants with unnaturally narrow bandwidths. These are caused by poles close to the unit circle. The bandwidths tend to become narrower for converging solutions after a larger number of iterations of the Gauss-Newton algorithm. This effect is illustrated in Fig. 3.13, which shows varying stages in the estimation of the transfer function of the OE model shown in Fig. 3.12(e). The initial ARX model gave a poor estimate of the transfer function, Fig. 3.13(a), and after a single iteration this estimate was improved, Fig. 3.13(b). After 5 iterations, the estimate of the first and second formants was greatly improved, but as the number of iterations was increased further, the bandwidths of the higher formants became unnaturally narrow. Fig. 3.13(f) shows the solution at convergence. This type of behaviour was also noted by Lim & Oppenheim (1978) in their work on the estimation of linear prediction parameters from noisy speech data.

3.6.4 Effect of Misalignment of Excitation on Estimation of Transfer Function

Cheng & O'Shaughnessy (1989) have shown that correct alignment between the instants of glottal closure in the excitation signal and the true instants of glottal closure for the original speech signal is necessary for accurate estimation of vocal tract parameters. To determine how misalignment affects the estimates of the vocal tract transfer function, the glottal volume velocity waveform used in analysis was offset by k samples from that used to generate the synthetic data. Fig. 3.14 shows the effect of varying k on the estimates of the transfer function for the synthetic vowel in 'hud' (input $dx(t)$). A similar pattern of results was obtained using input $x(t)$.

Synthetic data with an SNR of 30dB was used so that the transfer function estimates by ARX and OE models were similar at $k = 0$, and the OE parameter estimation did not converge to a local minimum. Table 3.3 shows the effect of advance and delay misalignment on the mean SNR for the nine synthetic vowels of Chandra & Lin (1974). Misalignment results in a reduction in SNR, in both prediction and synthesis. In most cases, an advance misalignment results in a smaller degradation in SNR than the equivalent delay misalignment. Fig. 3.15 shows the resulting prediction errors for the ARX models and output errors for the OE models. Comparison with the original excitation signal, $dx(t)$, shows that with misalignment of the excitation, peaks occur in the prediction or output error at the true instants of glottal closure. The resulting syntheses from models using pre-emphasis are also shown. For ARX models, increased misalignment results in smaller values for $B(q)$, and a small amplitude synthesis. In comparison to the syntheses from



(a) 20dB SNR, no pre-emphasis.

(b) 20dB SNR, pre-emphasis

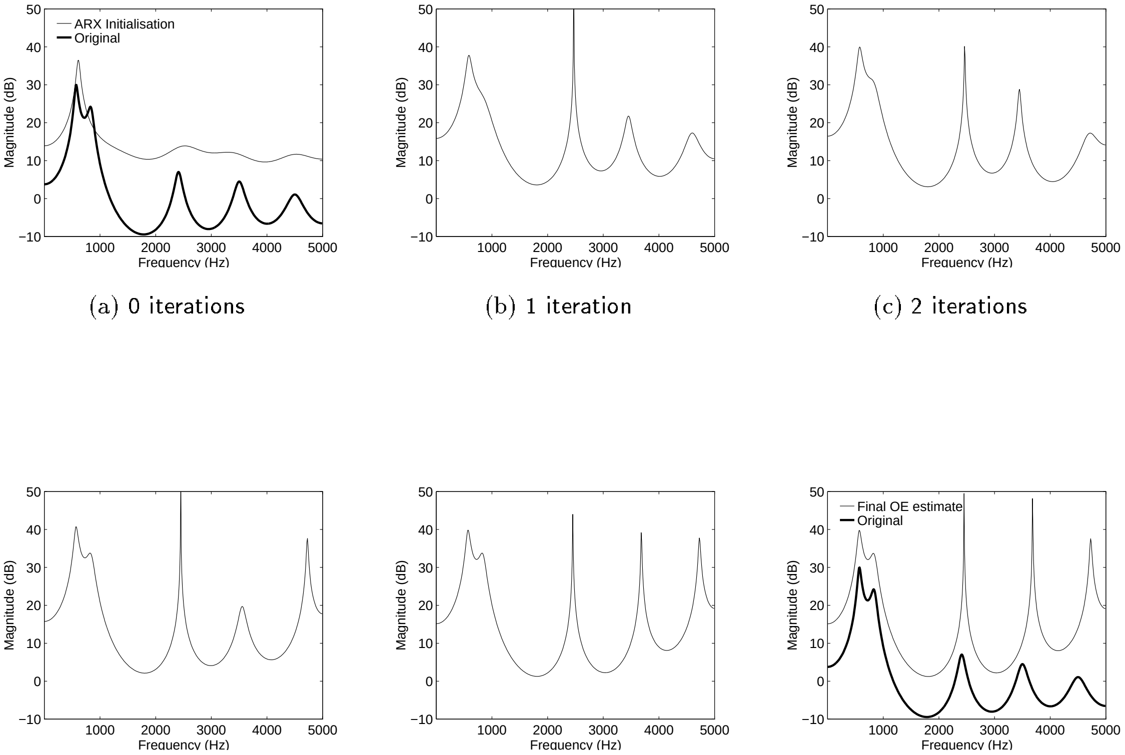
(c) 10dB SNR, no pre-emphasis

(d) 10dB SNR, pre-emphasis

(e) 5dB SNR, no pre-emphasis

(f) 5dB SNR, pre-emphasis

Figure 3.12: Effect of noise on estimation of transfer function of the synthetic vowel in 'hawed' (input $dx(t)$).



(a) 0 iterations

(b) 1 iteration

(c) 2 iterations

(d) 5 iterations

(e) 10 iterations

(f) 20 iterations

Figure 3.13: Varying stages in the estimation of the OE transfer function shown in Fig. 3.12(e). Note the formation of narrower bandwidths as the number of iterations of the Gauss-Newton procedure is increased.

ARX models, it can be seen that the OE models give a much better time-domain fit to the synthetic speech. This is also illustrated by a much higher synthesis SNR than for ARX models. However, Fig. 3.14(d) and 3.14(b) illustrate that both with and without pre-emphasis, the transfer function estimates by OE models are very poor when misalignment occurs. Although the estimates of the first (and sometimes second) formant are reasonable, estimates of higher formants fluctuate in position and often have very narrow bandwidths. In contrast, ARX models with pre-emphasis still give reasonably good estimates of the transfer function when misalignment occurs. Larger misalignments result in a flattening of the transfer function estimate in the region of the higher formants.

For the case of advance misalignment, ARX and OE models were also generated using a numerator order, $n_b = k + 1$ (as many zeros as the misalignment k). It was found that for both OE and ARX models, the additional zeros resulted in a transfer function estimate which was identical to that estimated with $k = 0$. Hence, inclusion of additional zeros in the transfer function, above those needed to model the vocal tract transfer function, allows the model to accommodate advance misalignment of the excitation from the true instants of glottal closure.

3.7 The Vocal Tract Modelling Framework

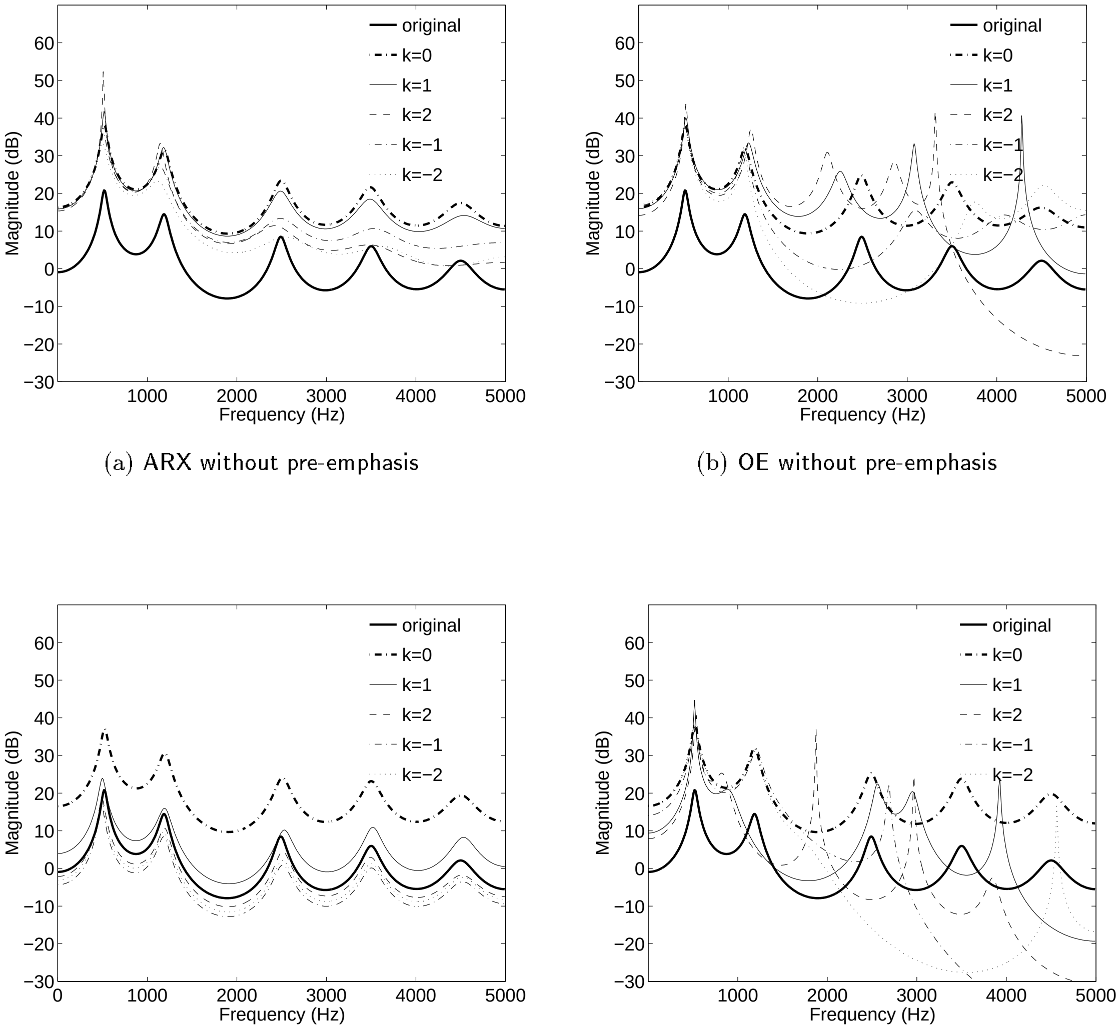
For analysis and synthesis of real speech utterances, the vocal tract modelling framework shown in Fig. 3.16 was used, and is based on the source-filter arrangement of Fig. 2.1.

For voiced speech, black-box ARX or OE models of the vocal tract were used. Pulse based representations of the glottal volume velocity waveform, $x(t)$, were set up as described in section 3.5.3. Fig. 3.16 illustrates the case where the vocal tract model is excited by the glottal volume velocity wave, $x(t)$. The alternative configuration illustrated in Fig. 3.3(b), in which models are excited by $dx(t)$, was also used. Laryngograph data recorded simultaneously with the speech was used to synchronise the pulses of the volume velocity waveform with the speech signal and also for accurate pitch extraction and voicing decisions.

Unvoiced speech was modelled by linear prediction analysis. In synthesis, the excitation was provided by a codebook of Gaussian sequences, scaled by Eqn. (2.4). A codebook size of 128 entries was used, which is slightly smaller than that typically used in CELP coding schemes (512-1024). The optimum codeword for synthesis was determined by synthesising speech for every entry in the codebook and selecting that entry which minimised the mean-square synthesis error.

For comparison of synthesis quality, a linear prediction framework was also set up, as described in chapter 2, in which both voiced and unvoiced speech was modelled by a linear prediction filter, and the voiced excited for synthesis was supplied by a scaled sequence of impulses at the desired pitch period.

Each speech utterance was processed by dividing the speech into a sequence of overlapping analysis windows at a particular frame rate and calculating a vocal tract model



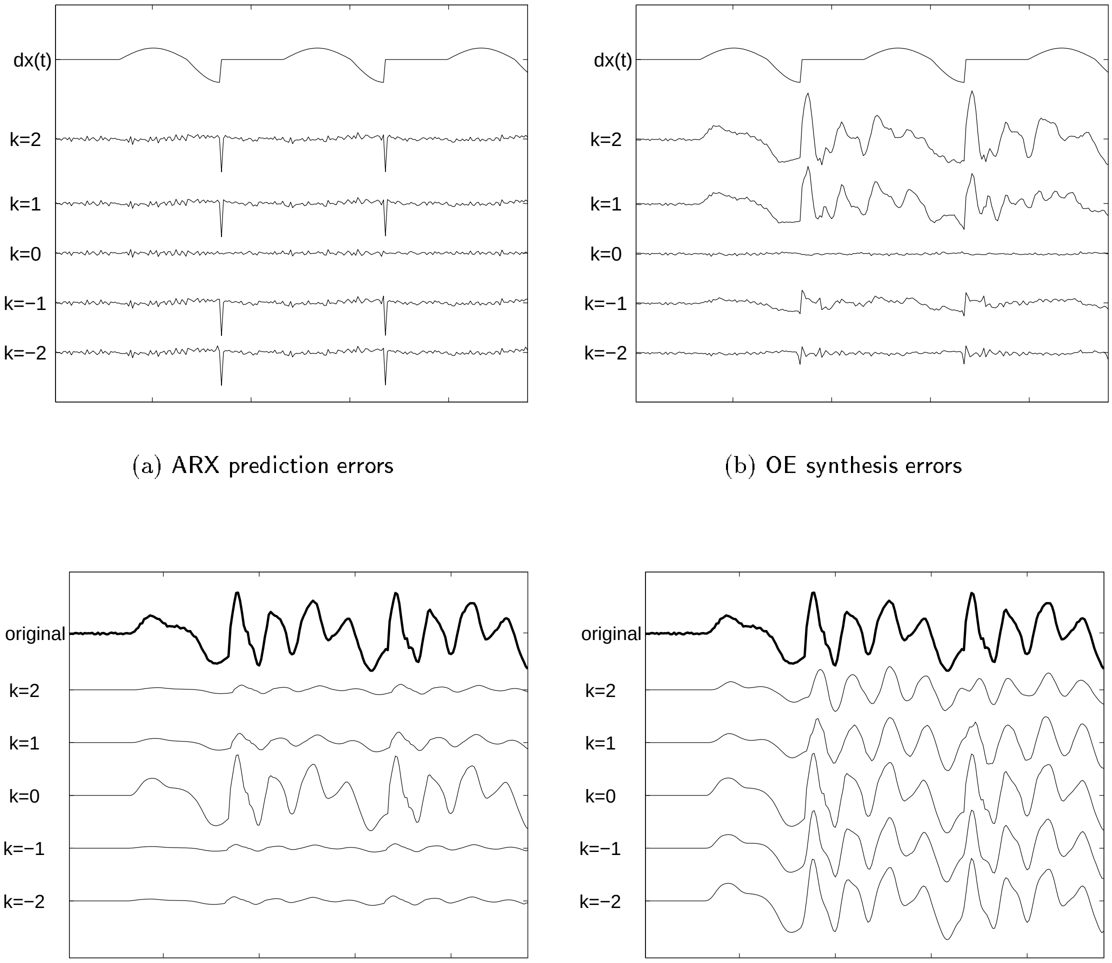
(a) ARX without pre-emphasis

(b) OE without pre-emphasis

(c) ARX with pre-emphasis

(d) OE with pre-emphasis

Figure 3.14: Effect of misalignment of excitation on transfer function estimates for the synthetic vowel in 'hud' (input $dx(t)$). Positive k indicates delay (the instants of glottal closure of the excitation in analysis occur after those used to generate the synthetic data).



(a) ARX prediction errors

(b) OE synthesis errors

(c) ARX syntheses

(d) OE syntheses

Figure 3.15: Effect of alignment errors on prediction error, synthesis error and synthesis for the synthetic vowel in 'hud' (input $dx(t)$, with pre-emphasis).

Alignment	ARX Prediction(dB)		ARX Synthesis (dB)		OE Synthesis (dB)	
	no pre	pre	no pre	pre	no pre	pre
k=0	**22.41**	**15.98**	**22.37**	**18.40**	**25.30**	**26.17**
k=1	14.84	6.89	6.60	1.39	10.83	5.08
k=-1	16.48	6.83	11.10	0.96	16.98	11.39
k=2	14.14	6.83	0.82	0.96	6.50	4.77
k=-2	15.25	6.81	7.61	0.63	13.68	11.85
k=4	13.55	6.82	-2.83	0.08	0.88	0.33
k=-4	14.32	6.80	4.87	0.38	9.52	5.00

Table 3.3: Effect of misalignment of excitation on prediction and synthesis SNR for ARX and OE models. The SNR are average values obtained for all nine synthetic vowels. Positive k indicates delay, 'pre' indicates pre-emphasis.

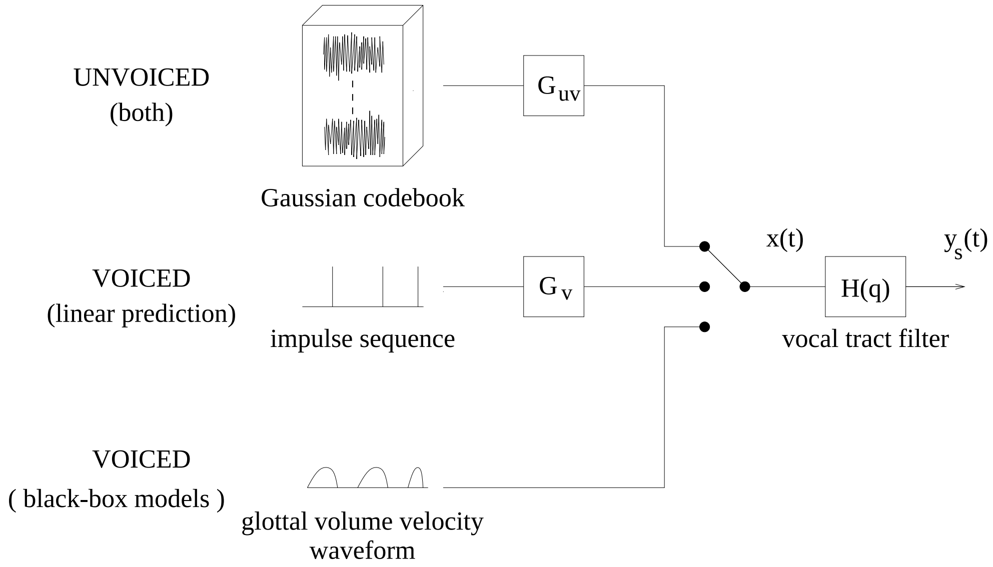
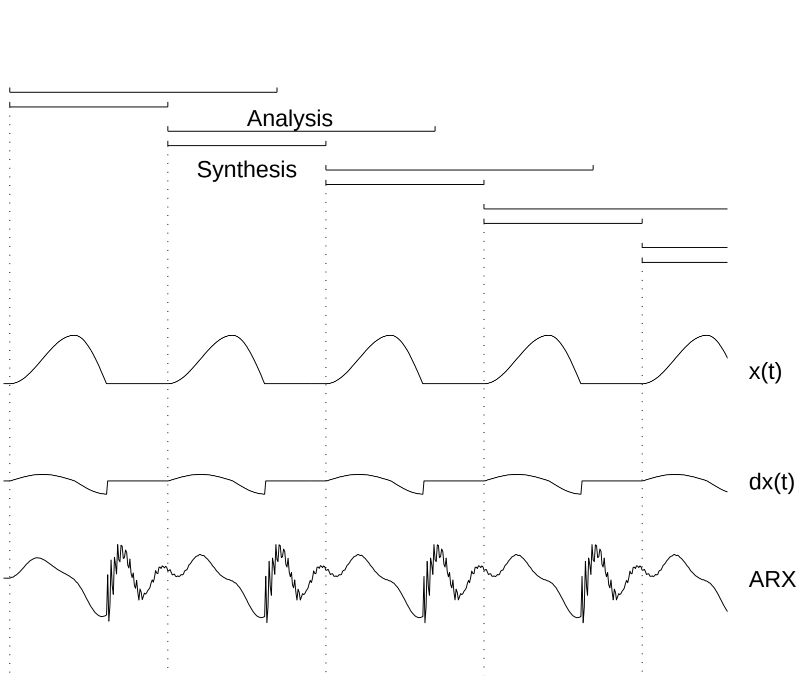


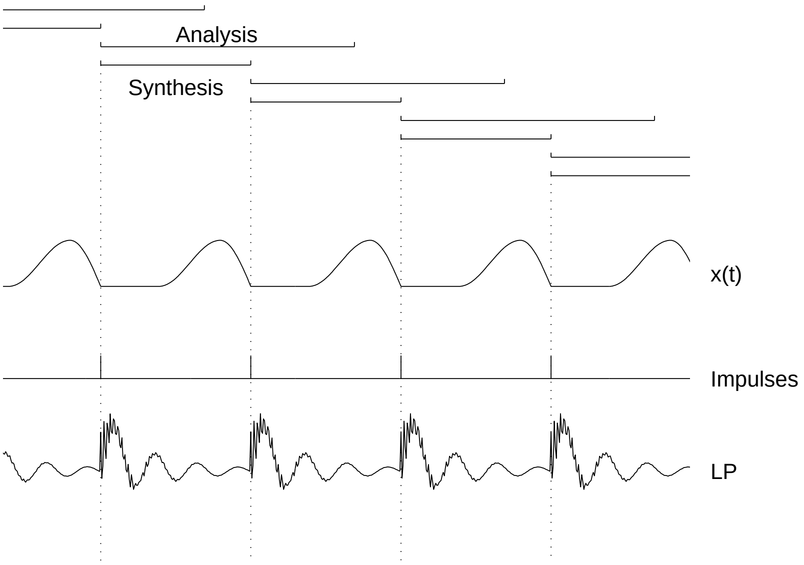
Figure 3.16: The vocal tract modelling framework for speech synthesis.

for each frame. The vocal tract model was then used to synthesise the speech for the duration of the frame rate, as illustrated in Fig. 3.17. An analysis window size of 25ms was used, which is a typical size used in linear prediction and CELP speech coders. This size gives sufficient samples for reliable estimation of model parameters and also allows for the inclusion of several excitation pulses in the analysis window. A frame rate of 7.5ms was chosen, which gave reasonably smooth pitch tracks (see section 3.8.3).

Poor quality synthetic speech is generated if the parameters vary considerably from frame to frame due to the effect of the filter memory when the parameters are updated. Using overlapping analysis windows helps to reduce this effect by taking account of the speech data ahead of the synthesis region, giving smoother parameter transitions from frame to frame. On unvoiced segments, a fixed frame rate was used and synthesis was performed over the duration of the frame rate. On voiced speech, the effect of the filter memory was minimised further by aligning the analysis and synthesis windows with the instants of glottal opening, as shown in Fig. 3.17(a). At these instants, the excitation signal has been zero for a short duration so the memory for $B(q)$ is zero and the filter memory for $A(q)$ (or $F(q)$) is small. For the linear prediction framework, the analysis windows were synchronised to the instants of glottal closure, as illustrated in Fig. 3.17(b). Using these procedures, model parameters are updated pitch-synchronously, which eliminates the frame rate buzz which is apparent when the parameters are updated at a fixed frame rate.



(a) ARX



(b) LP

Figure 3.17: Pitch-synchronous parameter update scheme. For black-box models, analysis windows were aligned with the instants of glottal opening, when the excitation has been zero for some time. For linear prediction models, the analysis windows were aligned with the instants of glottal closure. This results in pitch-synchronous update of parameters and reduces the effect of filter memory at update.

3.8 Preprocessing of Speech and Laryngograph Data

3.8.1 Sources of Speech and Laryngograph Data

Simultaneous recorded speech and laryngograph data was obtained from the Eurom 0 database which was compiled as part of the ESPRIT project 1541 (SAM) (Grice & Barry 1989). The data consists of 52 minutes of speech for each of 5 languages (Danish, Dutch, English, French, Italian) for two male and two female speakers. Recordings were made by calibrated condenser microphones in anechoic or quiet office rooms using Sony Pulse Code Modulation (PCM) digital audio recorders in a 14-bit error correcting mode. Segments of speech from a male English speaker, with an average pitch period of 63ms, were used.

3.8.2 Initial Preprocessing

All data was down-sampled from 16kHz to 8kHz. Following the work of Milenkovic (1986), laryngograph signals were converted to a filtered signal, $L(t)$, using a zero phase, high-pass IIR filter, $P(z)$, with transfer function

$$P(z) = \frac{(1 - z^{-1})(1 - z)}{(1 - 0.98z^{-1})(1 - 0.98z)} \tag{3.53}$$

The effect of this filter is to remove slow variations in dc level and low frequency phase distortion which is introduced by the recording process. The instants of glottal closure of the vocal tract coincide with peaks in the derivative of the filtered signal, $dL(t)$, which was approximated by the first difference

$$dL(t) = L(t) - L(t-1) \tag{3.54}$$

The propagation delay between the glottis and the microphone must be removed to ensure that the speech and laryngograph signals are correctly aligned[7]. For voiced segments, it was assumed that the instants of glottal closure coincide with the minima of the speech waveform and peaks in the linear prediction residual waveform. These points were aligned with the peaks in $dL(t)$, as illustrated in Fig. 3.5, by the following procedure. Using a 125ms segment of voiced segment for which distinct peaks were observed in the residual signal obtained from a 10th order linear prediction analysis, the minima in the speech waveform were aligned with the peaks of $dL(t)$ by locating the minimum in the cross-correlation between the two signals, and delaying $dL(t)$ by the corresponding number of samples. This gives an average alignment for the two signals over the 125ms segment, which is typical of the whole utterance (assuming that the head to microphone distance remains constant). This alignment was compared with that between the peaks in

[7] Linear black-box models are less sensitive to advance misalignment of excitation, therefore it is better to determine a conservative estimate of the number of samples to align the laryngograph signal so that any resulting misalignment is an advance error (the peaks in dL(t) occur before the true instants of glottal closure of the speech utterance).

the residual and the peaks in $dL(t)$ and was found to differ by only 1 or 2 samples. Typical alignments of 5-7 samples were obtained, corresponding to a glottis-to-microphone distance of approximately 21.8-30.6cm. These are reasonable estimates for an average male vocal tract of length 17.5cm. The positive peaks in $dL(t)$ (which correspond to the instants of glottal closure for voiced segments) were located by simple thresholding, and will be referred to as *pitch marks.* The speech signal was normalised to the range ± 1 and the mean value (over the entire utterance) removed. Normalisation helps to minimise the possibility of ill-conditioning of the parameter equations for ARX models (Thomson 1992).

3.8.3 Pitch and Voicing Analysis

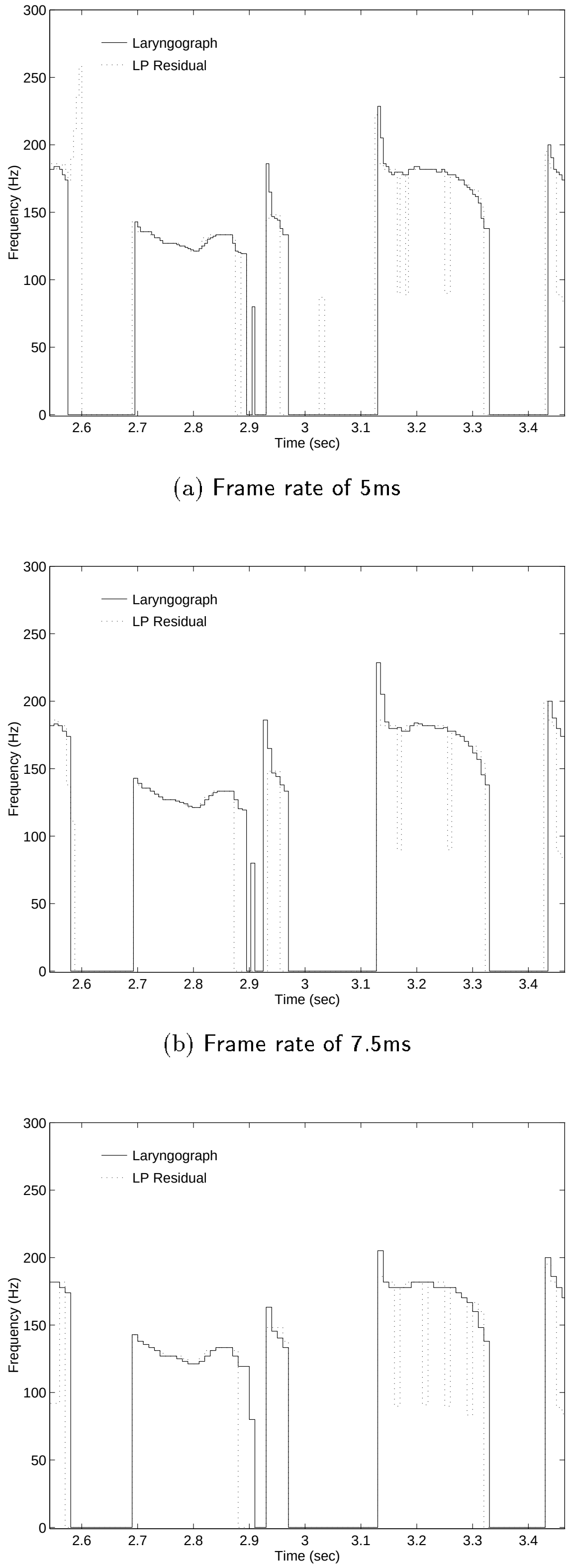
Accurate frame-by-frame pitch and voicing decisions are required for good quality synthesis of an utterance and can be obtained from the corresponding laryngograph signal. Each speech signal was divided into a sequence of overlapping pitch analysis windows, at a particular frame rate, with the centre of the window aligned with the centre of the current frame. An estimate of the local pitch period for each frame was calculated as the average spacing of pitch marks within the corresponding pitch analysis window. Pitch estimates were constrained to the range 80-380 Hz. Regions of unvoiced speech and silence were detected by an absence of pitch marks. The silence frames, which were omitted at the vocal tract analysis stage, were detected by applying an energy threshold to the speech signal in these regions.

To verify that laryngograph signals give accurate pitch estimates, the pitch contours were compared with those obtained by detecting peaks in the autocorrelation of the residual from linear prediction analysis. To obtain the contours, a 10th order linear prediction analysis was performed for each pitch analysis window and the pitch was estimated by the position of the first peak in the normalised autocorrelation of the residual signal, Eqn. (3.55), above a threshold of $0.28 \times R(0)$. $R(0)$ corresponds to the energy in the residual signal, $r(t)$, over a pitch analysis window of length N_p. The normalisation factor, $p(k)$, accounts for the decreasing number of samples used in the calculation of $R(k)$ as k increases. Frames with no peaks were marked as unvoiced and the pitch estimates were smoothed to remove isolated voiced/unvoiced decisions within long unvoiced/voiced segments.

$$R(k) = \frac{p(k)}{N_p} \sum_{t=1}^{N_p-k} r(t)r(t+k) \tag{3.55}$$

$$p(k) = \frac{1}{N_p - k + 1} \tag{3.56}$$

The two analysis procedures gave fairly smooth pitch contours and voicing decisions, as illustrated in Fig. 3.18 for pitch analyses of a short fragment "... itten with a small se ..." of the utterance "written with a small set of letters", at three different frame rates. The



(a) Frame rate of 5ms

(b) Frame rate of 7.5ms

(c) Frame rate of 10ms

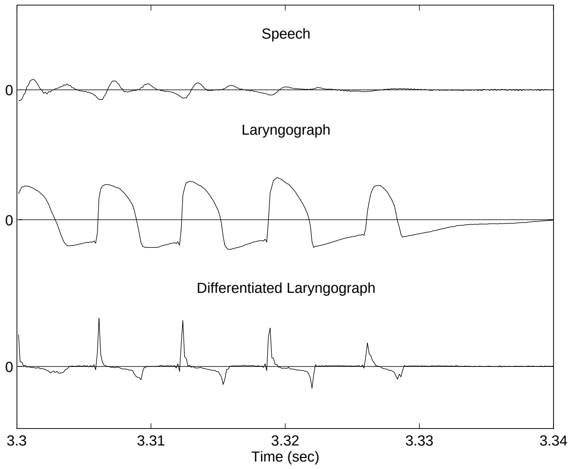
Figure 3.18: Comparison of pitch contours at different frame rates for the fragment 'itten with a small se' of the utterance 'written with a small set of letters'. Note the errors made by the LP algorithm at voicing transitions, for example at the voicing transition near 3.3ms.

speaker was male with an average pitch of 160Hz. At higher frame rates, the contours are smoother due to averaging of more pitch periods, but voicing transitions are not tracked as accurately. An example occurs for the 'i' to 'th' transition in 'with' at around 2.9ms. At a frame rate of 10ms, the isolated glottal pulse around 2.9ms is lumped in with the end of the last voiced segment. The main differences in the contours obtained from the laryngograph signal and the LP residual signal occur at voicing transitions, due to the averaging effect of the autocorrelation function over the pitch analysis window. Typical examples of voicing transitions where the analyses differ are shown in Fig. 3.19. They correspond to the following situations:

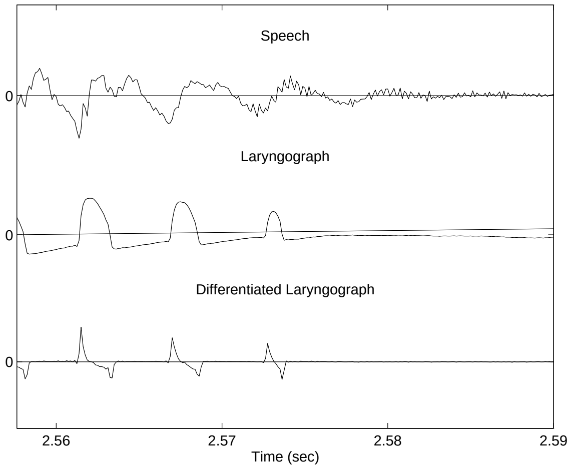
- voiced-unvoiced transitions
  - In some voiced-unvoiced transitions, vocal fold vibration can persist without generating significant acoustic energy and is due to the vocal folds slowly coming to a rest position (Krishnamurthy & Childers 1986). An example is the 'l' to 's' transition in 'small set' at around 3.3ms, Fig. 3.19(a). Due to the low energy of the speech signal, no peaks in the linear prediction residual occur and voiced frames at the end of the voiced segment are incorrectly marked as unvoiced by the residual analysis. The pitch of the voiced segment often decreases towards the end of a voiced segment (pitch declination), causing the pitch period to increase. Some of the voiced frames at the end of the voiced segment do not, therefore, contain a pitch mark.
  - In some voiced-unvoiced transitions, resonance of the vocal tract persists after the last glottal excitation pulse. The speech signal continues to show a periodic structure although vibration of the vocal folds has ceased. An example is the 'i' to 't' transition in 'written' at around 2.6ms, Fig. 3.19(b). Although the frames at the start of the unvoiced segment do not contain pitch marks, the speech signal has significant energy. Peaks occur in the linear prediction residual and unvoiced frames at the beginning of an unvoiced segment are incorrectly marked as voiced by the residual analysis.
- unvoiced-voiced transitions

  Due to the averaging effect of the autocorrelation function and a pitch analysis window which extends into the voiced segment the residual analysis may return a voiced decision even though there are no pitch marks until the onset of voicing. An example of this type of error occurs at the 's' to 'e' transition in 'set' at around 3.4ms, Fig. 3.19(c).

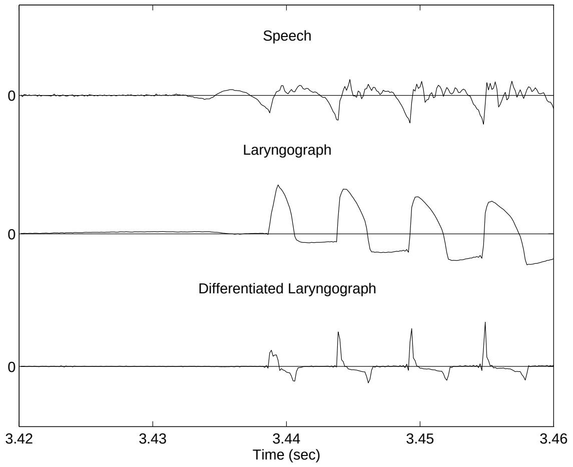
The main errors in the pitch and voicing analysis using the laryngograph occur due to artifacts in the signal $L(t)$ (which give spurious peaks in $dL(t)$), and very low amplitude peaks in $dL(t)$ which fall below the peak-picking threshold. Low amplitude peaks mainly occur in regions where the energy in the speech signal is low, for example at the end



(a) 'l' to 's' transition in 'small set'



(b) 'i' to 't' transition in 'written'



(c) 's' to 'e' transition in 'set'

Figure 3.19: Typical voicing transitions where errors in voicing decision occur.

of voicing, and omission of glottal volume velocity pulses in these regions did not cause perceptual degradation of the synthetic speech.

3.9 The Vocal Tract Filter

3.9.1 Model Order

Voiced speech sampled at 8kHz typically exhibits 4 formants which are modelled by 8 poles. Nasals require an additional complex-conjugate pole pair and a complex-conjugate zero pair, and lip radiation can be approximated by a zero at $z = 1$. To account for all these factors, a preliminary choice of model size was $n_a = 10$, $n_b = 4$ (10 poles, 3 zeros) for input $x(t)$ and $n_a = 10$, $n_b = 3$ for input $dx(t)$.

For linear prediction models of voiced speech, 2-3 poles are needed in addition to those required for modelling formants in order to account for the effects of glottal shaping and lip radiation. An additional 2 complex-conjugate pole pairs would also be needed to model a single nasal anti-resonance. Thus $n_a = 14$ is a suitable initial choice of model order.

For unvoiced speech, which was modelled by a linear prediction model, the spectrum may exhibit 1 or 2 bands of high frequency energy but has no clear formant structure. One pole is required to model lip radiation suggesting a model order as low as $n_a = 4 - 6$ is adequate. However, the number of poles was fixed to that used for voiced speech.

For LP and ARX models, the effect of varying the model order about these initial estimates is reported in sections 3.10.1 and 3.10.2 respectively.

3.9.2 Parameter Estimation

The parameters for OE models were estimated by the prediction-error method, which was solved using the Gauss-Newton procedure[8](Ljung 1987, pp. 282–284). In the case of ill-conditioning, a regularizing term was added to the approximation of the Hessian (the *Levenberg-Marquardt* procedure (Ljung 1987)). Initial estimates for model parameters were obtained from ARX models of the data. In order to minimise the effect of the filter memory from the previous frame, speech samples of the initial regression vectors of a frame which were outside the current analysis window were supplied by the synthesised values from the previous frame. For example, for an analysis window that starts at time t, the first regression vector in the current analysis window will be

$$\phi_{oe}(t) = [-\hat{y}_s(t-1) \ \ldots \ -\hat{y}_s(t-n_a)x(t) \ \ldots \ x(t-n_b+1)]^{\mathrm{T}}$$

where $[\hat{y}_s(t-1) \ \ldots \ \hat{y}_s(t-n_a)]$ are the last n_a samples of synthesised speech for the previous frame. For the analysis of long segments of speech, including these initial conditions at the analysis stage was important for obtaining a smooth synthesis.

[8] The vocal tract modelling framework described in this chapter was implemented in MATLAB$^{\mathrm{TM}}$. Modified versions of the oe.m, arx.m and lp.m algorithms, which are in the System Identification Toolbox (Ljung 1991), were used to estimate model parameters.

Parameters for ARX models were found by linear least-squares, which was solved by calculation of the Moore-Penrose pseudo-inverse. The regression vectors, $\phi_{arx}(t)$, Eqn. (3.11), were augmented as for OE models. This does not result in an analysis exactly analogous to the conventional covariance method of linear prediction analysis because synthesis samples, $[\hat{y}_s(t-1) \ldots \hat{y}_s(t-n_a)]$ were used to augment $\phi_{arx}(t)$, rather than original speech samples, $[y(t-1) \ldots y(t-n_a)]$.

Linear prediction models were also calculated by least-squares. Two variations of the parameter estimation were considered, the covariance method described above, and an autocorrelation method (Rabiner & Schafer 1978), in which values outside the current analysis window were set to zero and a Hamming window was applied to the data prior to analysis. The covariance and autocorrelation methods resulted in slightly different parameter estimates (see section 3.10.1).

The stability of the polynomials, $A(q)$ and $F(q)$, is not guaranteed and unstable poles were reflected inside the unit circle, to give stable models for synthesis.

3.9.3 Filter Implementation

The vocal tract filter, $H(q)$, was implemented in both cascade and parallel form. In the cascade form, the synthetic speech was calculated directly by filtering the excitation by $H(q)$. In the parallel form, $H(q)$ was factorized into second order filters by partial fraction expansion, analogous to Eqn. (2.13) for linear prediction models, and the synthetic speech was generated from the sum of the outputs of these filters.

3.10 Performance on Real Speech Data at Normal Pitch

Real speech data was used to evaluate the relative performance of linear prediction, ARX and OE models for synthesising speech. The performance was evaluated objectively using the prediction and synthesis SNR defined in Eqns. (3.28) and (3.29). The values reported in Tables 3.4 and 3.5 were averaged over 4.3 seconds of voiced speech and 2.1 seconds of unvoiced speech, which formed 3 utterances by a male speaker. The results of Table 3.8 were averaged over a slightly longer duration of 4.6 seconds. The synthesis SNR for linear prediction models has not been reported since these values were highly dependent on the alignment of the impulse excitation with respect to the original speech. For impulses placed at the instants of glottal closure (as illustrated in Fig. 3.17), the resulting linear prediction synthesis is nearly 180 degrees out-of-phase with the original speech and the synthesis SNR is therefore poor. Linear prediction performance was evaluated based on prediction SNR and the perceptual quality of the synthetic speech generated by these models. A subjective evaluation of the quality of syntheses from different linear models was based on informal listening tests. The tape demonstration in Appendix B contains examples of the synthetic speech for different models.

3.10.1 Linear Prediction Performance

The performance of linear prediction models was examined in order to select a suitable analysis method and model order for generation of good quality synthetic speech, for comparison with that generated by black-box models. The performance of models estimated by covariance and autocorrelation analysis was compared and the effect of varying the predictive order and using pre-emphasis was investigated. The effect of varying the predictive order on the prediction SNR, Eqn. (3.28), is shown in Tables 3.4(a) and 3.4(b), for the autocorrelation and covariance methods respectively. For both autocorrelation and covariance methods on voiced speech, the mean prediction SNR increases with model order, as expected. The high standard deviations, which remain fairly constant over all model orders, are a result of the highly variable nature of speech data in an utterance. There was a marked improvement in perceptual quality of synthetic speech when increasing model order from $n_a = 8$ to $n_a = 10$. There was a small perceptual difference in performance from $n_a = 12$ to $n_a = 16$ and a slight improvement in naturalness as the model order was increased. For unvoiced speech, Tables 3.4(a) and 3.4(b) show there is little difference in the prediction SNR from different predictive orders and the prediction SNR is much poorer than that for voiced speech. There was little difference in perceptual quality of the syntheses produced by different order models.

Comparison of Tables 3.4(a) and 3.4(b) shows improved prediction SNR for the covariance method of linear prediction analysis when compared to the autocorrelation method. A direct comparison of prediction SNR can be misleading however, since prediction residuals for the autocorrelation approach were calculated by filtering unwindowed speech, whereas model parameters were actually calculated to minimise the prediction error obtained using Hamming windowed speech. A better assessment of the relative performance of the two methods is obtained by comparing the perceptual quality of the resulting syntheses. Perceptually, utterances synthesised by covariance linear prediction sounded more buzzy than those synthesised by the autocorrelation method. As described in section 2.3.4, the effect of pre-emphasis is to enhance the higher frequencies prior to analysis and to alter the transfer function that is estimated. Perceptually, it was found that there was no audible difference between autocorrelation models estimated with or without pre-emphasis. For the covariance method, pre-emphasis removed the buzz in synthesis. With pre-emphasis, the methods of autocorrelation and covariance linear prediction were perceptually comparable, although the distortions for the two methods sounded different.

An example of the spectral match of the two methods to the original speech is shown in Fig. 3.20 for a 15ms segment of the phone ‘s’. The spectrum for the autocorrelation method without application of a Hamming window is also shown, to illustrate the improved spectral match to the original speech obtained by windowing the speech data with a Hamming window. The spectra estimated by the unwindowed autocorrelation and covariance methods are very similar, suggesting that for analysis windows of 15ms and above, the effect of initial conditions is not great.

MODEL ORDER	VOICED		UNVOICED	
n_a	mean	standard deviation	mean	standard deviation
8.00	9.73	3.98	5.82	3.60
10.00	9.92	3.92	6.03	3.66
12.00	9.93	3.92	6.13	3.66
13.00	9.95	3.95	6.16	3.67
14.00	9.97	3.94	6.20	3.68
16.00	10.00	3.95	6.24	3.68
19.00	10.04	3.97	6.31	3.68
25.00	10.06	3.99	6.38	3.67

(a) Autocorrelation linear prediction model

MODEL ORDER	VOICED		UNVOICED	
n_a	mean	standard deviation	mean	standard deviation
8.00	10.72	4.35	6.40	3.75
10.00	11.22	4.34	6.68	3.84
12.00	11.28	4.33	6.85	3.93
13.00	11.32	4.33	6.87	3.86
14.00	11.37	4.28	6.88	3.88
16.00	11.39	4.14	6.99	3.92
19.00	11.39	4.03	7.13	3.96
25.00	11.42	4.03	7.22	3.90

(b) Covariance linear prediction model

Table 3.4: Effect of variation of model order on mean prediction SNR for linear prediction models without pre-emphasis.

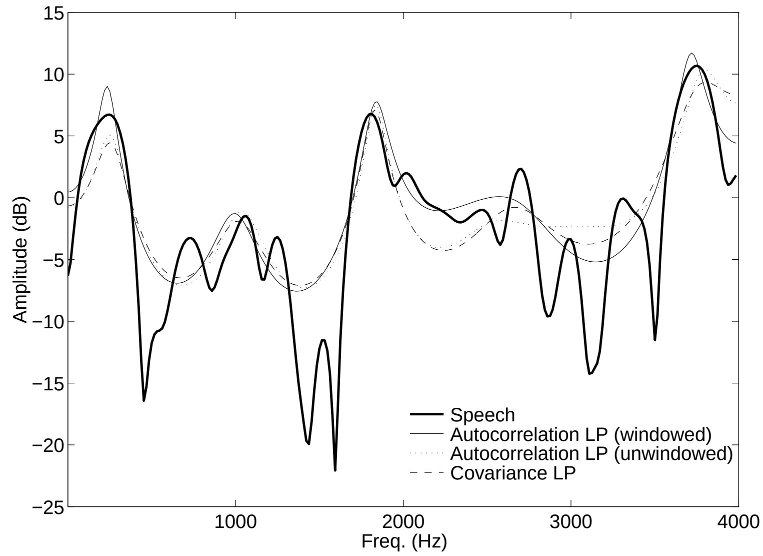


Figure 3.20: Comparison of spectra from autocorrelation and covariance methods of linear prediction ($n_a = 13$).

Model Order		Prediction SNR (dB)		Synthesis SNR (dB)	
n_a	n_b	mean	standard deviation	mean	standard deviation
8.00	2.00	14.38	5.84	5.62	5.03
8.00	3.00	14.90	5.90	6.17	5.22
8.00	4.00	15.24	5.83	6.30	5.35
10.00	2.00	14.94	5.74	5.96	5.11
10.00	3.00	15.60	5.81	6.61	5.31
10.00	4.00	16.09	5.65	6.49	5.56
10.00	9.00	16.77	5.70	6.95	5.75

Table 3.5: Effect of variation of model order on mean SNR (dB) for ARX models (input $x(t)$, no pre-emphasis).

3.10.2 ARX Performance

The effect of varying the model order, of using pre-emphasis and of varying the excitation ($x(t)$ or $dx(t)$) on the performance of ARX models was investigated. The performance of these models, in terms of prediction SNR and quality of synthetic speech produced, was compared with linear prediction models.

Effect of Variation of Model Order

The effect of varying the model order on the prediction SNR and synthesis SNR from ARX models is shown in Table 3.5, for models using input $x(t)$ and no pre-emphasis. The values of prediction SNR can be compared to those in Table 3.4(b) for covariance linear prediction analysis.

The results show an increase in mean prediction and synthesis SNR with increasing model complexity. For variation of the number of poles, n_a, there was a marked improvement in the synthesis from $n_a = 8$ to $n_a = 10$. Models with the same total number of parameters, but different number of poles, n_a, show similar performance. Perceptually, the syntheses from models with a higher number of poles were slightly better.

For the case $n_b = 1$ (not illustrated), the numerator simply performs a rescaling of the input signal and the resulting syntheses sounded muffled. Filtering these syntheses by the lip radiation filter, Eqn. (2.14), produced clear synthetic speech, showing that at least one zero is needed in the transfer function to model lip radiation ($n_b \geq 2$). For $n_b = 2$, only lip radiation can be accounted for by the transfer function. In this case, the coefficient of the zero, b_1, was found to be close or equal to -1 and the effect of $B(q)$ was primarily to calculate the first difference of the excitation. Hedelin (1984) also found this to be the case and fixed b_1 to -1 in synthesis without loss of performance. There was a marked improvement in naturalness and perceptual quality of the syntheses when n_b was increased from 2 to 4, which corresponds to the addition of one and two zeros in the transfer function, in addition to the zero required for lip radiation. This improvement by introducing zeros was also reported by Fujisaki & Ljungqvist (1987) for synthetic and natural nasalised vowel data.

Although further increases in n_b gave slight improvement in SNR and perceptual quality of synthetic speech, there is benefit to be derived from using the minimum number of zeros which give good perceptual quality synthetic speech, because over-parametrization by zeros sometimes resulted in estimates of the transfer function which exhibited unnatural spectral dips. In addition, the parameter equations can become ill-conditioned as n_b increases (Thomson 1992). This gives inaccurate estimates of model parameters and large variations in their values from frame to frame. The effect of the filter memory for $A(q)$ then becomes significant, causing clicks in the synthesis when the parameters are updated. For voiced frames, the memory for $B(q)$ is effectively zero because parameters are updated at the instants of glottal opening where the input has been zero for several samples.

The final selection of model order was the smallest model order which gave good perceptual performance, and is summarised in Table 3.6. For voiced speech modelled by an ARX model with an input $x(t)$, a model order of $n_a = 10, n_b = 3$ (2 zeros) was found to perform well on average and perceptually was as good as a model order $n_b = 4$, as initially suggested. With input $dx(t)$, the numerator was reduced to $n_b = 2$. For the linear prediction of unvoiced frames used in the black-box framework, $n_a = 10$ was selected so that the linear prediction model had the same number of poles as the ARX model. Since there is no formant structure to unvoiced frames, this is more than adequate and perceptually gave no distinguishable difference from the performance of a higher order model on unvoiced frames. For the linear prediction framework, the model order, $n_a = 13$ ($n_a = 12$ with pre-emphasis), was used for both voiced and unvoiced frames. This allows for direct comparison between linear prediction and black-box models with the same total number of parameters.

Modelling Framework	**VOICED SPEECH**	**UNVOICED SPEECH**
Black-box input $x(t)$	ARX or OE, $n_a = 10$, $n_b = 3$ with or without pre	LP, $n_a = 10$
Black-box input $dx(t)$	ARX or OE, $n_a = 10$, $n_b = 2$ with or without pre	LP, $n_a = 10$
Linear Prediction	LP, $n_a = 13$ without pre	LP, $n_a = 13$
Linear Prediction	LP, $n_a = 12$ with pre	LP, $n_a = 12$

Table 3.6: Summary of model orders used for vocal tract modelling using black-box and linear prediction models ('pre' indicates pre-emphasis).

Effect of Variation of Model Input

As described in section 3.4.2, using input $dx(t)$ results in estimates of the vocal tract transfer function from which the lip radiation component has been removed. The model order n_b was reduced to $n_b = 2$ for input $dx(t)$ because a zero is no longer needed for lip radiation. Models using input $dx(t)$ allow fair comparison with pre-emphasised linear prediction models, because the transfer function of the inverse LP filter also represents that of the vocal tract only. The final column of Table 3.8(a) shows a comparison of mean synthesis SNR for ARX models using input $x(t)$ and $dx(t)$. These figures show similar performance in synthesis. Perceptually, there was no difference in the quality of synthetic speech produced by models estimated using $x(t)$ or $dx(t)$ as input.

Effect of Pre-emphasis

On synthetic data, pre-emphasis of data was shown to give more accurate estimation of higher formants at the expense of estimates of lower formants. In practice, using pre-emphasis had little or no effect on the perceptual quality of the syntheses produced by ARX models. Table 3.8(a) shows the mean synthesis SNR for models with various combinations of input and pre-emphasis.

The synthesis SNR with pre-emphasis is much lower than the corresponding results without pre-emphasis. This is due to the fact that the synthesis SNR was calculated on the pre-emphasised syntheses and is therefore affected by the scaling of the pre-emphasis filter.

Comparison of Linear Prediction and ARX Models

For fair comparison of linear prediction and ARX models, the covariance method of linear prediction was used, because the analysis procedures for ARX and LP models are then equivalent. The improvement in prediction SNR of ARX models over linear prediction models with the same number of poles ($n_a = 10$) and the same total number of parameters ($n_a = 13$) is shown in Table 3.7.

Models Compared	**Improvement**
ARX ($n_a = 10$, $n_b = 3$, input $x(t)$) LP ($n_a = 10$)	4.3dB±1.2dB
ARX ($n_a = 10$, $n_b = 3$, input $x(t)$) LP ($n_a = 13$)	4.2dB±1.1dB

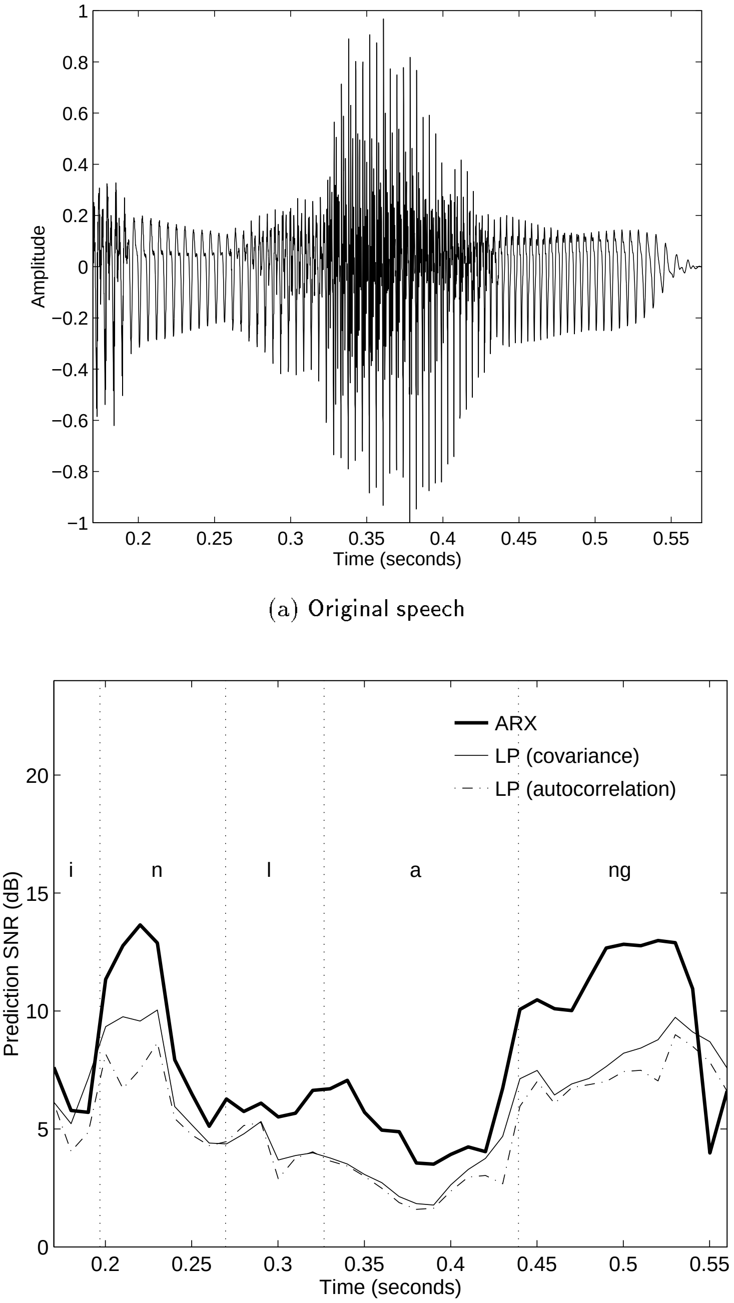
Table 3.7: Improvement in prediction SNR of ARX models over LP models. These improvements were based on the values in Table 3.4(b) and Table 3.5

These results compare favourably with the improvements of $2 - 7dB$ reported by Hedelin (1984), Fujisaki & Ljungqvist (1986) and Thomson (1992). The syntheses by the LP and ARX models were compared using informal listening tests. Perceptually, synthetic speech produced by ARX models was much clearer and sounded more natural than that from LP models. This improvement in perceptual quality may be verified by listening to the tape demonstration provided.

In Fig. 3.21(a), comparison of the performance of ARX and LP models is given for the speech segment 'in lang' taken from the utterance 'in language', spoken by a male speaker. The results were obtained for ARX models of order $n_a = 10, n_b = 2$ (input $dx(t)$) and LP models of order $n_a = 12$. The analysis was performed using 25ms analysis windows at a frame rate of 10ms and data was pre-emphasised by $P(q) = 1 - 0.95q^{-1}$.

As illustrated in Fig. 3.21(b), which shows a frame-by-frame comparison of prediction SNR for ARX and LP models, ARX analysis consistently improves the prediction SNR of voiced phones. In particular, there is a large improvement for nasals, for example for 'n' and 'ng'. The greater improvement in prediction SNR for nasals is derived from the inclusion of zeros in the vocal tract model. In general, the improved prediction SNR of ARX models is due to improved estimates of the vocal tract transfer function, especially in the region of the first formant. Fig. 3.22(a) illustrates the transfer functions estimated for a fragment of the phoneme 'ng' and compares them to the ETFE of the vocal tract system. It is important to bear in mind that the ETFE may not represent a good estimate of the underlying system since it is dependent on $dx(t)$, which is only a model of the true excitation of the vocal tract. Figs. 3.22(c) and 3.22(d) show the matches between the spectra of the synthesis errors and the spectra of the model noise.

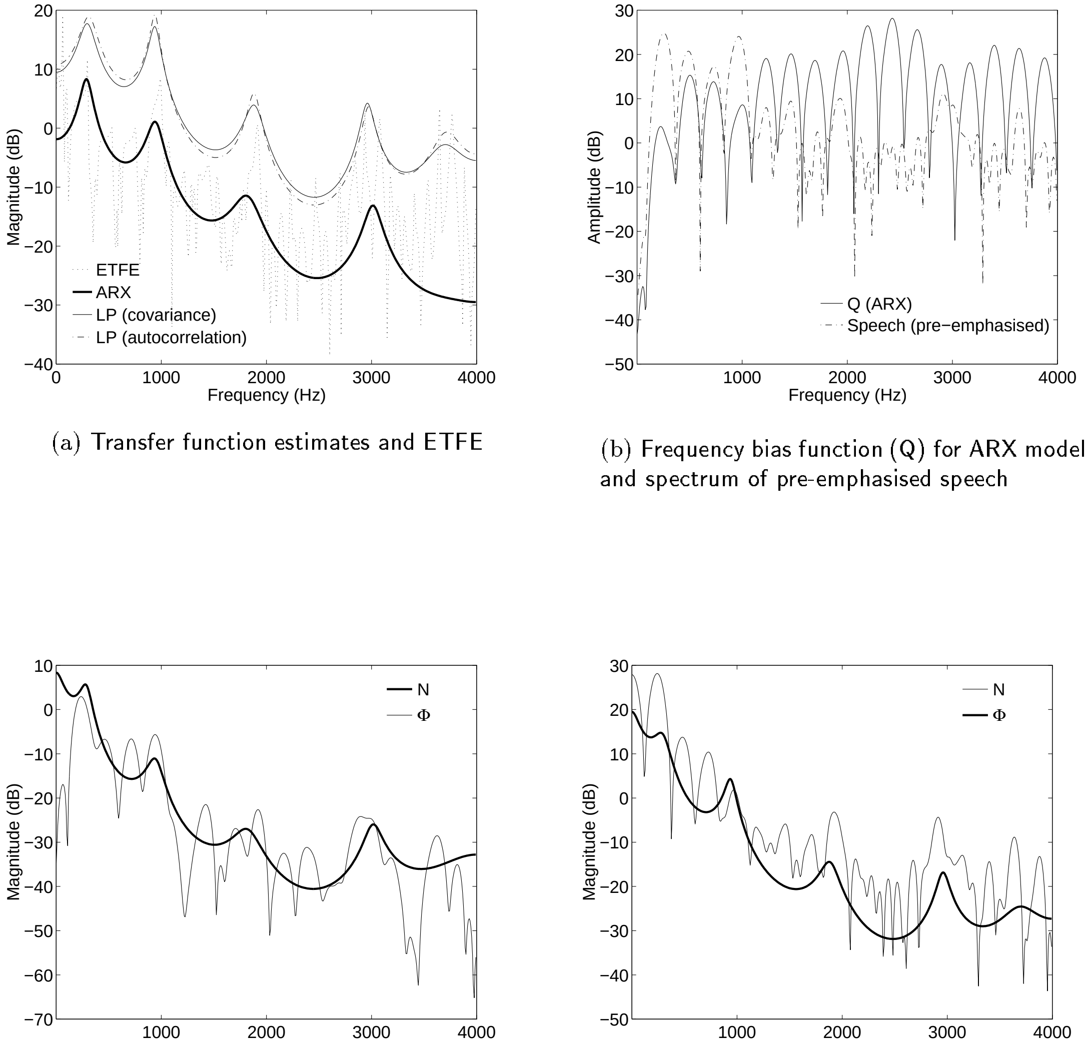
The improved performance of the ARX model, especially in the estimation of the first formant can be illustrated by examination of spectrograms of the resulting syntheses. An example of spectrograms of synthetic speech from LP and ARX models for the utterance "Germany's decision followed eight years later", are shown in Fig. 3.23. Comparison with the spectrogram of the original utterance, Fig. 3.23(a), shows accurate estimation of the formants and improved estimation of the first formant by ARX models.



(a) Original speech

(b) Prediction SNR for ARX and LP models

Figure 3.21: Comparison of ARX and LP models for the speech fragment 'in lang' taken from the utterance 'in language'. ARX models used input $dx(t)$. Pre-emphasis was used for both ARX and LP models.



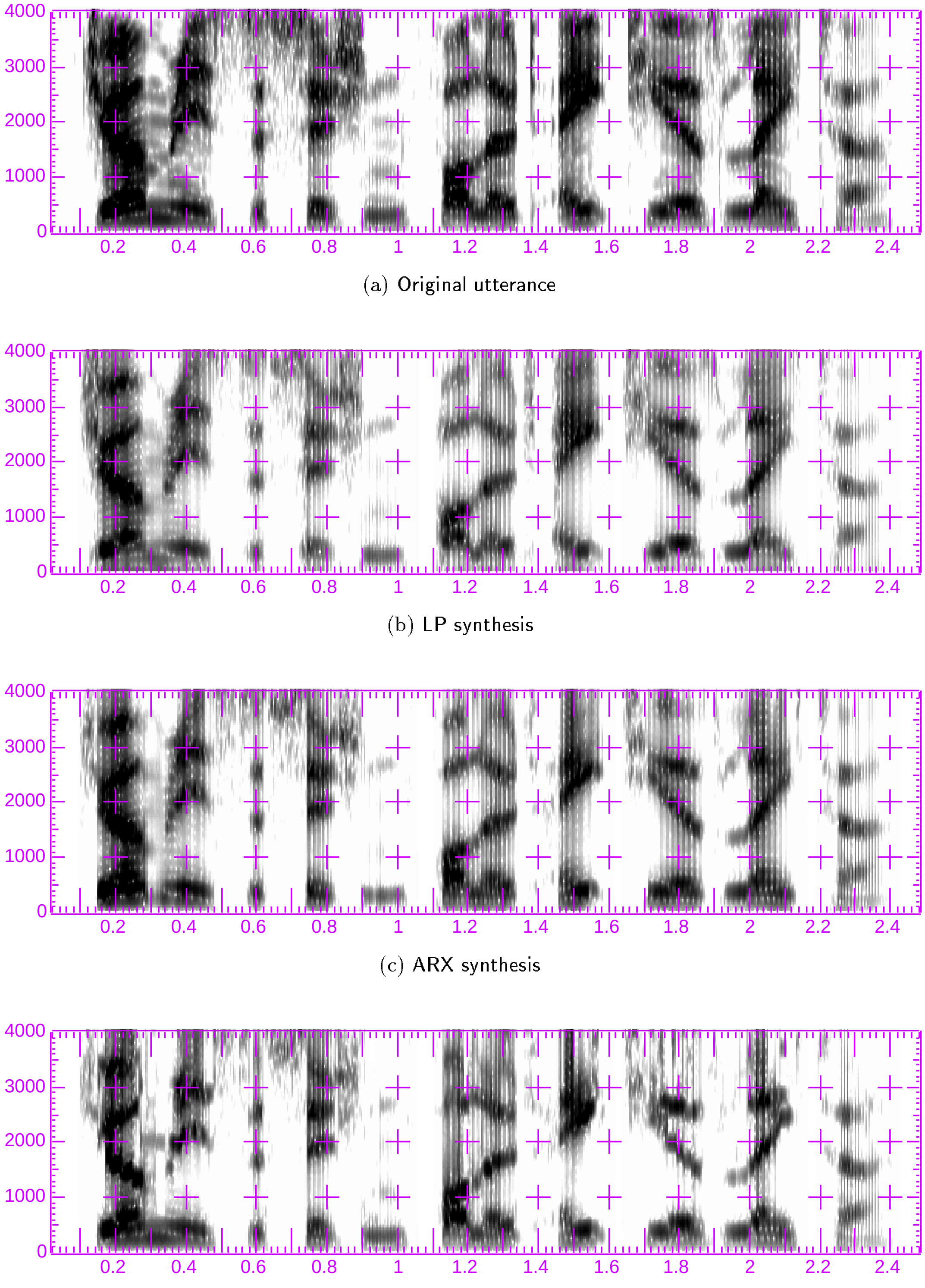
(a) Transfer function estimates and ETFE

(b) Frequency bias function (Q) for ARX model and spectrum of pre-emphasised speech

(c) Model noise (N) and synthesis error spectrum (Φ) for ARX model

(d) Model noise (N) and synthesis error spectrum (Φ) for LP model

Figure 3.22: Comparison of ARX and LP models for a segment of the phone 'ng' (0.45-0.46 seconds) taken from the utterance shown in Fig. 3.21(a).



(a) Original utterance

(b) LP synthesis

(c) ARX synthesis

(d) OE synthesis (no pre-emphasis)

Figure 3.23: Spectrograms of synthesis of the utterance 'Germany's decision followed eight years later' by different vocal tract models. Horizontal axis shows time in seconds, vertical axis shows frequency in Hz.

Model Order		OE(2)		OE(11)		ARX SNR		Comment
n_a	n_b	mean	std	mean	std	mean	std	
10.00	2.00	7.87	5.60	9.26	5.76	6.66	5.18	$dx(t)$,no pre
10.00	3.00	7.97	5.63	9.36	5.70	6.86	5.18	$x(t)$,no pre
10.00	2.00	4.57	4.71	5.24	4.72	3.32	4.22	$dx(t)$, pre
10.00	3.00	4.88	4.98	5.50	4.87	3.72	4.31	$x(t)$, pre
10.00	2.00	4.28	4.15	4.49	4.35	3.23	3.87	$dx(t)$, pre and constraints
10.00	3.00	4.59	4.33	4.74	4.40	3.67	3.78	$x(t)$, pre and constraints

(a) Mean synthesis SNR (dB)

Model Order		OE(2)		OE(11)		Comment
n_a	n_b	mean	std	mean	std	
10.00	2.00	1.36	1.83	2.60	2.61	$dx(t)$,no pre
10.00	3.00	1.29	1.64	2.50	2.57	$x(t)$,no pre
10.00	2.00	1.32	1.86	1.92	2.44	$dx(t)$,pre
10.00	3.00	1.21	1.78	1.78	2.24	$x(t)$,pre
10.00	2.00	1.08	1.51	1.26	1.74	$dx(t)$,pre and constraints
10.00	3.00	0.94	1.47	1.06	1.75	$x(t)$,pre and constraints

(b) Mean improvement in synthesis SNR (dB)

Table 3.8: Effect of variation of input $(x(t), dx(t))$ and use of pre-emphasis (pre) on OE and ARX models. OE(2)- 2 iteration, OE(11)- 11 iterations.

3.10.3 Comparison of OE and ARX Models

Using the model orders given in Table 3.6, the effects of varying model input and using pre-emphasis on the performance of OE models were investigated and the resulting syntheses compared with those from ARX models. The mean synthesis SNR for the different models is shown in Table 3.8(a) and the mean improvement in SNR over ARX models is shown in Table 3.8(b). The ARX models were calculated simultaneously with the OE models, using the same filter initial conditions at the start of each frame so that a fair comparison of SNR on a frame-by-frame basis could be made.

Table 3.8(a) shows a small difference in mean synthesis SNR between models using input $x(t)$ and $dx(t)$. Perceptually, there was no noticeable difference in the quality of synthetic speech from models using either input. With pre-emphasis, the reported mean synthesis SNR is consistently less than that without pre-emphasis because the values were calculated on pre-emphasised data. By comparing the synthesis SNR for OE and ARX models, it is seen that the OE models consistently improve the synthesis SNR above that of the ARX models. This is illustrated in Fig. 3.24(a), for the specific example utterance 'in lang' used previously.

Table 3.8(a) shows that the SNR is improved even after two iterations. Limiting the number of iterations of the OE algorithm means that for most frames the algorithm will not have converged to a local or global minimum. This is shown by the increased

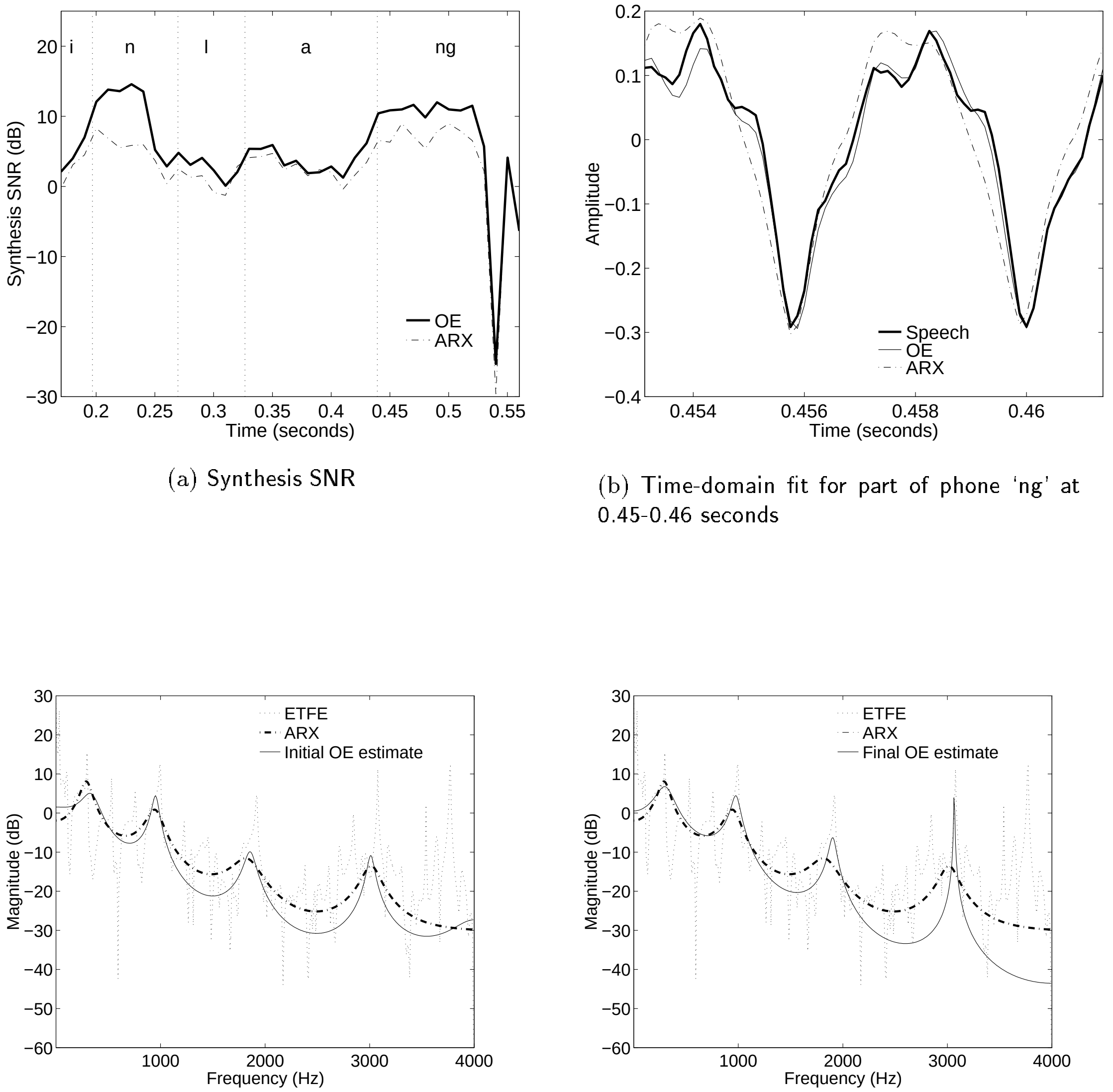
SNR after further iterations. Convergence typically occurred in 6-10 iterations. The improvement in synthesis SNR results in an improved fit between the original speech and the synthetic speech in the time-domain. This is illustrated in Fig. 3.24(b) for the fragment 'ng', taken from the utterance illustrated in Fig. 3.21(a). Table 3.8(b) shows that the mean improvement in synthesis SNR by OE models using a fixed number of iterations is typically $1 - 2dB$.

Despite higher synthesis SNR, syntheses from OE models were of poorer quality than those produced from ARX models and often exhibited distortion which sounded like running water or breaking glass. A similar 'waterfall' effect was reported by Wigren, Bergström, Harrysson, Jansson & Nilsson (1995) in work on the encoding of background noise in CELP coders and was described as background 'musical tones' by Lim & Oppenheim (1978) in their work on the estimation of linear prediction parameters for noisy speech.

The distortion is caused by unnatural temporal variation in the short-term spectrum which is due to the convergence of the OE algorithm to local minima with poor spectral structure for speech. Although the time-domain fit is improved, the OE algorithm causes movement of the formants from the positions given by the original ARX analysis and leads to large spectral changes from frame to frame. As with synthetic data, it was found that converging solutions after an increased number of iterations often resulted in formants with narrow bandwidths. An example of this behaviour is shown by comparison of Figs. 3.24(c) and 3.24(d). The formant movement and narrowing of bandwidths is particularly noticeable at higher frequencies due to the reduced frequency weighting of the transfer function estimate in the higher frequency region caused by the spectral roll-off of the excitation. This results in a much less constrained fit between the ETFE and model transfer function at higher frequencies. The temporal variation of the short-term spectrum is illustrated in Fig. 3.23(d), which shows the spectrogram for the synthesis of the utterance "Germany's decision followed eight years later" by OE models (no pre-emphasis). There is more variation in the formant tracks than in those for the corresponding spectrogram for ARX synthesis, Fig. 3.23(c), for example around 0.2 and 1.2 seconds, and narrower bandwidth formants are observed around 1.8 and 2 seconds.

3.10.4 Improving the Performance of the OE model

Although OE models consistently improved on the synthesis SNR from ARX models, the resulting syntheses from these models were perceptually poorer in quality. This was primarily due to the convergence of the parameter estimation procedure to local minima, which led to severe distortion of synthetic speech. In previous work in which this type of distortion was reported, several solutions were proposed. Lim & Oppenheim (1978) observed that with iterative parameter estimation techniques, narrow bandwidth solutions were generated at convergence after an increased number of iterations. They proposed limiting the number of iterations of the estimation procedure and applying some prior knowledge about the likely values of the parameters, in the form of the estimate from the



(a) Synthesis SNR

(b) Time-domain fit for part of phone 'ng' at 0.45-0.46 seconds

(c) Transfer function estimate after 1 iteration

(d) Transfer function estimate after 6 iterations (converged)

Figure 3.24: Comparison performance of OE and ARX models in synthesis (input $dx(t)$, with pre-emphasis).

previous frame. Hansen & Clements (1994) proposed a number of techniques to overcome the limitations of the approach of Lim & Oppenheim (1978), which were based on constraining the variability of the short-term spectrum from frame-to-frame. They proposed both intra-frame (iteration-to-iteration) and inter-frame (frame-to-frame) constraints and found that inter-frame constraints were most effective in reducing the distortion. Intra-frame constraints consisted of ensuring a stable synthesis filter, limiting the bandwidths of the formants and limiting the number of iterations of the parameter estimation procedure.

Although the above studies used equation error minimisation to estimate the parameters, the proposed constraints on the short-term spectrum are applicable to the OE estimation procedure. Of particular relevance is the work of Hansen & Clements (1994), because the parameter estimation procedure was iterative and therefore converged to local minima with poor spectral structure. The following constraints were applied to the OE estimation procedure in an attempt to limit the spectral variability from frame to frame. These constraints were applied in addition to a stability check on the resulting filter.

- limited number of iterations of Gauss-Newton algorithm
- bandwidth constraints
- regularization

Limited number of iterations

The Gauss-Newton procedure was initialised to the ARX model parameters and a limited number of iterations performed in an attempt to limit the movement of the parameters from the initial ARX values and hence limit the possible movement of the formants. There was still some perceivable 'waterfall' distortion after only two iterations, suggesting that the size of the update step results in significant movement of the parameters. In general, the resulting parameters correspond to a solution which has not converged and therefore requires that the initial ARX estimate is good. Limiting the number of iterations can be interpreted as a 'relaxation' of the parameter estimation procedure.

Bandwidth constraints

Constraints were placed on the permissible bandwidths of formants by imposing maximum limits on the radius of the poles of $F(q)$. Two bandwidth limits were imposed to account for a wider average formant bandwidth at higher frequencies. Based on average formant bandwidths for vowel sounds quoted in Deller et al. (1993) suitable minimum bandwidths were estimated as 40Hz for formants below 2kHz and 100Hz for formants above 2kHz. At 8kHz sampling rate, these correspond to a limit on the pole radius of 0.98 and 0.96 respectively. In most cases, the bandwidth limit for lower frequencies was not necessary.

Regularization

The application of bandwidth constraints and limited number of iterations had little effect in removing the 'waterfall' distortion. This suggests that the main cause of the distortion is the movement of the positions of the higher formants. The position of lower frequency formants does not change significantly with increased iterations because the bias function, $Q(\omega, \boldsymbol{\theta})$, enforces a better estimate of the transfer function at lower frequencies. Pre-emphasis was found to be important in reducing the movement of higher frequency formants as it enhances the higher frequencies of the frequency bias function and demands a closer fit between the transfer function estimate and the ETFE at higher frequencies. Pre-emphasis eliminated the distortion in some cases. In order to further constrain the movement of formants, a regularizing term was added to the error criterion to limit the movement of the poles from the initial ARX values, $\boldsymbol{\theta}^0$. The error criterion, $V(\boldsymbol{\theta})$, was augmented by a regularizing term to give

$$\tilde{V}_N(\boldsymbol{\theta}) = \frac{1}{2}\sum_{t=0}^{N-1} e_s^2(t) + \frac{\alpha}{2}\sum_{k=1}^{n_\theta}\left(\theta_k - \theta_k^0\right)^2 \tag{3.57}$$

where $n_\theta = n_f + n_b$. Defining

$$\boldsymbol{\psi}(t) = \left[\frac{\partial e_s(t)}{\partial \theta_1} \cdots \frac{\partial e_s(t)}{\partial \theta_{n_\theta}}\right]^{\mathrm{T}} \tag{3.58}$$

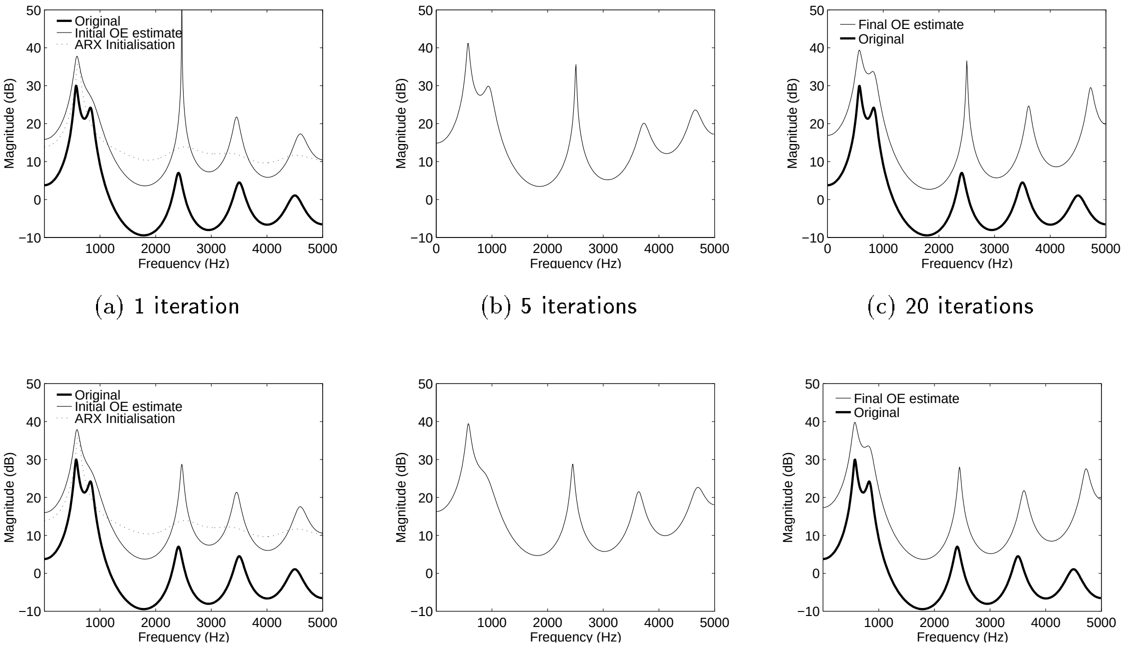
the corresponding gradients, $\tilde{V}'(\theta_k)$ and approximation to the Hessian, $\tilde{V}''(\theta_{jk})$ become

$$\begin{aligned} \tilde{V}'(\theta_k) &= \sum_{t=0}^{N-1} \psi_k(t) e_s(t) + \alpha(\theta_k - \theta_k^0) && (3.59) \\ \tilde{V}''(\theta_{jk}) &\approx \sum_{t=0}^{N-1} \psi_j(t)\psi_k(t) + \alpha\delta_{jk} && (3.60) \end{aligned}$$

where α is a constant which controls the amount of movement of the parameters from their initial values and $\delta_{jk} = 1$ for $j = k$ and $\delta_{jk} = 0$ otherwise. This can be expressed in matrix notation as

$$\begin{aligned} \boldsymbol{\Psi}_N &= [\psi(1) \ldots \psi(N)]^{\mathrm{T}} && (3.61) \\ \boldsymbol{E_N} &= [e_s(1) \ldots e_s(N)]^{\mathrm{T}} && (3.62) \\ \tilde{V}(\boldsymbol{\theta}) &= \frac{1}{2}\boldsymbol{E}_N^{\mathrm{T}}\boldsymbol{E}_N + \frac{1}{2}\alpha(\boldsymbol{\theta} - \boldsymbol{\theta}^0)^{\mathrm{T}}(\boldsymbol{\theta} - \boldsymbol{\theta}^0) && (3.63) \\ \tilde{\boldsymbol{V}}'(\boldsymbol{\theta}) &= \boldsymbol{\Psi}_N^{\mathrm{T}}\boldsymbol{E}_N + \alpha(\boldsymbol{\theta} - \boldsymbol{\theta}^0) && (3.64) \\ \tilde{\boldsymbol{V}}''(\boldsymbol{\theta}) &\approx \boldsymbol{\Psi}_N^{\mathrm{T}}\boldsymbol{\Psi}_N + \alpha\boldsymbol{I} && (3.65) \end{aligned}$$

where $\boldsymbol{I}$ is the $(n_\theta \times n_\theta)$ identity matrix. The resulting parameter update, with step size μ, is given by



(a) 1 iteration

(b) 5 iterations

(c) 20 iterations

(d) 1 iteration

(e) 5 iterations

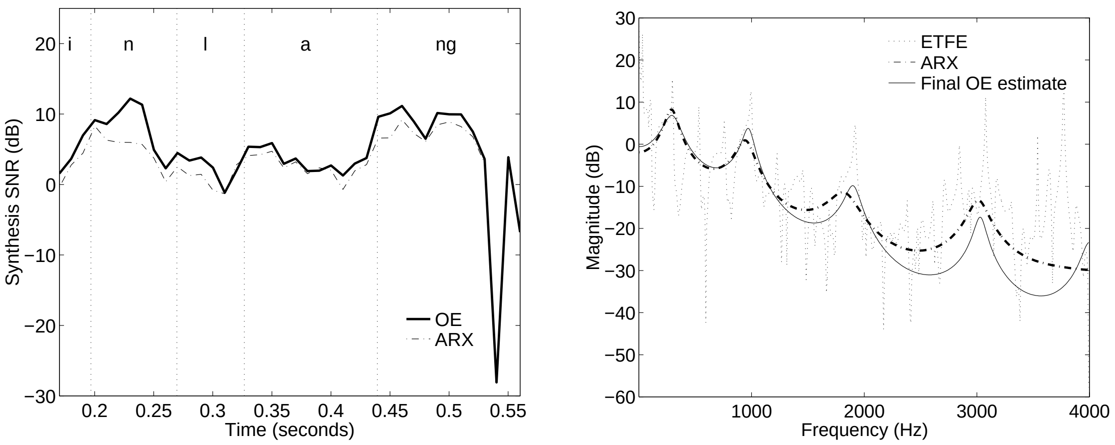
(f) 20 iterations

Figure 3.25: Transfer function estimates for the synthetic vowel in ' hawed' for models (input $dx(t)$ at 5dB SNR). Figs. 3.25(a), 3.25(b) and 3.25(c) are with pre-emphasis only. Figs. 3.25(d), 3.25(e) and 3.25(f) are with pre-emphasis, bandwidth constraints (40Hz and 100Hz) and regularization. Note the improved estimate of the vocal tract transfer function and more natural bandwidths obtained by using regularization and imposing bandwidth constraints.

$$\hat{\boldsymbol{\theta}}^{i+1} = \hat{\boldsymbol{\theta}}^{i} - \mu \left[\tilde{\boldsymbol{V}}''(\hat{\boldsymbol{\theta}}^{i})\right]^{-1} \tilde{\boldsymbol{V}}'(\hat{\boldsymbol{\theta}}^{i}) \tag{3.66}$$

For the synthetic vowel in ‘hawed’ at 5dB SNR, the effect of regularization on the resulting transfer function estimate is illustrated in Fig. 3.25. Figs. 3.25(a) - 3.25(c) show the transfer function estimates obtained with pre-emphasis, but no bandwidth constraints or regularization. Comparison with Fig. 3.13 shows that pre-emphasis of the data results in formants with more natural bandwidths. The effect of also applying regularization and bandwidth constraints is shown in Figs. 3.25(d) - 3.25(f). The effect of the bandwidth constraints is clearly evident in the estimate of the third formant in Fig. 3.25(f).

For natural speech data, applying constraints to the transfer function estimate reduced the ability of the OE algorithm to improve on the synthesis SNR from ARX models. This is illustrated by the reduced mean synthesis SNR of Table 3.8(a) and a lower mean improvement in SNR in Table 3.8(b). It was found that a fairly tight constraint on parameter movement was needed to prevent the ‘waterfall’ distortion. The resulting syntheses from regularized models were perceptually equivalent to those from ARX models, but gave a



(a) Synthesis SNR

(b) Transfer function estimate after 6 iterations (input $dx(t)$)

Figure 3.26: Comparison of performance of ARX and regularized OE models in estimation of the transfer function for phone 'ng' in 'in lang' (0.45-0.46 seconds). Pre-emphasis was used for ARX and OE models, For OE models, regularization and bandwidth constraints were also used.

higher synthesis SNR. For the example fragment 'in lang', the reduced improvement in SNR is seen when comparing Fig. 3.26(a) with the corresponding results for unregularized, unconstrained OE analysis shown in Fig. 3.24(a). The transfer function estimate obtained by a regularized OE model for the fragment of phone 'ng' is shown in Fig. 3.26(b). In comparison with the corresponding transfer function estimate by the unregularized OE model, Fig. 3.24(d), it is seen that the third and fourth formants have much more reasonable bandwidths.

The ability of the constraints to smooth the temporal variation in the short-term spectrum is seen in Fig. 3.27, which shows the spectrogram of the synthesis of the utterance "Germany's decision followed eight years later" produced by regularized OE models. In comparison to the spectrogram of the synthesis produced from unregularized OE models, Fig. 3.23(d), much smoother formant tracks are observed, for example around 1.2 seconds.

3.10.5 Effect of Misalignment Errors on Estimation of Vocal Tract Transfer Function

Errors in the estimation of the vocal tract parameters can arise when the speech and laryngograph signal are incorrectly aligned. The time delay between the laryngograph and speech signals due to the propagation time along the vocal tract was assumed constant for the duration of the utterance. This assumption is fairly accurate for the short duration utterances (typically 4.5 seconds) used in testing. It was found that errors of a couple of samples in the alignment resulted in a swirling distortion of the synthesised speech due

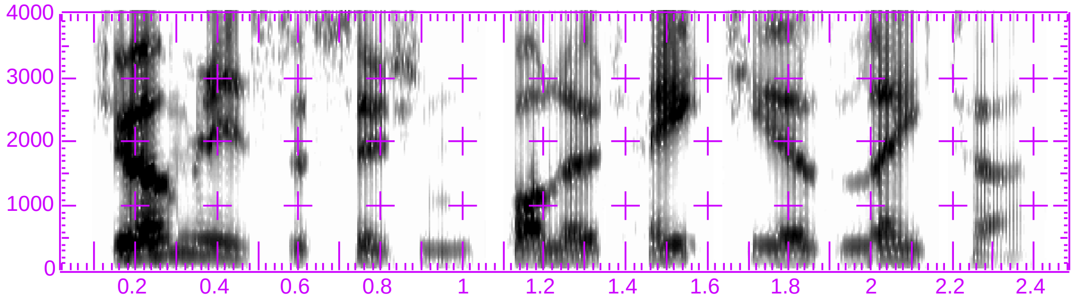


Figure 3.27: Spectrogram of synthesis of the utterance 'Germany's decision followed eight years later' by OE model with regularization, bandwidth constraints and pre-emphasis. Horizontal axis shows time in seconds, vertical axis shows frequency in Hz.

to inaccurate estimation of the vocal tract filter parameters. Alignment errors in which the laryngograph signal was delayed with respect to the speech signal where found to be more detrimental to accurate parameter estimation than the case where the laryngograph signal was advanced with respect to the speech signal. This sensitivity to alignment was also noted by Cheng & O'Shaughnessy (1989).

3.10.6 Parallel Implementation

Parallel implementations of black-box models generated synthetic speech that was audibly more noisy than direct implementations of $H(q)$. This was due to an increased sensitivity to the initial conditions at the onset of each frame.

3.11 Pitch Manipulation of Synthesised Speech

To manipulate the pitch of the synthetic speech generated from linear models, a new impulse sequence (LP) or glottal volume velocity wave (black-box models) was generated at the modified pitch period for voiced frames. For unvoiced frames, the codebook entry that was selected for synthesis at normal pitch was used. For LP synthesis, the scale factor, G, was recalculated to account for the change in energy of the excitation sequence.

To evaluate the pitch manipulation performance of vocal tract models, the pitch-synchronous overlap-add (PSOLA) method was used as a base-line for comparison (Hamon et al. 1989, Charpentier & Stella 1986). PSOLA is an existing method for producing high quality prosodic modifications of original speech waveforms and has been used successfully in text-to-speech systems to manipulate the pitch and timing of concatenated speech segments. It gives better quality synthetic speech than that obtained by direct manipulation of the impulse excitation of LP or multi-pulse models. The PSOLA method is outlined briefly in the following section and then the performance of different models will be described.

3.11.1 Pitch Manipulation by PSOLA

The implementation of PSOLA used in this work was based on the time-domain algorithm in Hamon et al. (1989) and will be outlined below. Details of other variants of PSOLA, such as frequency-domain PSOLA, are described by Moulines & Charpentier (1990).

The PSOLA method used for manipulating the pitch of a speech utterance is as follows: The original speech, $s(n)$, was converted into a sequence of overlapping short-term (ST) signals, $s_m(n)$, by multiplying by a pitch-synchronous sequence of analysis windows

$$s_m(n) = h_m(t_m - n)s(n) \tag{3.67}$$

The analysis windows, $h_m(n)$, were centred on pitch marks, t_m, which were located at the instants of glottal closure for voiced speech or at an arbitrary rate on unvoiced speech. A Hanning window was used, with length equal to twice the local pitch period so that adjacent ST-signals overlap. A synthesis pitch contour was defined by specifying new values of pitch for each frame of an utterance and this was converted into a sequence of synthesis pitch marks, $\tilde{t}_q$, with the spacing set to the synthesis pitch period for voiced frames, and an arbitrary spacing for unvoiced frames. A sequence of ST-synthesis signals, $\tilde{s}_q(n)$, was formed by copying the analysis ST-signal with pitch mark closest to $\tilde{t}_q$, to the synthesis time-axis, as illustrated in Fig. 3.28. This is equivalent to defining a delay between each synthesis ST-signal and its corresponding analysis ST-signal

$$\tilde{s}_q(n) = s_m(n - t_m + \tilde{t}_q) \tag{3.68}$$

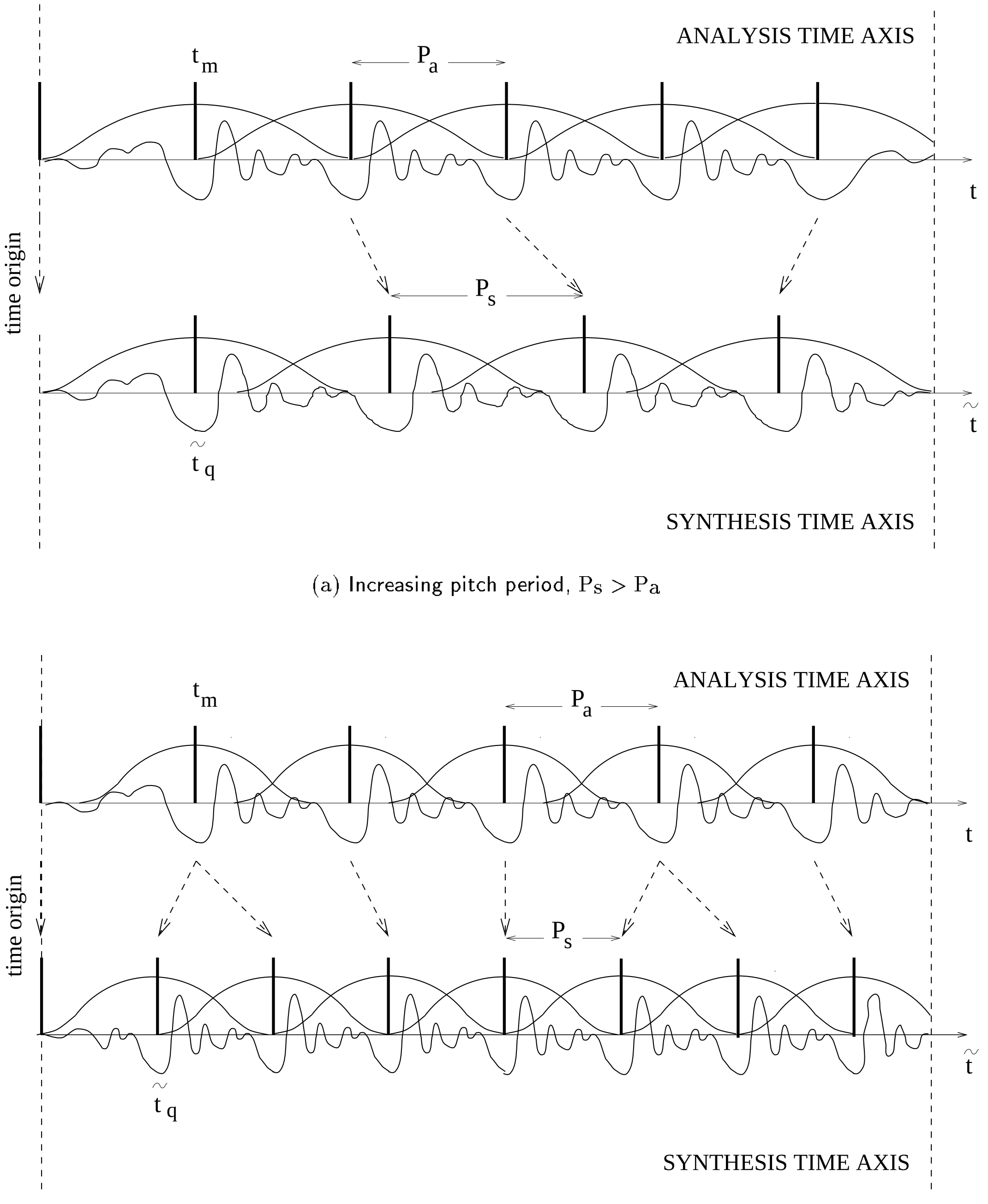
The synthetic speech, $\tilde{s}(n)$, was constructed by simply adding up the contributions from all ST-signals

$$\tilde{s}(n) = \sum_q \alpha_q \tilde{s}_q(n) \tag{3.69}$$

where the normalisation factor, α_q, which compensates for the energy modifications due to altering the pitch, was set to 1. This is known as the simplified OLA method. Other more complicated procedures, such as the least-squares OLA and the simple OLA method (Moulines & Charpentier 1990), were also tested, but for the range of pitch factors used, there was no significant difference in the perceptual quality of synthetic speech from all methods.

3.11.2 Pitch Manipulation Performance

The effect of varying the pitch of an utterance was first evaluated by examination of spectrograms of the synthetic speech produced by the different models when the pitch of the excitation was modified by a constant scale factor. Fig. 3.29 shows a comparison of spectrograms for the utterance "Germany's decision followed eight years later", synthesised at $1.25T_0$, where T_0 is the original pitch period. For all models and for the PSOLA method,



(a) Increasing pitch period, Ps > Pa

(b) Decreasing pitch period, Ps < Pa

Figure 3.28: Modifying the pitch of voiced speech using the PSOLA method. Short-term analysis signals, formed by windowing at the original pitch-synchronous rate, P_a, are copied to the synthesis time axis at the synthesis pitch-synchronous rate, P_s, and assembled using an overlap-add procedure.

the original formant structure of the speech is maintained in the pitch manipulated synthesis, but the vertical pitch striations are farther apart than in the spectrograms of the synthesis produced at the original pitch (Fig. 3.23), due to the longer pitch period. This is illustrated, for example, around 2.4ms.

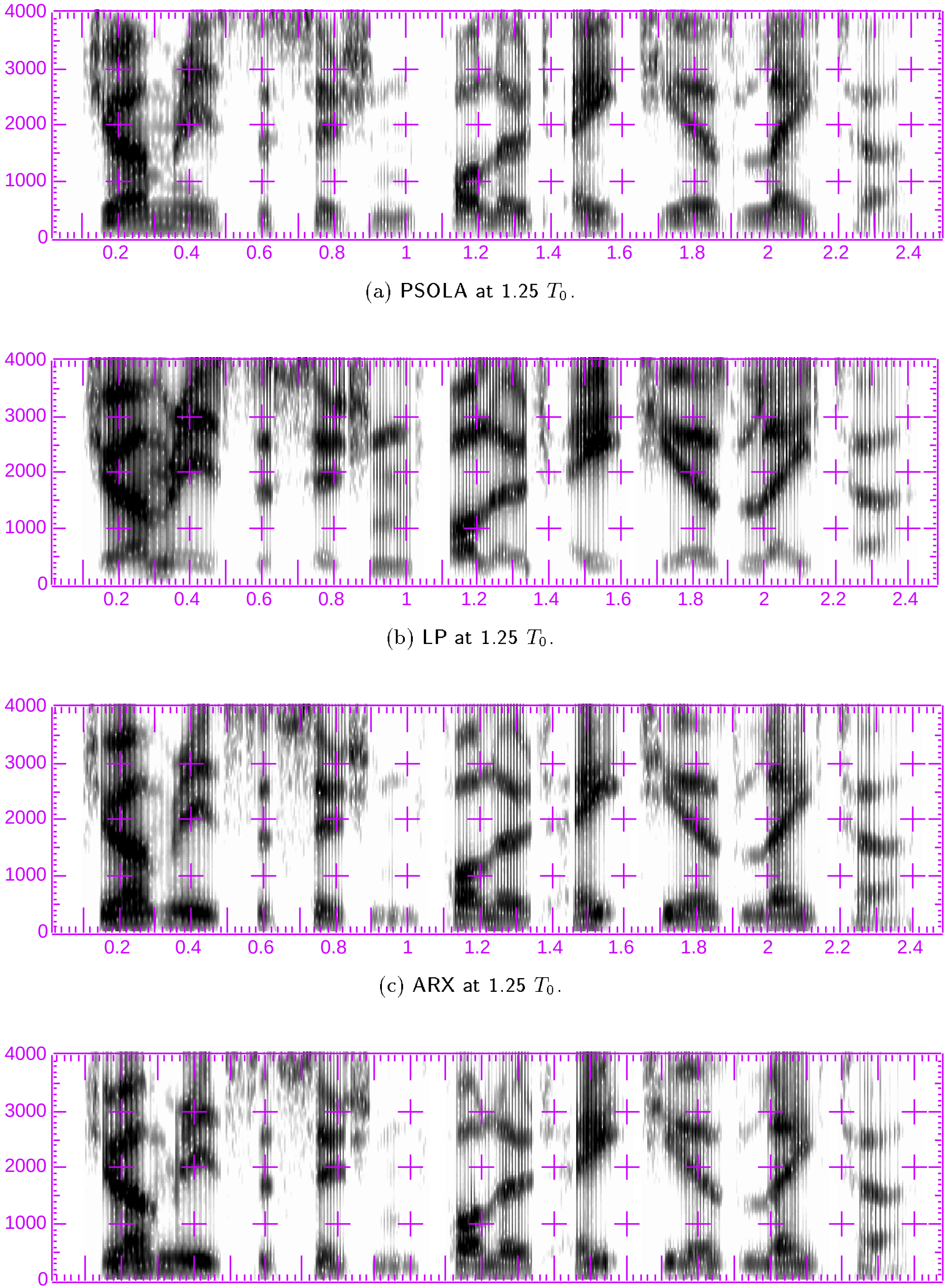
The spectrogram for linear prediction, Fig. 3.29(b), shows that the position of the first formant is altered when the pitch of the impulse excitation is changed. (Compare this spectrogram with that of the original utterance, Fig. 3.23(a), at around 0.8ms for example.) In comparison, the spectrograms of ARX and OE syntheses, Figs. 3.29(c) and 3.29(d), show improved synthesis in the region of the first formant.

The perceptual quality of pitch manipulated synthetic speech was evaluated subjectively by modifying the pitch contours of several utterances and listening to the resulting synthesis from different models. The tape demonstration included in Appendix B contains several examples of applying falling, rising, rise-fall and fall-rise pitch contours to the synthetic speech. For linear prediction models, even altering the pitch of the excitation by ±20% of the original pitch period resulted in unnatural sounding synthetic speech which was poorer in quality than that of synthetic speech produced from ARX models or by the PSOLA method.

The pitch manipulated synthetic speech obtainable from ARX models sounded more natural than that from LP models and was comparable in quality with that obtainable by the PSOLA method. The advantage of using ARX models, however, is they provide a parametric representation of speech waveforms, which is of benefit for applications such as text-to-speech synthesis, in which data compression is an issue. Both ARX and PSOLA methods have the limitation that accurate location of the instants of glottal closure for an utterance are required. By gradually scaling the pitch contours of several utterances, it was estimated that both ARX and PSOLA can support pitch modifications of up to ±40% of the original pitch period, without appreciable distortion. Distortion became quite severe at modifications of ±60% of the original pitch period. The nature of the distortion was very different, sounding metallic (decreasing T_0) or reverberant (increasing T_0) for the PSOLA method , and halting (decreasing T_0) or creaky (increasing T_0) for ARX.

OE models were much less robust to pitch manipulation than ARX models and the resulting synthesis often suffered from glitches and spikes. For regularized models, the pitch manipulation performance was comparable to that with ARX models.

Only informal listening tests were used to compare the subjective quality of pitch manipulated synthetic speech. More rigorous subjective comparisons, such as those suggested in Deller et al. (1993), are needed to assess the full potential of the pitch manipulation capabilities of ARX models.



(a) PSOLA at 1.25 T_0.

(b) LP at 1.25 T_0.

(c) ARX at 1.25 T_0.

(d) OE (with constraints) at 1.25 T_0.

Figure 3.29: Spectrograms of pitch manipulated synthesis from different models. Horizontal axis shows time in seconds, vertical axis shows frequency in Hz.

3.12 Concluding Remarks

3.12.1 Evaluation of Performance

This chapter has compared the performance of different types of linear model of the vocal tract and has demonstrated that minimising a mean squared synthesis error criterion does not necessarily lead to perceptually higher quality synthetic speech. Although OE models consistently improve synthesis SNR over that of ARX or LP models, the resulting syntheses suffered from distortion. In contrast, the syntheses produced from ARX models sounded very natural and were superior to those from LP models, in particular when pitch modifications were applied. The difference in performance of ARX and OE models was attributed to three main factors:

- iterative vs closed form parameter estimation procedure
- different frequency distribution of bias in estimation of transfer function
- different model noise

It was shown that when an iterative procedure is used to estimate parameters, convergence to local minima results in poor estimates of the vocal tract transfer function. The position of local minima, and the global minimum, is influenced by the frequency bias of the transfer function estimate. For OE models, the bias is dependent on the spectrum of the excitation, which for voiced speech, falls off at higher frequencies. It was shown that, as a result of this, there was frame-to-frame variation in the estimates of higher frequency formants in particular, and that these formants often exhibited unnaturally narrow bandwidths. ARX models were demonstrated to be more suitable for speech processing applications than OE models, because their model noise spectrum and frequency distribution of bias in transfer function estimates are perceptually more relevant. For ARX models, the spectrum of the model noise is shaped to the formant structure of the synthetic speech, $1/A(e^{j\omega}, \boldsymbol{\theta})$, which allows the masking properties of that signal to be exploited in reducing the loudness of the noise. However, this shaping results in an LP synthesis SNR that is approximately constant across the frequency range of interest, which does not reflect the varying sensitivity of the ear at different frequencies.

Atal & Schroeder (1979) have shown that a perceptually better synthesis can be obtained for noise with a spectrum $A(e^{j\omega}, \boldsymbol{\theta}, \alpha)/A(e^{j\omega}, \boldsymbol{\theta})$ than for noise with spectrum $1/A(e^{j\omega}, \boldsymbol{\theta})$, even when the latter has a higher SNR. Thus a more suitable weighting filter may be one which results in a synthesis error with spectrum $A(e^{j\omega}, \boldsymbol{\theta}, \alpha)/A(e^{j\omega}, \boldsymbol{\theta})$. This shaping can be achieved using a suitable pre-emphasis filter.

3.12.2 Choosing a Suitable Pre-emphasis Filter

The effect of pre-emphasis on the spectrum of the synthesis error is analogous to that of perceptual weighting, which was discussed in section 3.3.1. In this study, a fixed pre-emphasis filter, $P(q) = 1 - 0.95q^{-1}$, was used. The effect of this filter is to boost the frequency bias function at higher frequencies and to contribute a −6dB/octave (approximately) shift to the model noise spectrum. This was shown to improve estimates of the transfer function in regions of higher frequency. Further improvements in performance might be achieved by considering other forms of pre-filter which allow perceptually relevant characteristics of hearing to be exploited. As described in section 2.4.2, a perceptual weighting filter which is typically used in speech coding applications is $W(q) = A(q)/A(q/\alpha)$, where α is a bandwidth expansion factor. This gives an output error with spectrum

$$\Phi_{\mathrm{ER}}(\omega, \boldsymbol{\theta}) = A(e^{j\omega}, \boldsymbol{\theta}, \alpha)/A(e^{j\omega}, \boldsymbol{\theta}) \tag{3.70}$$

The analogous form for OE models is $F(e^{j\omega}, \boldsymbol{\theta}, \alpha)/F(e^{j\omega}, \boldsymbol{\theta})$. To obtain ARX models with a noise spectrum of this form requires pre-emphasis by a filter $P_{arx}(e^{j\omega}, \boldsymbol{\theta})$ such that

$$\begin{aligned} P_{arx}(e^{j\omega}, \boldsymbol{\theta}) &= 1/A(e^{j\omega}, \boldsymbol{\theta}, \alpha) && (3.71) \\ \tilde{Q}_{arx}(\omega, \boldsymbol{\theta}) &= \|X(e^{j\omega})\|^2 \|A(e^{j\omega}, \boldsymbol{\theta})\|^2 / \|A(e^{j\omega}, \boldsymbol{\theta}, \alpha)\|^2 && (3.72) \\ \tilde{N}_{arx}(e^{j\omega}, \boldsymbol{\theta}) &= A(e^{j\omega}, \boldsymbol{\theta}, \alpha)/A(e^{j\omega}, \boldsymbol{\theta}) && (3.73) \end{aligned}$$

Since the pre-emphasis filter is dependent on the model parameters, the system of equations is nonlinear and an iterative technique is required for solution. The model essentially defines an ARMA description of the vocal tract system,

$$A(q)y(t) = B(q)x(t) + \frac{C(q)}{D(q)}e(t)$$

where the noise model $N(q) = C(q)/D(q)$ is constrained to be a function of $A(q)$.

For OE models, the parameter estimation procedure does not constrain the spectrum of the synthesis error to match the model noise, unless the model noise is made to be dependent on the model parameters. This is the case if the pre-filter is dependent on these parameters. To obtain an OE model with the desired noise spectrum requires a pre-filter such that

$$\begin{aligned} P_{oe}(e^{j\omega}, \boldsymbol{\theta}) &= F(e^{j\omega}, \boldsymbol{\theta})/F(e^{j\omega}, \boldsymbol{\theta}, \alpha) && (3.74) \\ \tilde{Q}_{oe}(\omega, \boldsymbol{\theta}) &= \|X(e^{j\omega})\|^2 \|F(e^{j\omega}, \boldsymbol{\theta})\|^2 / \|F(e^{j\omega}, \boldsymbol{\theta}, \alpha)\|^2 && (3.75) \\ \tilde{N}_{oe}(e^{j\omega}, \boldsymbol{\theta}) &= F(e^{j\omega}, \boldsymbol{\theta}, \alpha)/F(e^{j\omega}, \boldsymbol{\theta}) && (3.76) \end{aligned}$$

The frequency bias, $\tilde{Q}_{oe}(\omega, \boldsymbol{\theta})$, gives de-emphasis of formant regions. Using this pre-filter, the OE model defines a Box-Jenkins model of the vocal tract system

$$y(t) = \frac{B(q)}{F(q)}x(t) + \frac{C(q)}{D(q)}e(t)$$

where the noise model $N(q) = C(q)/D(q)$ is constrained to be a function of $F(q)$.

Comparison of the above equations for $\tilde{Q}$ and $\tilde{N}$ show that the OE and ARX models have been converted to the same underlying model structure by careful choice of pre-filter. This link can be developed further by considering the effect on OE estimation of a simplified pre-emphasis filter, $P(q) = F(q)$. Filtering the output error by $F(q)$ at every iteration of the Gauss-Newton estimation transforms the parameter estimation procedure into an equation error minimisation problem. This is seen by comparison of the filtered synthesis error, $e_{\mathrm{F}}(t)$, which is minimised in parameter estimation, and the link between prediction and synthesis error given in Eqn. (3.27)

$$\begin{aligned} e_{\mathrm{F}}(t) &= F(q)e_s(t) && (3.77) \\ e_p(t) &= \tilde{N}^{-1}(q)e_s(t) && (3.78) \\ &= F(q)e_s(t) && (3.79) \end{aligned}$$

The resulting model parameters define an ARX model of the system, which have been estimated by an iterative procedure. Thus the ARX model can loosely be interpreted as a 'perceptually weighted' OE model.

3.12.3 Speech Coding

Black-box models are of potential use in speech coding applications, for improving the perceptual quality and naturalness of the decoded (synthesised) speech. For such applications, quantization of the model parameters is required. Additional parameters, over those for a linear prediction based system, are the glottal pulse parameters and the coefficients of $B(q)$. For a fixed pulse shape, only transmission of the local pitch is required, which incurs no additional bits. For variable pulse shapes, previous work by Hedelin (1984), using a fixed $B(q)$, has studied the effect of quantizing glottal timing parameters. Further study of how quantization of variable $B(q)$ affects synthesis quality would be of use.

3.12.4 Limitations of the Linear Vocal Tract System

The vocal tract modelling framework has several limitations, which include

- A simplified excitation model. The glottal volume velocity wave model was simplified by using a fixed pulse shape and setting the opening and closing durations to be a constant fraction of the local pitch period. This representation has limited capabilities for representing the true excitation, in which the shape of glottal volume velocity pulses changes with pitch period and other factors such as stress and loudness.

- Use of external timing signal. Laryngograph signals were used to locate the instants of glottal closure for speech waveforms. In this implementation, it was assumed that the offset arising from the propagation delay between the glottis and lips remained constant for the duration of the utterance. This is a limitation because it is unlikely that the lip to microphone distance will remain constant for long utterances. Misalignment between the instants of glottal closure of the excitation model and the true instants of glottal closure for the speech signal can result and lead to inaccurate estimates of the vocal tract transfer function. The need for an additional timing signal in this modelling framework would be a disadvantage if it were to be applied to speech coding systems, for example. Location of the instants of glottal closure from the speech signal by inverse-filtering would be preferable in this case.

- Fixed model order. The model order should reflect the number of formants and anti-resonances that are expected in the data, and should therefore vary with time. In this implementation, a fixed model order, large enough to accommodate the highest expected number of formants and anti-resonances expected in an utterance, was used. Although over-parametrizing the model was shown to reduce the prediction SNR, it was also observed that increasing n_b resulted in unnatural spectral dips in the transfer function estimate. For ARX models, it is favourable to limit n_b to maintain good numerical stability. In modelling long utterances, the selection of a suitable model size is thus a compromise; a sufficient number of parameters are needed to model nasals, without degrading the performance on vowels.

- Linear vocal tract model. The performance of a linear model of a system that is known to be nonlinear is inherently limited, since it can fail to capture the underlying dynamics of the system.

Despite the limited flexibility of the glottal volume velocity wave model and a pulse location which is at best only accurate to within one sampling period, the $4dB$ improvement in prediction gain of the ARX models over linear prediction, which was achieved in this work, is comparable to that reported in similar work in which more accurate excitation models and alignments were used. Hedelin (1984) reported a 2-7dB improvement in prediction gain for a model of order $n_a = 10, n_b = 2$, and used a recursive procedure to re-estimate the glottal volume velocity wave model and ARX parameters. Fujisaki & Ljungqvist (1986) reported $3 - 4dB$ improvement in prediction SNR in their evaluation of the effect of different pulse shapes on the performance of an ARX model. They used very accurate positioning of the pulses, down to 1/10th the sampling period. Using an excitation model based on polynomial basis functions and an extensive search procedure for accurate location of the instants of glottal opening and closure, Thomson (1992) also reported improvement in prediction SNR in the region of 3dB. This implementation is also less computationally intensive because it does not require an exhaustive search for optimum parameters for the excitation model. This is avoided by using a laryngograph to locate the instants of glottal closure, and by assuming a fixed alignment between this signal and the speech waveform for the duration of the utterance. The need for a laryngograph signal in the analysis stage is not such a drawback because it is easily obtained and is not required for synthesis.

The nonlinearities and non-stationarity of speech are not well modelled by a short-term linear analysis. This provides motivation for the use of nonlinear models, which can be implemented, for example, using neural networks. In the following chapter, their application to speech synthesis using the vocal tract modelling framework developed in this chapter, is explored.

Chapter 4

Neural Network Models of the Vocal Tract

A limitation of linear modelling of speech signals, which was considered in the previous chapter, arises from the inherent nonlinearities of the vocal tract system. This chapter investigates nonlinear neural network models of the vocal tract. The single hidden layer multilayer perceptron architecture with sigmoidal nonlinearities, which is widely used in this area, is considered, and is studied in feed-forward and recurrent form.

One observation in this chapter is that, when used in modelling continuous data systems, the above neural network architectures converge to solutions in which the sigmoidal nonlinear units operate outside their saturation regions. Motivated by this observation, methods are developed for initialising network weights from linear models, derived by a linear system identification of the data. The advantages of such initialisation, over initialisation from arbitrary random values, are discussed. A modification of the recurrent architecture, which enables linear initialisation, is proposed.

The quality of speech synthesis from feed-forward and recurrent architectures is described, and demonstrated on the audio tape. The need for learning rate adaptation to enhance the gradient descent training process, and regularization of the error criterion to exploit the correlation across adjacent frames of data are discussed, and are demonstrated to be beneficial in improving the quality of synthetic speech.

4.1 Difficulties in Modelling Long Utterances with Neural Networks

Short-time analysis, which is typically used in processing long speech utterances, presents difficulties for modelling speech with neural networks for two main reasons. Firstly, many models (one for each frame) have to be trained. The back-propagation algorithm, which is typically used, is slow for large architectures. To keep training times to a minimum, the number of epochs of training (the number of times a frame of training data is presented to the network) can be reduced. Reducing the number of training epochs limits the possible change in the values of weights from their initial values and requires good initial estimates of these weights. A second problem is the limited data (the current analysis window) on which to train each network. This implies that the size of networks, in terms of number of parameters, must be limited to ensure reasonable estimates of the parameters. Typically the ratio of data points to parameters for reasonable training is 10:1. For an analysis window of 25-30ms duration, this suggests an upper limit on the network size of 20-25 parameters for data sampled at 8kHz. Architectures were kept small by using only a single hidden layer of nonlinear units. Small networks are beneficial for data compression and reduce the potential for overfitting the data.

4.2 The Neural Network Model

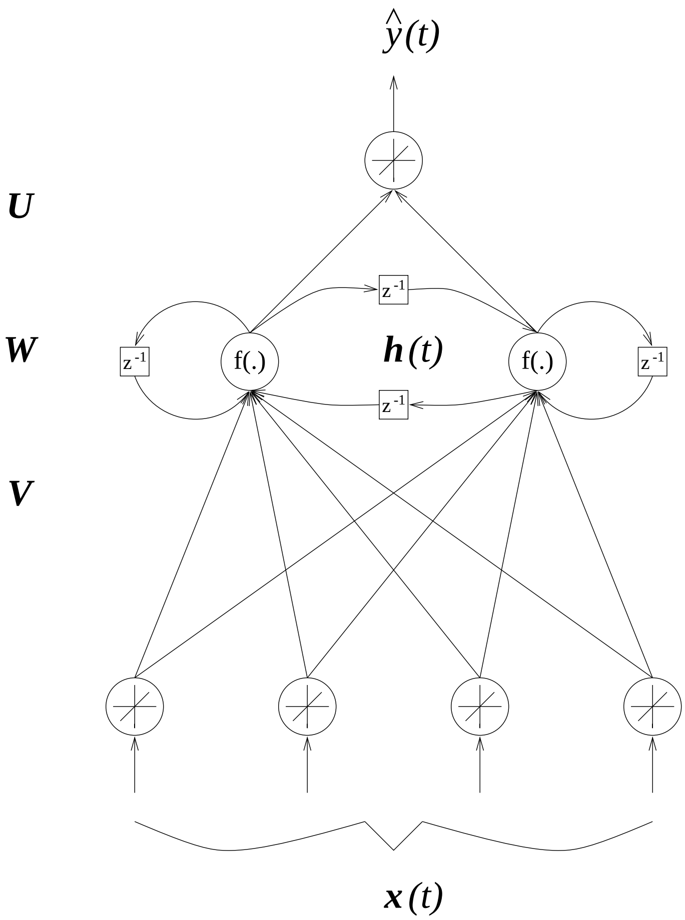


Figure 4.1: Structure of a neural network model of the vocal tract.

The single hidden layer neural network architecture of Fig. 4.1 was used to model the vocal tract. The network defines a nonlinear mapping between the input vector, $\boldsymbol{x}(t)$, and the network output, $\hat{y}(t)$. The network has a small number of nodes in the hidden layer, n_h, a single linear output node, and a linear input layer with n_i nodes, where n_i is given by $n_i = n_b + n_a$ or $n_i = n_b + n_a + 1$ with inclusion of a bias term. z^{-1} represents the z-transform delay operator[1]. $\boldsymbol{V}$, $\boldsymbol{U}$, and $\boldsymbol{W}$ are the input, output, and feedback weight matrices, $\boldsymbol{h}(t)$ is a vector of hidden node outputs and $f(.)$ is a differentiable nonlinear function. In this case, $f(x) = \tanh(x)$. The network output is given by Eqns. (4.1) and (4.2)

$$\boldsymbol{h}(t) = f(\boldsymbol{V}\boldsymbol{x}(t) + \boldsymbol{W}\boldsymbol{h}(t-1)) \tag{4.1}$$

$$\hat{y}(t) = \boldsymbol{U}\boldsymbol{h}(t) \tag{4.2}$$

Several different input vectors were considered for vocal tract modelling, and are based on regression vectors. Input vectors $\boldsymbol{x}_1(t)$ and $\boldsymbol{x}_2(t)$ are equivalent to the regression vectors for ARX and OE models respectively.

$$\begin{aligned}
\boldsymbol{x}_1(t) &= [y(t-1) \ldots y(t-n_a)\; x(t) \ldots x(t-n_b+1)\,]^{\mathrm{T}} && (4.3)\\
\boldsymbol{x}_2(t) &= [\hat{y}(t-1) \ldots \hat{y}(t-n_a)\; x(t) \ldots x(t-n_b+1)\,]^{\mathrm{T}} && (4.4)\\
\boldsymbol{x}_3(t) &= [x(t) \ldots x(t-n_b+1)\,]^{\mathrm{T}} && (4.5)
\end{aligned}$$

where n_a and $n_b - 1$ are the number of past output and past input samples used in prediction or synthesis of the current output sample. The input vector can be augmented by a fixed input of magnitude 1 to represent input biases. The input vector and the feedback weights determine the type of architecture, feed-forward or recurrent. For input vectors $\boldsymbol{x}_1(t)$ or $\boldsymbol{x}_3(t)$, a feed-forward network is represented by setting $\boldsymbol{W} = 0$. For input vector $\boldsymbol{x}_2(t)$, the network is recurrent, regardless of $\boldsymbol{W}$. For $\boldsymbol{W} = 0$, the architecture is a Jordan-type recurrent network (Jordan 1986), in which there are recurrent connections from output to input. For $\boldsymbol{W} \neq 0$, the architecture is recurrent, regardless of $\boldsymbol{x}(t)$. For input vectors $\boldsymbol{x}_1(t)$ or $\boldsymbol{x}_2(t)$, the architecture is an Elman-type recurrent network (Elman 1990). The network gives a linear mapping between input and output, $\hat{y}(t) = \boldsymbol{U}\boldsymbol{x}(t)$, when $f(\sigma) = \sigma$, $\boldsymbol{W} = 0$ and $\boldsymbol{V} = I$ (I is the identity matrix).

Networks were trained by back-propagation to minimise the mean squared error over an N-length training sequence, Eqn. (4.6). For recurrent structures, Real Time Recurrent Learning (RTRL) was used (Williams & Zipser 1989*b*) .

$$E = \frac{1}{2}\sum_{t=0}^{N-1}(y(t) - \hat{y}(t))^2 \tag{4.6}$$

[1] z-transform notation will be used throughout this chapter and the vocal tract transfer function will be referred to as $H(z)$.

For recurrent networks, the performance was measured by the mean synthesis SNR, Eqn. (3.29). For feed-forward networks, performance was measured by the mean prediction SNR, Eqn. (3.28). When synthesising speech with feed-forward networks, input $\boldsymbol{x}_1(t)$ was replaced by $\boldsymbol{x}_2(t)$ and the synthesis performance was measured by Eqn. (3.29).

4.2.1 Operation of Hidden Nodes of Small Networks on Continuous Data

Insight into the operation of neural networks can be gained by examination of the outputs of the hidden nodes. Some of the conclusions drawn by previous researchers are as follows.

- For classification tasks, hidden nodes perform linear discriminant analysis of the data by forming a piece-wise linear approximation to the class boundaries (Webb & Lowe 1990).
- For prediction of scalar functions, each hidden node can be interpreted as forming a piece-wise linear approximation to the output data over some small interval of the full data range (Nguyen & Widrow 1990).
- The action of a single hidden layer is to perform principal component analysis of the input data (Elman 1990).
- In auto-association networks, the hidden layer determines a low rank matrix approximation of the output data which is equivalent to singular value decomposition (Bourland & Kamp 1988).
- Singular value decomposition of the input weight matrix or the covariance matrix of the hidden node outputs can be used to prune oversized networks by determining which hidden nodes are important in the approximation procedure (Weigend & Rumelhart 1991, Psichogios & Ungar 1994, Xue et al. 1990).
- Hidden units with recurrent connections attempt to model the pole-zero dynamics of a system (Back & Tsoi 1991*a*).

In the following discussion, the operation of the hidden units of small, single hidden layer networks is considered, for the case when such networks are trained on continuous, real-valued data such as speech. To facilitate this discussion, the range of operation of the nonlinearity, $f(.)$, is divided into three regions, as illustrated in Fig. 4.2. These will loosely be referred to as 'linear', 'nonlinear' and 'saturated'. The linear region refers to that region of the nonlinearity for which a fairly close straight-line approximation of appropriate gradient is possible. The saturated regions refer to regions where the function remains constant and the nonlinear regions refer to curved portions between the linear and saturated regions.

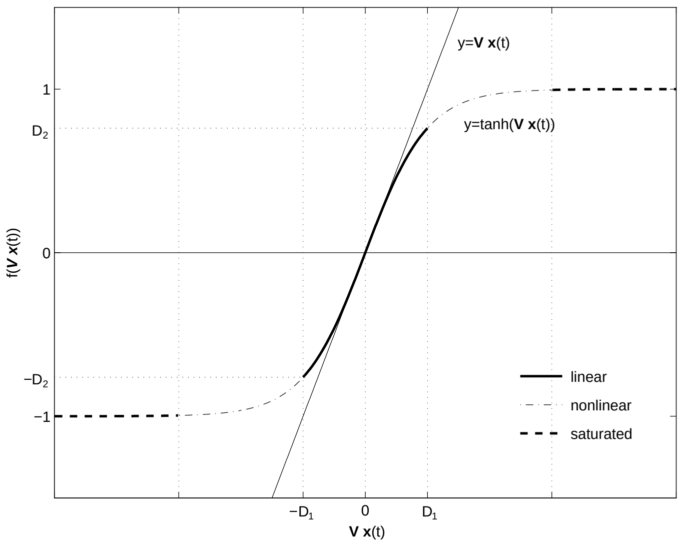


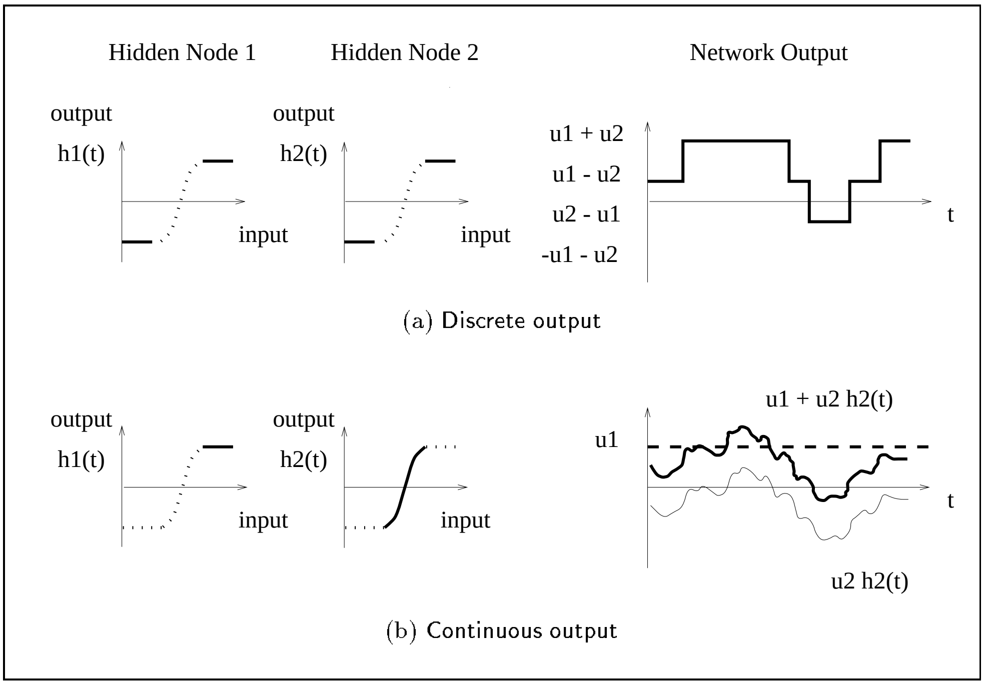
Figure 4.2: Regions of operation of tanh nonlinearity.

In a neural network based model, the nonlinear functions at the hidden nodes usually have saturation regions. Despite these regions, such models must be able to represent continuous functions if they are to be applied to continuous, real-valued signals. Cybenko (1989) and White (1989) have shown that feed-forward networks with a single hidden layer of nonlinear nodes have universal approximation properties. Such networks can approximate continuous functions with arbitrary precision, provided no constraints are placed on the number of hidden nodes or the size of their weights. As the number of hidden nodes tends to infinity, the network can simply represent the function by a piecewise constant approximation, similar to a look-up table. For such a solution, the weights take on large values which cause all the nodes to operate in saturation.

Similarly, for a single hidden layer architecture with a finite number of hidden units, a discrete output with 2^{n_h} levels is obtained when all n_h hidden nodes of the network operate exclusively in saturation regions. Regardless of the number of input nodes, the network operates like an analogue-to-digital converter, as illustrated in Fig. 4.3(a), for $n_h = 2$. Saturation of all nodes can easily occur at the onset of training if the initialisations for input weights are too large.

For small, single hidden layer networks, with only a few hidden nodes, saturation of hidden nodes considerably reduces the flexibility of the network model. As illustrated in Fig. 4.3(b), at least one hidden node must operate out of saturation for the continuous nature of the output to be maintained. A small network is more likely to form a good model of a continuous, real-valued function (which has continuous, non-zero first derivative) if the hidden nodes operate predominantly out of the saturation regions of the sigmoid function.

It is also required that training converge to a network solution which represents the underlying dynamics of the system and which will generate the limit cycles of that system when self-excited (if these exist). A linear network model (with $f(\sigma) = \sigma$) cannot generate



(a) Discrete output

(b) Continuous output

Figure 4.3: Illustration of the operation of hidden nodes of a two node network for tasks involving continuous data. A discrete output results from saturated operation of all hidden units. A continuous output is obtained when operation of hidden units remains out of saturation.

limit cycles, because self-exciting the model by its own output simply produces the impulse response of the linear model, which decays to zero. The predominantly linear operation of the hidden nodes in small feed-forward and recurrent networks, and the generation of limit cycles by such networks, has been demonstrated by Burrows & Niranjan (1993) for prediction of real and simulated time-series data. We showed that generation of limit cycles is still possible, even when hidden nodes operate only slightly towards the nonlinear regions of operation. Even with predominantly linear operation of hidden nodes, we obtained improved prediction performance from feed-forward and recurrent networks over linear networks, for prediction of speech segments from the TIMIT database (Garofolo 1988). Thus, it is beneficial to retain the nonlinear functions despite the predominantly linear operation of these functions.

For the short-term processing of speech utterances, small networks are more desirable than large networks because they are trained more rapidly, require less training data, are easier to analyse and are potentially useful for data reduction applications such as low bit-rate coding. The observation that small networks operate predominantly out of saturation is useful, since it allows experience to be drawn from the field of linear system identification. Linear system identification offers a wide range of well-established theories for studying issues such as model validation, observability and identifiability, suitable model structure, model complexity and parameter bias (Ljung 1987). Many researchers, for example Chen, Billings & Grant (1990), Billings, Jamaluddin & Chen (1992) and Sjöberg (1995), have already successfully applied these theories to small neural networks. An overview of the parallels between neural networks and system identification is given by Ljung & Sjöberg (1992). In the following section, black box linear models are used for providing initial values for network weights.

4.3 Initialisation of Neural Network Weights

4.3.1 Review of Work on Initialisation of Neural Network Weights

Training by back-propagation is not guaranteed to converge to the global minimum of the error criterion, and the minimum to which it converges depends on the initial values of the network weights. Poor local minima often result when the hidden node functions fail to cover the interesting domain of the input space or when duplication of functions occurs (Wessels & Barnard 1992). Traditionally, the initial weights of a network are set to small random values which are drawn from a zero mean, uniform or Gaussian distribution. A suitable scaling is to divide by the fan-in, the number of incoming connections to a node (Weigend & Nix 1994). Random initialisations impose no constraints on how the functions of the hidden nodes are distributed and repeated trials are often needed before an initial set of weights are found from which training converges to a suitable solution.

Previous research into more constrained initialisations for the weights of neural networks has focused on initialisations for single hidden layer feed-forward networks. The methods fall into two main categories: those which aim to distribute the hidden node functions evenly across the regions of the input space which contain data, and those which utilise an existing simpler solution, for example a linear solution, which can be mapped onto the network architecture. The first approach is general, the second is problem dependent and determines the network architecture used.

The first category includes the method of Nguyen & Widrow (1990) and the method of Wessels & Barnard (1992), which were proposed for function approximation and classification respectively.

The method of Nguyen & Widrow (1990) can be used to initialise the input weights and biases of single hidden layer, feed-forward networks which are used for function approximation. For functions of a single variable, the procedure assumes that each hidden node learns to linearly approximate the desired function over some small interval of the input range, and that the desired function is approximated over the whole input range by summing the contributions from each hidden node. The initialisation procedure divides the input range into a number of overlapping intervals and assigns an interval to each hidden node at the start of training. To ensure full coverage of the input range by the hidden node functions, the intervals are uniformly distributed over the input range by drawing the bias weights for the hidden nodes from a uniform distribution. Each hidden node operates linearly over a small interval only. There is therefore some nonlinear operation of hidden nodes at the onset of training. Using this method, Nguyen & Widrow (1990) reported significant improvement in training times, especially for large problems. This method will be used in section 4.3.3, as a comparison for the methods proposed for initialising single hidden layer feed-forward networks from linear models of the vocal tract.

Wessels & Barnard (1992) used a similar strategy for the initialisation of the input weights of single hidden layer feed-forward networks used for 1-out-of-M classification tasks. The procedure ensures that the decision hyperplanes are uniformly orientated throughout the input space and that there is maximum difference between orientations.

The second category includes methods by Hirschauer, Larzabal & Clergeot (1994), Yam & Chow (1995), Goggin, Gustafson & Johnson (1991) and Kulkarni (1991) for function approximation, and methods by Wilson & Tufts (1993) and Niles (1991) for classification.

For system identification tasks, Hirschauer et al. (1994) described a method for initialising the weights of single hidden layer feed-forward networks from a linear least squares solution. This method is described in detail in section 4.3.3, and is referred to as the SVD method because singular value decomposition (SVD) is used to solve the least squares problem and the network weights are derived directly from the components of this decomposition.

Yam & Chow (1995) proposed a similar method for initialising the weights of multilayer feed-forward networks, by sequentially solving a linear least squares problem for each hidden layer. Goggin et al. (1991) used a set of coupled Moore-Penrose pseudo-inverse equations to determine the network weights and optimal number of hidden units for a binary feed-forward network with two layers of nonlinear units. Another method based on singular value decomposition was proposed by Kulkarni (1991) for the initialisation of linear feed-forward networks used to restore images degraded by noise. The images can be restored by inversion of the system impulse response matrix, but direct inversion by singular value decomposition yields unacceptable results due to ill-conditioning. Kulkarni (1991) initialised the weights of the network from the direct inversion.

For classification tasks, an example of a method for initialising the weights of a feed-forward network from an existing solution is that of Wilson & Tufts (1993). They used a singular value decomposition method to initialise single hidden layer feed-forward networks that were trained to detect Gaussian signals in Gaussian noise by a generalised likelihood ratio test.

Previous work on the initialisation of weights of recurrent networks has exploited the links between hidden Markov models (HMM) and neural networks. Niles (1991) used HMMs trained by the maximum likelihood Baum-Welch algorithm to initialise the weights of recurrent networks which were used for classifying phonetic segments. The recurrent network architecture was derived by casting the HMM in neural network terms and the weights were set from the transition and output probabilities of the HMM.

4.3.2 Motivation for Initialisation from Linear Models

The frame-by-frame analysis of speech requires many models to be trained. Using random initialisations of network weights for each frame can be time-consuming as repeated trials may be necessary before training converges to a solution which gives improved performance over a linear model. Random initialisations of network weights have the added disadvantage that they impose no continuity in the values of weights from one frame to the next. In chapter 3, it was shown that such continuity is important for maintaining smooth synthetic speech at frame boundaries and that linear models can already perform the vocal tract modelling task fairly well. Therefore, methods for initialising the weights of feed-forward and recurrent networks from linear models will be investigated in the following two sections. To derive a linear initialisation for network weights, it is necessary to identify links between different neural network architectures and different structures of linear model. An extensive review of the links between different families of linear model and neural network architecture is given by Sjöberg (1995) and also by Nerrand, Roussel-Ragot, Urbani, Personnaz & Dreyfus (1994).

Initialising the weights of networks from linear models has the advantage that the linear initialisation can always provide a fall-back solution for the network weights in the event that training does not converge, or converges to a poorer solution than the original linear model. This is an important feature for the frame-by-frame processing of speech because poor models can affect the performance on future frames through the model memory. However, it will be shown in section 4.7 that a linear initialisation did not always prove successful, especially in cases where the initial linear model gave poor performance.

4.3.3 Weight Initialisations for Feed-forward Networks from ARX Models

In this section, an equivalence between the parameters of a linear ARX model of the vocal tract and a feed-forward architecture is defined. The feed-forward architecture of Fig. 4.1, with $\boldsymbol{W} = \mathbf{0}$ and input $\boldsymbol{x}_1(t)$, is considered. For linear hidden units, the network is equivalent to the linear configuration shown in Fig. 3.2(a). The network output, Eqn. (4.2), can be rewritten as

$$\begin{aligned}\hat{y}(t) &= -\sum_{j=1}^{n_a}\left(-\sum_{i=1}^{n_h} u_i v_{ij}\right) y(t-j) \\ &\quad + \sum_{k=0}^{n_b-1}\left(\sum_{i=1}^{n_h} u_i v_{i(n_a+k+1)}\right) x(t-k) \qquad (4.7)\end{aligned}$$

where u_i, the i^{th} element of $\boldsymbol{U}$, represents the connection from the i^{th} hidden node to the output and v_{ij}, the ij^{th} element of $\boldsymbol{V}$, represents the connection from input j to hidden node i. Comparison with the prediction component of Eqn. (3.3) gives direct equivalence to an ARX model for

$$a_j = -\sum_{i=1}^{n_h} u_i v_{ij} \qquad j = 1 \dots n_a \tag{4.8}$$

$$b_k = \sum_{i=1}^{n_h} u_i v_{i(n_a+k+1)} \qquad k = 0 \dots n_b - 1 \tag{4.9}$$

Arranging the parameters of the ARX model into a vector, $\boldsymbol{\theta}_{arx}$, Eqn. (3.10), the linear equivalence is given by

$$\boldsymbol{\theta}_{arx}^{\mathrm{T}} = \boldsymbol{U}\boldsymbol{V} \tag{4.10}$$

This can be satisfied by many possible choices of $\boldsymbol{U}$ and $\boldsymbol{V}$. For an arbitrary choice, the output from the network and the linear model may not be similar, due to saturation or nonlinear operation of hidden nodes. For the SNR of the network and linear model to be similar in magnitude, so that training the network improves on the performance of the linear model, the network output should operate like a linear function for the chosen combination of $\boldsymbol{U}$ and $\boldsymbol{V}$. A choice of weights for the network output to be approximately linear is one for which all hidden nodes operate in the linear region. This requires scaling of $\boldsymbol{U}$ and $\boldsymbol{V}$ so that for all hidden nodes i

$$\boldsymbol{v}_i \boldsymbol{x}(t) \leq D_1 \qquad t = 0 \dots N - 1 \tag{4.11}$$

$$u_i \boldsymbol{v}_i \boldsymbol{x}(t) \leq D_2 \qquad t = 0 \dots N - 1 \tag{4.12}$$

where D_1 and D_2 are illustrated in Fig. 4.2. After an initial choice of $\boldsymbol{U}$ and $\boldsymbol{V}$, rescaling by an $(n_h \times n_h)$ non-singular matrix, T, can be carried out so that these criteria are met for all input vectors, $\boldsymbol{x}(t)$, $t = 0 \dots N - 1$

$$\boldsymbol{\theta}_{arx}^{\mathrm{T}} = \boldsymbol{U}\boldsymbol{T}^{-1}\boldsymbol{T}\boldsymbol{V} \tag{4.13}$$

This method of placing the operation of all hidden nodes in the linear region requires small values for the elements of $\boldsymbol{V}$, which prevents saturation of nodes at the onset of training. The network linearisation of Eqn. (4.7) assumes that the hidden node function is $f(x) = \tanh(x)$. For a general sigmoid function with linearised equation $f(x) = mx + k$, the initialisation is modified to

$$\boldsymbol{\theta}_{arx}^{\mathrm{T}} = m\boldsymbol{U}\left[\boldsymbol{V} + \boldsymbol{J}\right] \tag{4.14}$$

where $\boldsymbol{J}$ is a matrix which adds an additional contribution to $\boldsymbol{V}$, to account for k. All elements of the last column of $\boldsymbol{J}$ are set to k/m and the remaining elements of $\boldsymbol{J}$ are set to zero. For the case where a bias term is included in $\boldsymbol{x}(t)$ and $\boldsymbol{V}$, $\boldsymbol{J}$ is of dimension $(n_h \times n_i)$. When there are no bias terms in $\boldsymbol{x}(t)$ and $\boldsymbol{V}$, $\boldsymbol{J}$ is of dimension $(n_h \times (n_i + 1))$ and thus appends an additional column to $\boldsymbol{V}$.

As described in section 3.3, the ARX parameters can be found in closed-form by the least squares method. In matrix notation, the least-squares error criterion is written as

$$V_N(\boldsymbol{\theta}) = \frac{1}{2}||\boldsymbol{Y} - \Phi_N \boldsymbol{\theta}||^2 \tag{4.15}$$

$$\begin{aligned} \boldsymbol{Y} &= [y(0) \ \ldots \ y(N-1)]^{\mathrm{T}} && (4.16) \\ \Phi_N &= [\boldsymbol{x}(0) \ldots \boldsymbol{x}(N-1)]^{\mathrm{T}} && (4.17) \end{aligned}$$

The solution minimising Eqn. (4.15) is given by

$$\boldsymbol{\theta}_{arx}^{\mathrm{T}} = \boldsymbol{Y}^{\mathrm{T}}[\Phi_N^{-1}]^{\mathrm{T}} \tag{4.18}$$

For the case where the system of equations is over-determined $(N > n_a + n_b)$, Φ_N^{-1} is replaced by the Moore-Penrose pseudo-inverse, $\Phi_N^{+} = [\Phi_N^{\mathrm{T}} \Phi_N]^{-1} \Phi_N^{\mathrm{T}}$, which gives the solution with minimum norm.

$$\boldsymbol{\theta}_{arx}^{\mathrm{T}} = \boldsymbol{Y}^{\mathrm{T}}[\Phi_N^{+}]^{\mathrm{T}} \tag{4.19}$$

Φ_N^{+} can be found by Singular Value Decomposition, QR factorisation, Cholesky Decomposition or Gauss Elimination and the methods differ in numerical stability and computational requirements. $\boldsymbol{U}$ and $\boldsymbol{V}$ can be calculated directly from the linear equations without explicitly calculating $\boldsymbol{\theta}_{arx}$. There are several possibilities for splitting $\boldsymbol{\theta}_{arx}$ into $\boldsymbol{U}$ and $\boldsymbol{V}$, which follow from the different methods for calculating Φ_N^{+}.

SVD Method

This method was used by Hirschauer et al. (1994) and extended to feed-forward networks of more than one hidden layer by Yam & Chow (1995). Singular value decomposition (SVD) of Φ_N gives

$$\begin{aligned} \Phi_N^{\mathrm{T}} &= \boldsymbol{P} \Sigma \boldsymbol{S}^{\mathrm{T}} && (4.20) \\ \boldsymbol{\theta}_{arx}^{\mathrm{T}} &= \boldsymbol{Y}^{\mathrm{T}} \boldsymbol{S} \boldsymbol{T}^{-1} \boldsymbol{T} \Sigma^{-1} \boldsymbol{P}^{\mathrm{T}} && (4.21) \end{aligned}$$

where $\boldsymbol{P}$ and $\boldsymbol{S}$ are unitary matrices of dimensions $(n_i \times n_i)$ and $(N \times N)$ respectively, and Σ is a diagonal matrix of singular values, of dimension $(n_i \times N)$. For the case $n_h = n_i$, $\boldsymbol{U}$ and $\boldsymbol{V}$ are set directly from these matrices and the initialisation exactly gives the linear ARX model, $\boldsymbol{\theta}_{arx}^{\mathrm{T}} = \boldsymbol{U}\boldsymbol{V}$.

$$U = \boldsymbol{Y}^{\mathrm{T}}\boldsymbol{S}\boldsymbol{T}^{-1} \quad (4.22)$$
$$V = \boldsymbol{T}\Sigma^{-1}\boldsymbol{P}^{\mathrm{T}} \quad (4.23)$$

$\boldsymbol{T}$ is a non-singular scaling matrix of dimension $(n_h \times n_h)$. For $n_h < n_i$, only the first n_h columns of $\boldsymbol{P}$ and $\boldsymbol{S}$, and the first n_h rows and columns of Σ, are used to give matrices $\boldsymbol{P}_{n_h}$, Σ_{n_h}, $\boldsymbol{S}_{n_h}$, which have dimensions $(n_i \times n_h)$, $(n_h \times n_h)$ and $(N \times n_h)$ respectively. In this case, only the largest n_h singular values are considered important and the initialisation $\boldsymbol{UV}$ represents a rank n_h approximation to the best least-squares ARX model given by $\boldsymbol{\theta}_{arx}^{\mathrm{T}}$. The prediction SNR for the approximation, $\boldsymbol{UV}$, is less than that for the original ARX model, $\boldsymbol{\theta}_{arx}^{\mathrm{T}}$.

QR-Factorisation Method

This is simply a variation of the previous method which results in matrices $\boldsymbol{U}$ and $\boldsymbol{V}$ with a different internal structure. Using QR-factorisation of Φ_N to find $\boldsymbol{\theta}_{arx}$ gives

$$\Phi_N^{\mathrm{T}} = \boldsymbol{QR} \quad (4.24)$$
$$\boldsymbol{\theta}_{arx}^{\mathrm{T}} = \boldsymbol{Y}^{\mathrm{T}}\boldsymbol{R}^{\mathrm{T}}\left[\boldsymbol{R}\boldsymbol{R}^{\mathrm{T}}\right]^{-1}\boldsymbol{T}^{-1}\boldsymbol{T}\left[\boldsymbol{Q}^{\mathrm{T}}\boldsymbol{Q}\right]^{-1}\boldsymbol{Q}^{\mathrm{T}} \quad (4.25)$$

where $\boldsymbol{Q}$ is a unitary matrix of dimension $(n_i \times n_i)$ and $\boldsymbol{R}$ is an upper triangular matrix of dimension $(n_i \times N)$. As with the SVD method, for $n_h = n_i$, $\boldsymbol{\theta}_{arx}^{\mathrm{T}} = \boldsymbol{UV}$, and $\boldsymbol{U}$ and $\boldsymbol{V}$ are set from

$$\boldsymbol{U} = \boldsymbol{Y}^{\mathrm{T}}\boldsymbol{R}^{\mathrm{T}}\left[\boldsymbol{R}\boldsymbol{R}^{\mathrm{T}}\right]^{-1}\boldsymbol{T}^{-1} \quad (4.26)$$
$$\boldsymbol{V} = \boldsymbol{T}\left[\boldsymbol{Q}^{\mathrm{T}}\boldsymbol{Q}\right]^{-1}\boldsymbol{Q}^{\mathrm{T}} \quad (4.27)$$

$\boldsymbol{T}$ is a non-singular $(n_h \times n_h)$ scaling matrix. For $n_h < n_i$, the matrices $\boldsymbol{Q}$ and $\boldsymbol{R}$ are truncated to give matrices $\boldsymbol{Q}_{n_h}$ and $\boldsymbol{R}_{n_h}$ which have dimensions $(n_i \times n_h)$ and $(n_h \times N)$ respectively. As before, the initialisation $\boldsymbol{UV}$ represents a rank n_h approximation to $\boldsymbol{\theta}_{arx}^{\mathrm{T}}$ and has a lower prediction SNR.

Using this method, only the leading diagonal of $\boldsymbol{V}$ is non-zero at initialisation. Each hidden node processes only one element of the input. For all inputs to be taken into consideration requires $n_h = n_i$. Rank n_h approximations are equivalent to ignoring certain inputs at initialisation and result in $n_i - n_h$ zeros in $\boldsymbol{\theta}_{arx}$.

ARX-QR Method

This is a new method which is proposed for the initialisation of small, single hidden layer feed-forward networks, such as those used for the vocal tract modelling task. Unlike the SVD and QR methods, the initial values of $\boldsymbol{U}$ and $\boldsymbol{V}$ are such that $\boldsymbol{UV} = \boldsymbol{\theta}_{arx}^{\mathrm{T}}$, even when $n_h < n_i$. The initial matrix $\boldsymbol{V}$ is also full rank[2]. In this method, $\boldsymbol{\theta}_{arx}^{\mathrm{T}}$ was found by singular value decomposition or QR-factorisation (with no truncation). A set of vectors orthogonal to $\boldsymbol{\theta}_{arx}^{\mathrm{T}}$ was found by QR-factorisation of $\boldsymbol{\theta}_{arx}^{\mathrm{T}}$

$$\boldsymbol{\theta}_{arx}^{\mathrm{T}} = \boldsymbol{R}^{\mathrm{T}}\boldsymbol{T}^{-1}\boldsymbol{T}\boldsymbol{Q}^{\mathrm{T}} \tag{4.28}$$

where $\boldsymbol{Q}$ is a unitary matrix of dimension $(n_i \times n_i)$ and $\boldsymbol{R}$ is an upper triangular matrix of dimension $(n_i \times 1)$. For the case $n_h = n_i$, the initial weights are set to $\boldsymbol{V} = \boldsymbol{T}\boldsymbol{Q}^{\mathrm{T}}$ and $\boldsymbol{U} = \boldsymbol{R}^{\mathrm{T}}\boldsymbol{T}^{-1}$. When $n_h < n_i$, $\boldsymbol{Q}$ and $\boldsymbol{R}$ are truncated to dimensions $(n_i \times n_h)$ and $(n_h \times 1)$. By taking the first n_h columns of $\boldsymbol{Q}$ and the first n_h elements of $\boldsymbol{R}$, the first row of $\boldsymbol{V}$ is given by $\boldsymbol{\theta}_{arx}^{\mathrm{T}}/u_1$, and all other rows of $\boldsymbol{V}$ are perpendicular to $\boldsymbol{\theta}_{arx}^{\mathrm{T}}$. $\boldsymbol{U}$ has a single non-zero element, u_1, because $\boldsymbol{R}$ is an upper triangular matrix. Hence even with truncation, $\boldsymbol{UV} = \boldsymbol{\theta}_{arx}^{\mathrm{T}}$ exactly.

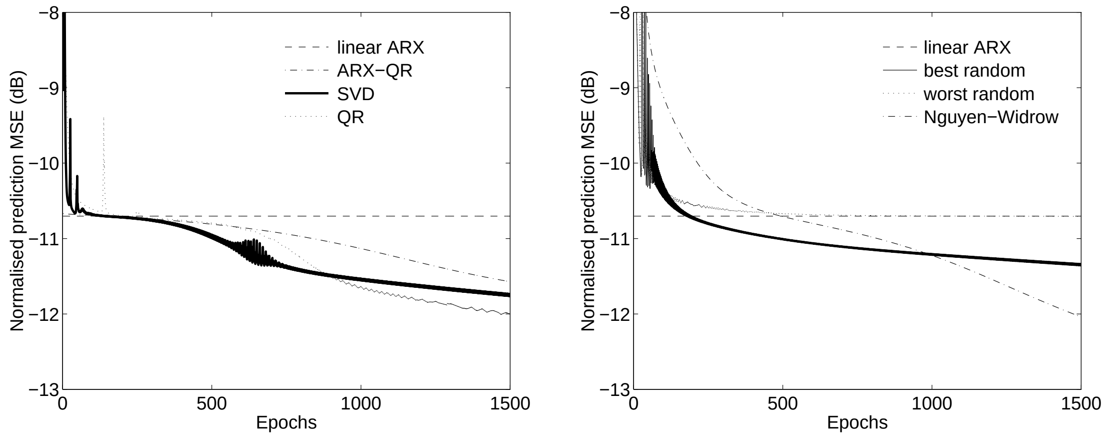
Performance Comparison of Initialisation Methods

The different methods of initialisation of the feed-forward network were tested on the vocal tract modelling problem. The networks were trained by back-propagation to minimise the mean-squared prediction error. $\boldsymbol{U}$ was calculated by the least-squares method (Barton 1991).

Fig. 4.4(a) shows typical learning curves for feed-forward networks trained on a vowel segment. The normalised mean squared prediction error is the negative of the SNR value in Eqn. (3.28). All learning rates, architectures and weight update schemes were identical, the only difference was in the initialisation of the weights. The learning curves were obtained for a feed-forward architecture with $n_h = 5$, $n_i = 13$. An ARX model of size $n_a = 10$, $n_b = 2$, plus a bias term, was used for initialisation and was excited by $dx(t)$. T was taken as a diagonal matrix, and different scale factors in the range $[-1, 1]$ were tested. Only the learning curves for the networks with the best performance are illustrated.

For comparison, Fig. 4.4(b) illustrates the best and worst curves obtained from 20 repeated trials of the random initialisations from a uniform distribution. The method proposed by Nguyen & Widrow (1990), which was described in section 4.3.1, is also shown. For this method, $\boldsymbol{U}$ was initialised from a uniform distribution and the best performance out of 20 trials is plotted in Fig. 4.4(b).

[2] Obvious choices for $\boldsymbol{V}$, which satisfy $\boldsymbol{UV} = \boldsymbol{\theta}_{arx}^{\mathrm{T}}$, are either to set the first row of $\boldsymbol{V}$ to $\boldsymbol{\theta}_{arx}^{\mathrm{T}}$ and all other rows to zero, or to set all rows of $\boldsymbol{V}$ to $\boldsymbol{\theta}_{arx}^{\mathrm{T}}$. The corresponding $\boldsymbol{U}$ is either a vector with a single non-zero element of magnitude 1, or a vector with all elements of equal magnitude, $1/n_h$. These initialisations did not perform well because $\boldsymbol{V}$ was not full rank and did not form a basis for projections of $\boldsymbol{x}(t)$.



(a) Linear initialisations

(b) Random initialisations

Figure 4.4: Comparison of initialisation techniques for feed-forward networks ($n_i = 13$,$n_h = 5$). Linear initialisations were obtained from an ARX model of order $n_a = 10$, $n_b = 2$, plus input bias. Best and worst performance from random initialisations are also shown.

Fig. 4.4 shows that the proposed linear initialisation techniques performed better, on average, than random initialisations of network weights. The variability in performance of networks initialised by linear methods was less than that for networks initialised by the method of Nguyen & Widrow (1990). However, for this segment of data, the method of Nguyen & Widrow (1990) gave the best overall performance. The relative performance of the linear methods was dependent on the data and choice of T. The spike observed in the QR learning curve was due to ill-conditioning of the matrix used in the calculation of $\boldsymbol{U}$. This can cause a sudden large change in $\boldsymbol{U}$, which results in a large change in the prediction error.

4.3.4 Weight Initialisations for Single Delay Recurrent Networks

In this section, new methods for initialising the weights of single hidden layer recurrent networks are developed. Four different recurrent architectures, based on Fig. 4.1, are considered

- **Output feedback** $\boldsymbol{W} = 0$, input $\boldsymbol{x}_2(t)$ Eqn. (4.4)
- **Internal feedback** $\boldsymbol{W} \neq 0$
  - RNN1, input $\boldsymbol{x}_1(t)$ Eqn. (4.3)
  - RNN2, input $\boldsymbol{x}_3(t)$ Eqn. (4.5)
- **Modified architecture**
  - RNN3, $\boldsymbol{x}_3(t)$ Eqn. (4.5)

Output Feedback - Initialisation from OE or ARX Models

Input vector $\boldsymbol{x}_2(t)$, Eqn. (4.4), requires recurrent connections from the network output to the network input. With $\boldsymbol{W} = \mathbf{0}$ and linear hidden units, the network output is given by

$$\begin{aligned} \hat{y}(t) &= -\sum_{j=1}^{n_a}\left(-\sum_{i=1}^{n_h} u_i v_{ij}\right)\hat{y}(t-j) \\ &\quad + \sum_{k=0}^{n_b-1}\left(\sum_{i=1}^{n_h} u_i v_{i(n_a+k+1)}\right)x(t-k) \end{aligned} \tag{4.29}$$

Comparison with Eqn. (3.12) shows this recurrent architecture is equivalent to an OE model (or an ARX model in synthesis mode), which corresponds to the linear configurations of Figs. 3.2(b) (or 3.2(c)). Back-propagation training is analogous to an output error minimisation procedure because the mean-squared synthesis (output) error is minimised in training. Linear initialisation of these networks is not considered here, but can be derived directly from an ARX or OE model as for the feed-forward architecture.

Internal Feedback - Initialisation from State-Space Models

Consider the discrete time state-space innovation form of a dynamical system

$$\boldsymbol{h}(t+1) = \boldsymbol{A}\boldsymbol{h}(t) + \boldsymbol{B}\boldsymbol{x}(t) + \boldsymbol{K}e(t) \tag{4.30}$$

$$y(t) = \boldsymbol{C}\boldsymbol{h}(t) + e(t) \tag{4.31}$$

where $\boldsymbol{h}(t)$ is a $(n_h \times 1)$ vector of internal states, $\boldsymbol{A}$, $\boldsymbol{B}$, $\boldsymbol{C}$ are parameter matrices of dimensions $(n_h \times n_h)$, $(n_h \times n_i)$ and $(n_o \times n_h)$ respectively and n_o, n_h and n_i are the dimensions of the output, state, and input vectors. $\boldsymbol{K}$ is a gain matrix of dimension $(n_h \times n_i)$ which is calculated from $\boldsymbol{A}$, $\boldsymbol{C}$, and the covariance of $e(t)$. Taking z-transforms of Eqns. (4.30) and (4.31), the transfer function[3] and noise model for a state-space representation of the vocal tract system are given by

$$H_{\text{state-space}}(z) = \boldsymbol{C}[z\boldsymbol{I} - \boldsymbol{A}]^{-1}\boldsymbol{B} \tag{4.32}$$

$$N(z) = \boldsymbol{I} + \boldsymbol{C}[z\boldsymbol{I} - \boldsymbol{A}]^{-1}\boldsymbol{K} \tag{4.33}$$

[3] Here, $H(z)$ represents the system transfer function, not the z-transform of the outputs of the hidden units of the recurrent neural network.

For recurrent networks with feedback around the hidden nodes ($\boldsymbol{W} \neq \mathbf{0}$), linearisation of Eqns. (4.1) and (4.2) gives

$$\begin{aligned} \boldsymbol{h}(t) &= \boldsymbol{V}\boldsymbol{x}(t) + \boldsymbol{W}\boldsymbol{h}(t-1) & (4.34) \\ \hat{y}(t) &= \boldsymbol{U}\boldsymbol{h}(t) & (4.35) \end{aligned}$$

These equations are equivalent to Eqns. (4.30) and (4.31) respectively, for $\boldsymbol{K} = 0$. The transfer function for the linearised network is given by

$$H_{\mathrm{net}}(z) = z\boldsymbol{U}\left[z\boldsymbol{I} - \boldsymbol{W}\right]^{-1}\boldsymbol{V} \tag{4.36}$$

Assuming $\boldsymbol{K} = 0$, the network can be initialised from a state-space model by setting

$$\begin{aligned} \boldsymbol{U} &= \boldsymbol{C} & (4.37) \\ \boldsymbol{W} &= \boldsymbol{A} & (4.38) \\ \boldsymbol{V} &= \boldsymbol{B} & (4.39) \end{aligned}$$

The additional z in the transfer function of the linearised network requires that the input used to train the network is a delayed version of that used to generate the state-space model

$$X_{\mathrm{net}}(z) = z^{-1}X_{\mathrm{state\text{-}space}}(z) \tag{4.40}$$

where $X(z)$ is the z-transform of the input. For a general linearised sigmoid function, $f(x) = mx + k$, the z-transform of the linearised network output is

$$Y(z) = z\boldsymbol{U}\left[z\boldsymbol{I} - m\boldsymbol{W}\right]^{-1}\left[\boldsymbol{V}X(z) + k/m\right] \tag{4.41}$$

which requires the initialisation to be modified to

$$\begin{aligned} \boldsymbol{U} &= \boldsymbol{C} & (4.42) \\ \boldsymbol{W} &= \boldsymbol{A}/m & (4.43) \\ \boldsymbol{V} &= [\boldsymbol{B}/m - \boldsymbol{J}] & (4.44) \end{aligned}$$

where $\boldsymbol{J}$ is a matrix of dimension $(n_h \times n_i)$ which subtracts out, from the bias weights in $\boldsymbol{V}$, the contribution which comes from the offset k. All elements of the last column of $\boldsymbol{J}$ are set to k/m and the remaining elements of $\boldsymbol{J}$ are zero.

For recurrent architectures, scaling individual elements of $\boldsymbol{V}$ by different values, to shift the operation of hidden units into linear regions, changes the overall transfer function of the state-space model. The transfer function is unaltered if all elements of $\boldsymbol{V}$ are scaled by the same constant, p say, such that $T = p\boldsymbol{I}$, where $\boldsymbol{I}$ is the identity matrix. All elements of $\boldsymbol{U}$ are scaled by $1/p$ where p is chosen so that the operation of all hidden nodes is shifted into the linear region.

For recurrent architectures RNN1 and RNN2, the performance of state-space initialisations will now be compared with random initialisations of weights. Performance is measured by normalised mean-squared prediction (synthesis) error, which is the negative value of the prediction (synthesis) SNR of Eqn. (3.28) (Eqn. (3.29)). In the following simulations, state-space models were calculated by the prediction-error method, using the Gauss-Newton procedure (Ljung 1987). Initial parameter values were drawn from uniform or Gaussian distributions. To ensure a stable initial state-space model, $\boldsymbol{A}$ was adjusted so that its eigenvalues were all real and of magnitude less than 1. For simulations in which the initial weights were set to random values, the same stability constraints were applied to $\boldsymbol{W}$.

RNN1

For linear hidden units, the output of this architecture is given by

$$\begin{aligned} \hat{y}(t) &= -\sum_{j=1}^{n_a}\left(-\sum_{i=1}^{n_h} u_i v_{ij}\right) y(t-j) \\ &\quad + \sum_{k=0}^{n_b-1}\left(\sum_{i=1}^{n_h} u_i v_{i(n_a+k+1)}\right) x(t-k) - \sum_{j=1}^{n_h}\left(-\sum_{i=1}^{n_h} u_i w_{ij}\right) h_j(t-1) \qquad (4.45) \end{aligned}$$

Setting $\boldsymbol{W} = \boldsymbol{0}$ at initialisation, the architecture is equivalent to an ARX model, Eqn. (3.3). Training by back-propagation minimises the mean-squared prediction error and is analogous to equation error minimisation. For this architecture, an alternative to state-space initialisation was also tested, in which $\boldsymbol{U}$ and $\boldsymbol{V}$ were initialised from an ARX model, as for the feed-forward architecture. $\boldsymbol{W}$ was initialised with very small amplitude random values, applying the constraint that all eigenvalues of $\boldsymbol{W}$ be real and of magnitude less than 1. The determinant of $\boldsymbol{W}$ was adjusted until the prediction SNR for the ARX model and the initial network were similar.

For a segment of voiced speech, 20 networks were trained for each initialisation method and for random initial weights. The best and worst learning curves obtained for each method are shown in Fig. 4.5. The same value of learning rate was used in all trials. For comparison, the prediction SNR for the linear ARX model, and the best and worst performance from linear state-space models, are also shown. An input vector of dimension $n_i = 13$ and an ARX model of dimensions $n_a = 10$, $n_b = 2$, were used. The number of hidden nodes in the networks was set to $n_h = 2$.

Fig. 4.5(b) shows that none of the networks initialised with random weights converged to a prediction MSE below that of the linear ARX model. In comparison, all of those initialised from the ARX model, Fig.4.5(c), and most of those initialised from state-space models, Fig. 4.5(a), gave lower MSE than the ARX model. The best learning curves obtained for each method from the 20 repeated trials are shown in Fig. 4.5(d). Although the best performance from state-space initialised networks was only slightly better than that of the best linear state-space model, the variation in the performance of these network models was much less than that from networks with either random initial weights or initial weights obtained from ARX models.

RNN2

With linear hidden units, the output of RNN2 is given by

$$\hat{y}(t) \;=\; -\sum_{j=1}^{n_h}\left(-\sum_{i=1}^{n_h} u_i w_{ij}\right) h_j(t-1) + \sum_{k=0}^{n_b-1}\left(\sum_{i=1}^{n_h} u_i v_{i(k+1)}\right) x(t-k) \qquad (4.46)$$

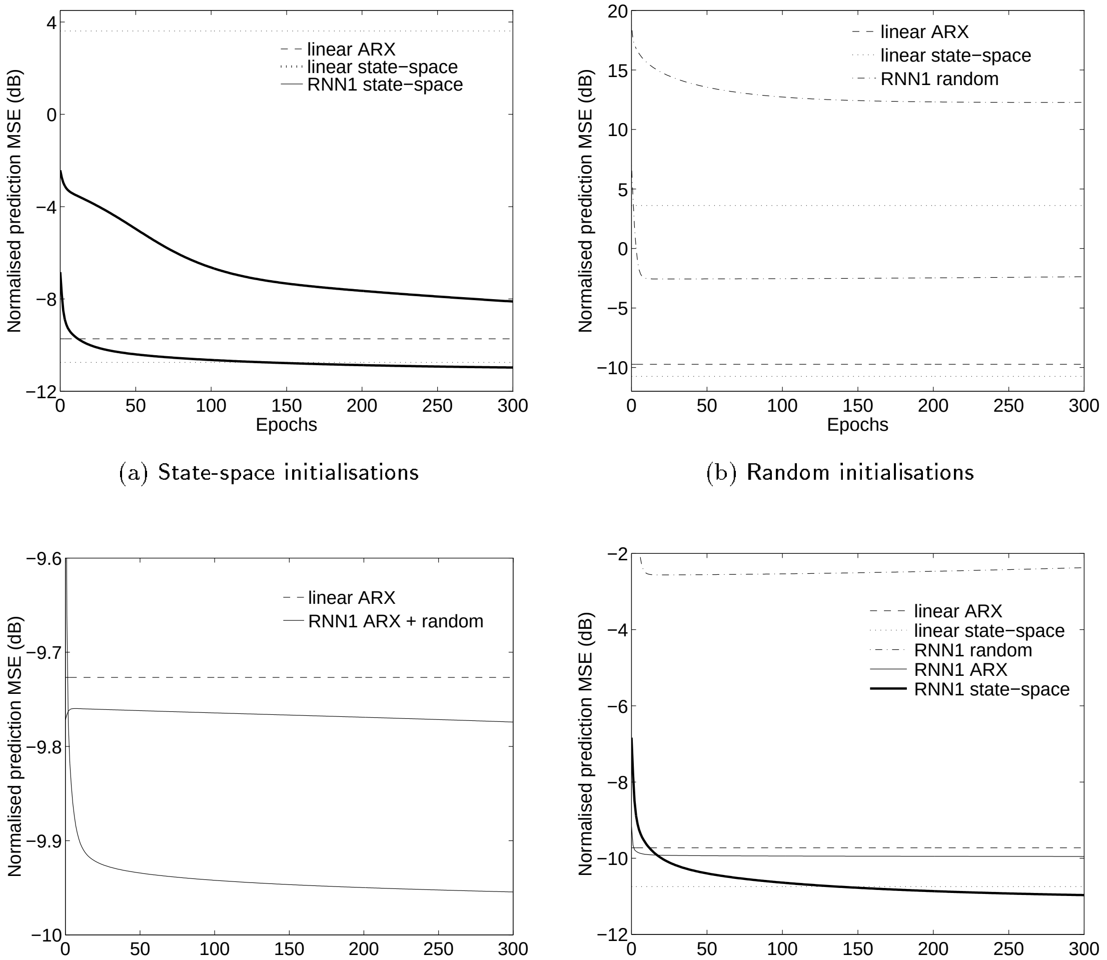
Comparison with Eqn. (3.12) shows that this architecture is not directly equivalent to an OE model, Eqn. (3.12). However, training by back-propagation is analogous to an output error minimisation technique because the mean squared synthesis (output) error is minimised directly in training and the resulting parameters are optimal for synthesis (in a mean-squared error sense).

For architecture RNN2, with $n_h = 4$ and $n_i = 3$ (includes a bias term), networks were trained from state-space initialisations and the resulting learning curves compared with those for training from random initial weights. The best and worst curves obtained from 20 repeated trials are shown in Fig. 4.6(c). All learning rates were the same. The learning curves show that, in general, the variation in training performance was less for networks initialised from state-space models, compared to networks initialised with random weights.

In summary, the learning curve comparisons show that more constrained initial estimates for network weights can reduce the variation in training performance. Due to convergence to local minima, state-space models do not offer as reliable a source of initial weight estimates as those obtained from ARX models, for which the parameters can be found in closed-form. None of the recurrent architectures considered so far allow complete initialisation from an ARX model. In the next section, a modified recurrent architecture (RNN3) is proposed, which permits initialisation from a partial fraction expansion of an ARX model.

4.4 Modified Recurrent Network Architecture

The RNN3 architecture is based on the partial fraction expansion of the ARX model transfer function. In the linear case, this architecture resembles the parallel formant



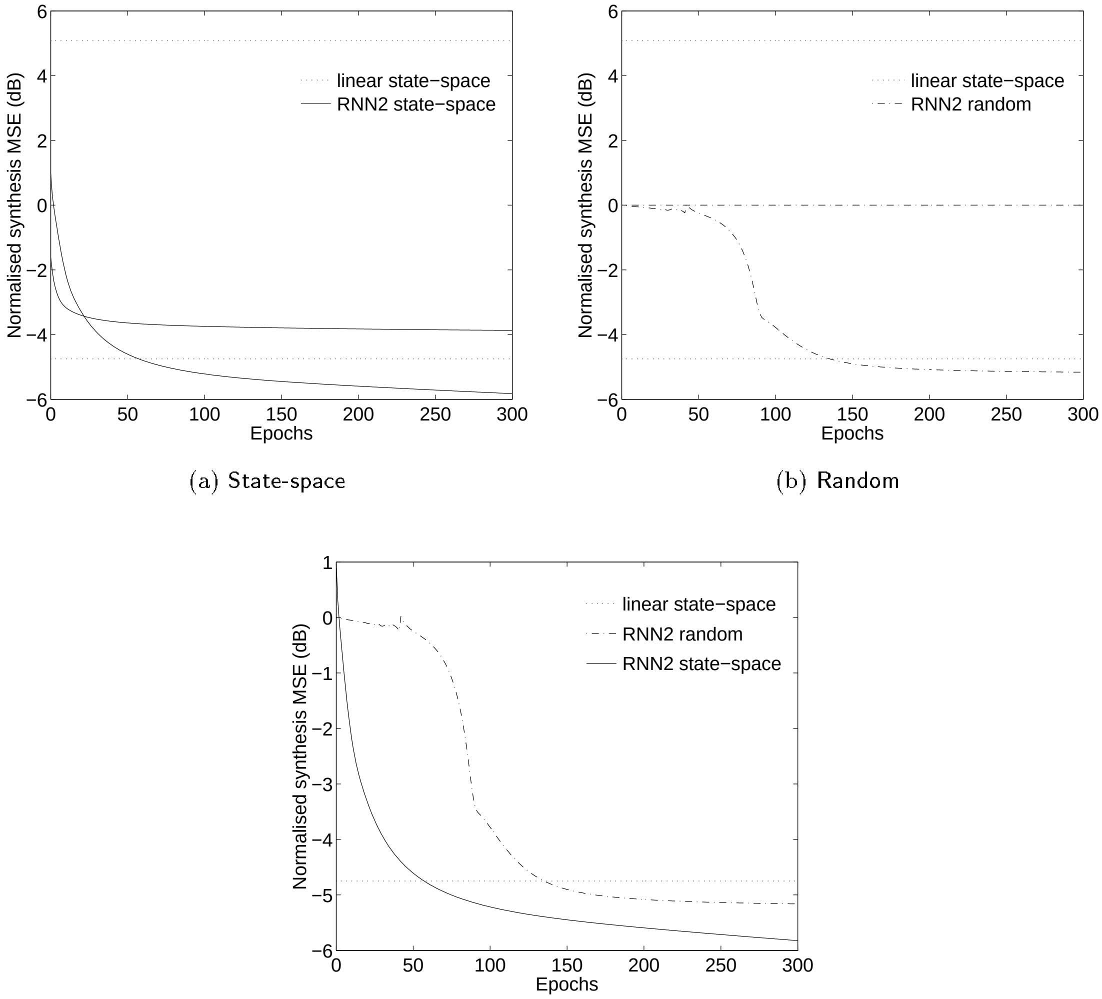
(a) State-space initialisations

(b) Random initialisations

(c) ARX initialisation with random W ($n_a = 10$,$n_b = 2$, plus bias term)

(d) Best of 20 trials

Figure 4.5: Comparison of initialisation techniques for RNN1 with $n_h = 2, n_i = 13$ (includes bias term). In a), b), and c), the best and worst curves from 20 trials are shown. In d), the best curves for each method are compared.



(a) State-space

(b) Random

(c) Best of 20 trials

Figure 4.6: Comparison of initialisation techniques for RNN2 with $n_h = 4$,$n_i = 3$ (includes bias term). In a) and b), the best and worst curves from 20 trials are shown. In c), the best curves for each method are compared.

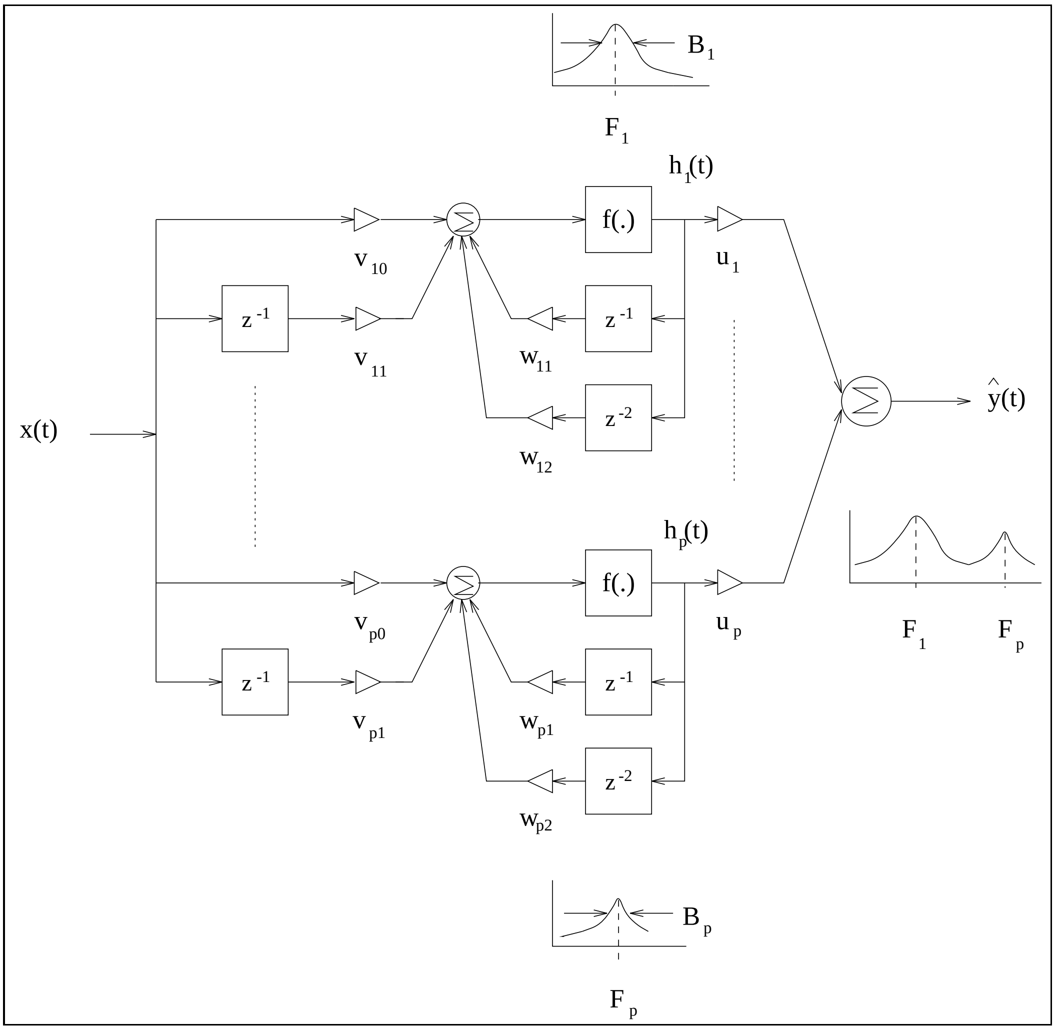


Figure 4.7: Structure of modified recurrent network.

model of Fig. 2.2. The structure of this architecture is shown in Fig. 4.7, in which there are feedback delays of one and two samples around a single layer of hidden nodes with nonlinearities $f(\sigma) = \tanh(\sigma)$. There are no cross-connections between hidden nodes and the single output node is linear. The network output, $\hat{y}(t)$, is given by Eqn. (4.47)

$$\begin{aligned} \hat{y}(t) &= \sum_{i=1}^{p} u_i h_i(t) && (4.47) \\ h_i(t) &= f(\sum_{j=0}^{1} v_{ij} x(t-j) + \sum_{k=1}^{2} w_{ik} h_i(t-k)) && (4.48) \end{aligned}$$

where p is the number of hidden nodes and $h_i(t)$ is the output of hidden node i.

4.4.1 Weight Initialisation from ARX model

For initialisation of architecture RNN3, the equivalence with a partial fraction expansion of the transfer function, $H(z)$, of a linear ARX model is used. The poles and residues of the expansion are combined in complex-conjugate and real pairs to give p quadratic factors, $H_i(z)$, from which the network weights are initialised.

$$\begin{aligned} H(z) &= \frac{b_0 + b_1 z^{-1} + \ldots + b_{n_b-1} z^{-n_b+1}}{1 + a_1 z^{-1} + a_2 z^{-2} + \ldots + a_{n_a} z^{-n_a}} && (4.49) \\ H(z) &= \sum_{i=1}^{p} u_i H_i(z) && (4.50) \\ H_i(z) &= \frac{v_{i0} + v_{i1} z^{-1}}{1 - w_{i1} z^{-1} - w_{i2} z^{-2}} \qquad i = 1 \ldots p && (4.51) \end{aligned}$$

All output weights, u_i, were initially set to 1, thus assuming an equal contribution from each formant in Eqn. (4.50). The decoupled hidden nodes allow the weights u_i, $\boldsymbol{v}_i$, for each hidden node to be scaled independently, to shift the operation of each hidden unit into the linear region. A constant scale factor, k, was used such that $k\boldsymbol{v}_i\boldsymbol{x}(t) < D_2$, for $t = 1 \ldots N$. D_2 was a chosen limit on the linear region of operation of $f(.)$, as defined in Fig. 4.2. The corresponding u_i were scaled to u_i/k so that $H_i(z)$ were unchanged. The order, n_a, of the ARX model is given by $n_a = 2p$, where the number of hidden nodes equals the number of formants of voiced speech, typically 4 or 5 at 8kHz. n_b was chosen to satisfy $1 \leq n_b \leq n_a - 1$. This avoids direct terms in the partial fraction expansion. Zeros in $H(z)$ are desirable when modelling nasals and are also needed to model the effect of lip radiation when the excitation is $x(t)$. Fig. 4.8 shows an example of the spectra of the formant factors for an ARX model with $n_a = 8$ and $n_b = 1$.

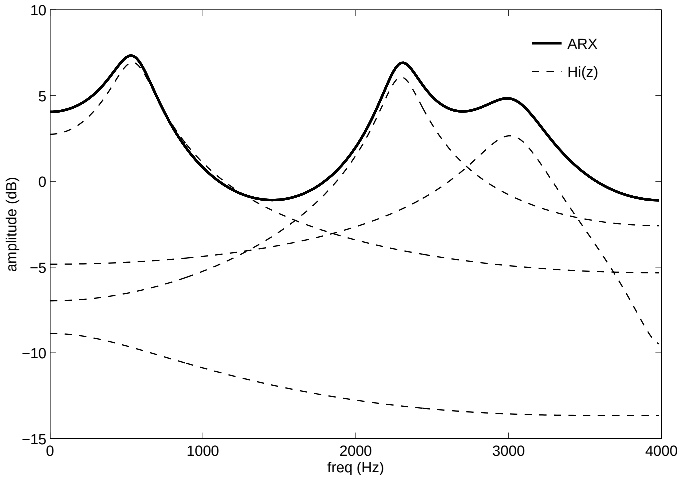


Figure 4.8: Spectra of ARX model and $H_i(z)$ $(n_a = 8, n_b = 1)$ for initialisation of RNN3.

4.4.2 Training by back-propagation

In Appendix A, the Real Time Recurrent Learning algorithm is derived for the modified recurrent architecture. This learning algorithm is similar to that of Ku & Lee (1995), for diagonal, single delay recurrent networks. The weights were updated stochastically after the presentation of every sample, calculating the update and updating first $\boldsymbol{U}$, then $\boldsymbol{W}$, then $\boldsymbol{V}$. An example learning curve is shown in Fig. 4.9(c).

4.5 Nonlinear Vocal Tract Modelling Using Neural Networks

So far in this chapter, modelling of short segments of speech by a single network of varying architecture has been considered. For modelling long utterances, the framework of Fig. 3.16 was used, in which the linear filter, $H(q)$, was replaced by a nonlinear neural network. Preprocessing of data and generation of the glottal volume velocity waveform was performed as in chapter 3. Input data samples in the network input vector, $\boldsymbol{x}(t)$, were supplied by the first difference of the glottal volume velocity waveform, $dx(t)$. For unvoiced frames, the codebook entry that minimised the mean-squared synthesis error at the output of the untrained network, was selected and used in training. This was computationally less expensive than training a network for each codebook entry and the excitation can be re-optimised in an analysis-by-synthesis approach, once training of the networks is completed. Two architectures were chosen for the vocal tract model, the feed-forward architecture (FNN) and the modified recurrent architecture, RNN3. The reasons for this selection, and the trade-off that this choice represents, are described in the following section.

4.5.1 Selecting a Network Architecture

The requirements of a suitable architecture are: stable synthesis; an initialisation from a closed-form ARX model; as many hidden nodes as possible (within the limit imposed on n_θ by the size of the analysis window). In selecting a network architecture for the vocal tract modelling problem, the following points were taken into consideration:

FNN The feed-forward architecture described in section 4.3.3 gives an equation error model[4]. The parameters determined by back-propagation are therefore not optimal for synthesis. In synthesis, the delayed output samples, $[y(t-1) \ldots y(t-n_a)]$, in the input vector are replaced by the delayed network outputs, $[\hat{y}(t-1) \ldots \hat{y}(t-n_a)]$. Since the network is not trained in this mode, the synthesis is not necessarily good and may not be stable. A lock-up situation can occur, in which all hidden nodes saturate. Feed-forward networks define a static mapping between the input and output of the network. The current output is independent of previous inputs and outputs and contextual effects can only be captured by using a fixed number of delayed input and output samples in $\boldsymbol{x}(t)$. The dimension of the input vector is therefore quite high and the total number of network weights is large unless a small number of hidden units are used.

Recurrent networks with single delays around hidden units can account for the contextual effects of previous system inputs and outputs on the current system output without explicitly including delayed system inputs in the input vector. The recurrent network defines a dynamic mapping in which the current network output depends on previous network inputs. The duration of this contextual dependence is not explicitly defined as in the feed-forward case. In theory, it can extend back to the beginning of the utterance, but in practice it is limited by the relative magnitudes of the network weights $\boldsymbol{V}$ and $\boldsymbol{W}$.

RNN1 For this architecture, the input vector is the same as that used for FNN . To keep the number of network parameters within limits, the only realistic number of hidden units for the vocal tract modelling task is $n_h = 2$. This limits the nonlinear capabilities of the model. Back-propagation training minimises the mean-squared prediction error, so the parameters are not optimal for synthesis. As for the feed-forward network, the network synthesis may not be stable. Initialisation of this architecture from a state-space model is possible, but the effectiveness of this initialisation is variable. Initialisation from an ARX model requires random initial values for $\boldsymbol{W}$.

RNN2 is analogous to an output error model, in the sense that training by back-propagation minimises the synthesis (output) error. Provided training of the network remains stable, a stable synthesis is obtained. This is an advantage over the feed-forward architecture and RNN1. The reduced dimension of the input vector also permits

[4] For linear functions at the hidden units, the network gives an ARX model of the vocal tract system.

architectures with up to $n_h = 4$. Unfortunately, RNN2 cannot be compared directly to a linear ARX or OE model and no initialisation from an ARX model is possible. As for RNN1, state-space initialisations were not reliable.

RNN3 is the only recurrent architecture that satisfies all the requirements. The reduced dimension of the input vector and the absence of cross-connections between hidden nodes allows up to 5 hidden nodes to be used and gives reduced training times when compared with the fully connected architectures, RNN1 and RNN2. The synthesis error is minimised directly by back-propagation training which means that models which give an unstable synthesis can be discarded at the training stage. With linear functions at the hidden units, RNN3 is analogous to a parallel formant implementation of the vocal tract model.

The feed-forward architecture and recurrent architecture RNN3 were finally selected because they can be initialised from ARX models and allow comparison of parallel versus series implementation, output error versus equation error minimisation, and static versus dynamic mapping in a nonlinear model.

4.5.2 Size of Networks and Initial ARX models

In the sequential processing of long utterances, each network model is generated from a short window of data. In order that the analysis window contain sufficient information for accurate estimates of the network parameters, a window length of 30ms (240 samples at 8kHz) was used at a frame rate of 10ms. This places an upper limit of about 24 on the total number of network parameters, n_θ. The trained networks were used to synthesise speech over the duration of the frame rate only (10ms), as illustrated in Fig. 3.17.

Networks were initialised from stable ARX models, for both voiced and unvoiced data. Scaling of the linear initialisations was such that the outputs of all hidden nodes was less than D_2, Fig. 4.2. Because this limit constrains the operation of hidden nodes to the 'linear' region of $f(\sigma)$, the initial weights give stable networks. The method of initialisation, size of networks and order of initial ARX models are summarised in Table 4.1.

For an input vector containing samples of $dx(t)$, an ARX model with $n_b = 2$ is sufficient to model the vocal tract (and includes a single zero) because the lip radiation effects are lumped in with the excitation. For architecture RNN3, the number of hidden units is determined by the expected number of formants at 8kHz (4-5), and the initialisation procedure fixes the number of network inputs to $n_i = 2$. However, any value of n_b can be used for the initial ARX model, provided $n_b <= n_a - 1$. Slight alteration of the parameters of the linear partial fraction expansion may result in a model for which the order of the numerator is as high as $n_b = n_a - 1$, because some zeros may no longer cancel out. In training, adjustment of the weights can cause this effect, so that the transfer function for the linearised network is no longer of the same order n_b as the initial ARX model. For comparison purposes, initialisation from an ARX model of order $n_b = n_a - 1$, was

	FNN	**RNN3**
Analysis Window	30ms	30ms
Frame Rate	10ms	10ms
n_i	$n_a + n_b$, no bias	2, no bias
n_h	2	5
n_o	1	1
n_θ	26	25
Input $\boldsymbol{x}(t)$	$[y(t-1)\ldots y(t-n_a)\; dx(t)\ldots dx(t-n_b+1)]^{\mathrm{T}}$	$[dx(t)\ldots dx(t-n_b+1)]^{\mathrm{T}}$
Initialisation	ARX-QR	parallel ARX
Initial ARX Order	$n_a = 10$, $n_b = 2$	$n_a = 10$, $n_b = 2$ or 9
D_2	0.2	0.13
Learning Rate (η)	$\eta_u = \eta_v = 10^{-2}$	$\eta_u = \eta_v = 10^{-3}$, $\eta_w = \eta_u/10$

Table 4.1: Summary of parameter values for vocal tract modelling with neural networks.

also considered. This value for n_b is the maximum possible order of a linearised network transfer function.

To minimise the effect of the memory in the hidden nodes from frame to frame, it was important to ensure that the allocation of the formant factors, $H_i(z)$, resulted in the smallest possible change in the values of the network weights from frame to frame. For the case when more than 2 real poles occurred in the partial fraction expansion, it was also necessary to consider the optimum pairing of the real poles. The optimisation was performed sequentially by matching the poles of the initial ARX model for the current frame to the poles of the (linearised) trained network for the previous frame. A simpler method which proved equally effective was to allocate the formant factors to hidden nodes $1 \ldots n_h$ in increasing order of formant frequency. This is equivalent to ensuring that each hidden node at initialisation represents a similar formant frequency from frame to frame.

For the feed-forward architecture, the number of input parameters is given by $n_i = n_a + n_b$, and is determined by the size of the ARX model used for initialisation. To keep the total number of network weights low, model dimensions of $n_h = 2$, $n_b = 2$ and $n_a = 10$ were used, giving a total number of network parameters similar to that for the recurrent architecture. The networks were initialised by the ARX-QR method of section 4.3.3, because $n_h << n_i$. For the feed-forward architecture, all permutations (over hidden nodes) of the weight initialisation result in the same synthesis because there is no internal feedback and an ordered allocation of weights to hidden nodes is not necessary.

The sign of the initial output weights, $\boldsymbol{U}_0$, was adjusted to match that of the corresponding weight (after training) for the previous frame. The sign of $\boldsymbol{V}_0$ was adjusted accordingly so that the initial transfer functions, $H_i(z)$, were unaltered.

4.5.3 Issues for Back-propagation

Feed-forward networks were trained by the back-propagation algorithm (Rumelhart, Hinton & Williams 1986) to minimise the mean-squared prediction error. Weights were updated stochastically after the presentation of each speech sample in the analysis window. Due to the stochastic update, the learning curves may not be monotonically decreasing. Two sets of optimal weights were recorded, those which gave the minimum mean-squared prediction error, and those which gave the minimum mean-squared synthesis error, over all training epochs.

Recurrent networks were trained to minimise the mean-squared synthesis error using the extended back-propagation algorithm of Appendix A. The weights were updated stochastically after the presentation of each sample in the analysis window and the final network weights recorded were those which gave the minimum mean-squared synthesis error over all training epochs.

All networks used for the vocal tract modelling task were small and good solutions in which hidden nodes saturate are therefore unlikely. If nodes do saturate during training, they are likely to take a long time to recover because $f'(.)$, the derivative of $f(.)$, approaches zero. The flat spots in the derivative were eliminated by adding a constant 0.1 to $f'(.)$, as proposed by Fahlman (1988). This was found to be beneficial in speeding up the training procedure for both feed-forward and recurrent networks[5].

In an attempt to smooth parameter transitions from frame-to-frame, a limited number of epochs of training, using small learning rates, were performed for each network. Limiting the total number of epochs of training therefore limits the total change in the weight values in training, but also limits the ability of the back-propagation procedure to converge to the global minimum. Work on regularization by Sjöberg & Ljung (1992) has shown that this early stopping in the minimisation of an error function, with respect to some parameters has the same effect as regularization towards the initial estimates of those parameters. Hence network solutions with weights in the vicinity of the linear initialisation will result.

4.6 Performance of Network Models

Initial estimates of suitable values of learning rate and scale factor for hidden units, D_2, were made based on the performance of the networks on short segments of voiced data. In order to maintain the stability of the recurrent network architecture, small learning rates for feedback weights were needed. In Table 4.1, the final selection of learning rates and scale factors for FNN and RNN3 are given, where η_u, η_v and η_w, are the learning rates for weights $\boldsymbol{U}$, $\boldsymbol{V}$ and $\boldsymbol{W}$ respectively (fixed for all elements of the matrices).

On individual frames of data, the feed-forward network gave improvements in prediction gain (SNR) of $2 - 3$dB over the linear initialisation model. These results are in the

[5] The elimination of flat spots in $f'(.)$ is further enhanced by the addition of a regularizing term to the mean-squared error criterion (see section 4.7.1).

region of results reported by other researchers for the improvement of nonlinear predictors on speech data (Tishby 1990, Townshend 1991, Wu & Fallside 1992). The improvements in synthesis SNR obtained by recurrent networks over initial ARX values were of comparable magnitude. However, when averaged over all frames of an utterance, the improvements in prediction and synthesis SNR obtained from networks over initial linear models were much lower. This was due to using the same, fixed learning rate for all frames of data. Using the same value of learning rate for all frames limited the performance because smooth learning curves were not obtained for the training of all networks. Often training did not improve on the synthesis SNR obtained from the initial weight values. In particular, this was the case for feed-forward networks, because weights are not trained to minimise the mean-squared synthesis error. Minimisation of the mean-squared prediction error does not necessarily simultaneously minimise the mean-squared synthesis error, as illustrated by comparing the prediction and synthesis MSE in Figs. 4.9(a) and 4.9(b). Using a very small learning rate for all frames increased the average number of frames for which training was successful, however the increase in SNR over linear models from such training was small. The slow convergence of the back-propagation algorithm and a limited number of epochs of training for each network also contributed to the low average performance.

The performance of network models was particularly poor on unvoiced frames of data. This was due to the poor performance of the initial ARX models and the fact that network weights trained slowly from their initial values.

The following section describes how learning rate adaptation and regularization were used to improve the performance of back-propagation training and to enforce smooth parameter transitions from frame to frame.

4.7 Improving The Perceptual Quality of Network Synthesis

To improve the synthesis from neural network models, it was necessary to improve the average performance over all frames of an utterance. This was achieved by increasing the number of networks that train successfully (by choosing a more suitable learning rate at each frame) and by improving the convergence of simple back-propagation training. Methods for automatically adjusting learning rates and for improving the convergence of back-propagation have been extensively reviewed by Schiffmann, Joost & Werner (1992). Of the speed-up methods reviewed, the learning rate adaptation scheme proposed by Silva & Almeida (1990) was most successful, and was tested for this application. Initial estimates of suitable values for the learning rate scale factor were made using segments of vowel data, and the values finally selected were 1.7 and 2, for FNN and RNN3 respectively. Maximum and minimum limits were imposed on the learning rates to prevent underflow and to ensure that the maximum values did not become too high. Initial values of the learning rate were set to η_u, η_v and η_w. Examples of the learning curves obtained for training architectures FNN and RNN3, using learning rate adaptation, are shown in Fig. 4.9. These graphs show

	FNN	**RNN3** ($n_b = 2$)	**RNN3** ($n_b = 9$)
Prediction SNR (dB)	12.0 (14.7,7.5)	-	-
Mean Improvement (dB)	0.23 (0.24,0.23)	-	-
Max. Improvement (dB)	6.6 (6.6,2.1)	-	-
Synthesis SNR (dB) (voiced,unvoiced)	5.2 (6.5,1.4)	5.0 (6.4,1.6)	5.2 (6.6,1.7)
Mean Improvement (dB)	1.0 (1.2,0.4)	0.8 (1.1,0.2)	1.0 (1.4,0.3)
Max. Improvement (dB)	12.6 (12.6,1.8)	12.1 (12.1,1.4)	13.9 (13.9,1.8)
Last Frame Init. (%)	79	31	35
No Improvement in Synthesis(%)	17	26	25

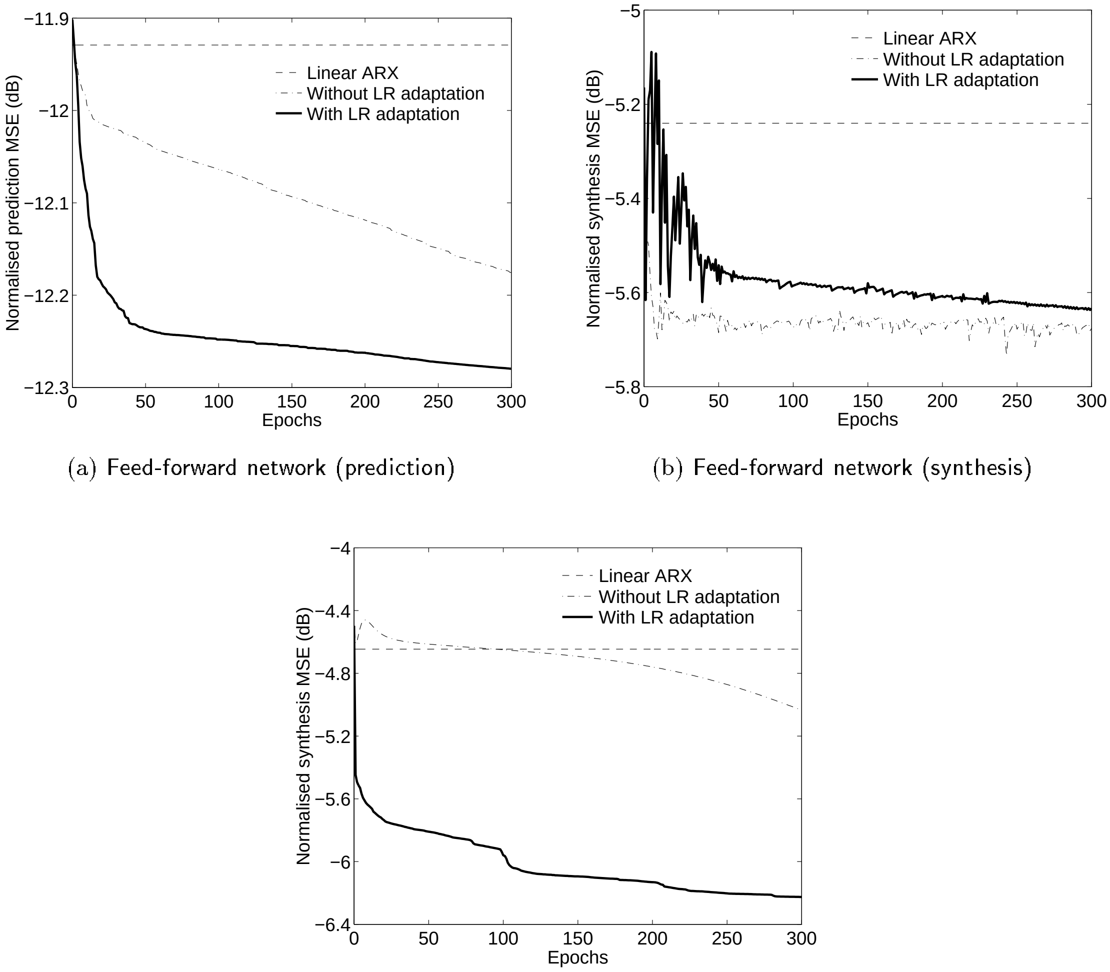
Table 4.2: Performance of networks trained with learning rate adaptation for the utterance "Germany's decision followed eight years later". Performance figures show synthesis SNR averaged over all frames (voiced, unvoiced).

that learning rate adaptation successfully improved the convergence of back-propagation training for both architectures. The overall effect of learning rate adaptation was to increase the average number of frames for which networks were successfully trained.

For unvoiced speech, linear models performed poorly and did not, in general, provide a good initialisation for network weights. Because speech is produced by a slowly time-varying mechanism, the statistical properties of short-time frames of the speech signal vary slowly from one frame to the next, and the parameters of models for adjacent frames are similar. Thus, final values of network weights from the previous frame may provide a better initial estimate for the weights of the current frame than a linear model. Even for voiced speech, this may also be the case, especially with improved convergence of back-propagation training, where it is likely that the values of weights will change more from their initial values. To allow for this possibility, networks were initialised both from the weights of the network from the previous frame and from a linear model. The initialisation giving the best synthesis SNR was retained.

For the utterance "Germany's decision followed eight years later", the average performance of networks trained with learning rate adaptation is shown in Table 4.2. The values for '% Last Frame' give the percentage of frames for which the weights were initialised from those of the network from the previous frame. The values for '% No Improvement in Synthesis' give the percentage of frames for which the network did not improve on the synthesis SNR of the linear initialisation model.

The high percentage of last frame weights used in initialisation of feed-forward networks suggests that training of these networks is more successful. The average synthesis SNR for feed-forward and recurrent architectures shows comparable performance. The maximum improvement obtained from RNN3 initialised from $n_b = 9$ is greater than that for the feed-forward architecture. This is to be expected, since the initial linear model is of higher order and the weights of architecture RNN3 are optimised for synthesis. The lower maximum improvement (in synthesis SNR) of architecture RNN3 initialised from



(a) Feed-forward network (prediction)

(b) Feed-forward network (synthesis)

(c) Recurrent network (synthesis)

Figure 4.9: Effect of learning rate adaptation on back-propagation training of feed-forward and recurrent networks on a fragment of vowel data.

$n_b = 2$, compared to the feed-forward architecture is surprising since the weights of the feed-forward architecture are not optimised for synthesis, unlike those of the recurrent architecture. This may reflect that $n_b = 2$ gives an unsuitable linear initialisation of the recurrent network. Although the average synthesis SNR for architectures FNN and RNN3 are comparable, the resulting synthetic speech from architecture RNN3 was perceptually noisier and slight background waterfall effects, similar to those obtained with the linear OE models, were detectable. The synthesis from feed-forward networks was less noisy, but also suffered from slight waterfall effects. For linear models, these effects were associated with excessive temporal variability in the spectrum of the synthetic speech. For architecture RNN3, these effects were less pronounced for the lower order initial ARX model, $n_b = 2$, which suggests that the effects of the filter memory (on unvoiced frames) becomes more important as n_b is increased.

With the improved learning made possible by learning rate adaptation, better nonlinear models were developed, but the resulting parameter transitions were not as smooth. Using a small learning rate and a limited number of epochs of training was no longer sufficient to ensure smooth weight transitions at frame boundaries and an explicit regularization term in the mean-squared error criterion was needed to constrain the change in weight values from frame to frame.

4.7.1 Back-propagation with Regularization

Smooth weight transitions from frame to frame were enforced by addition of a regularizing term to the mean-squared error criterion, which penalises large changes in value of network weights, $\boldsymbol{\theta} = \{\boldsymbol{U}, \boldsymbol{V}, \boldsymbol{W}\}$ from their initial values $\boldsymbol{\theta}^0 = \{\boldsymbol{U}^0, \boldsymbol{V}^0, \boldsymbol{W}^0\}$, to give the regularized error criterion, E_R

$$E_R = E + \frac{\alpha}{2}||\boldsymbol{\theta}^i - \boldsymbol{\theta}^0||^2 \tag{4.52}$$

where $||\boldsymbol{\theta}^i - \boldsymbol{\theta}^0||^2 = ||\boldsymbol{U}^i - \boldsymbol{U}^0||^2 + ||\boldsymbol{V}^i - \boldsymbol{V}^0||^2 + ||\boldsymbol{W}^i - \boldsymbol{W}^0||^2$ and α is a constant that determines the weight movement that is tolerated. The new gradient and update are given by

$$\begin{aligned} \frac{dE_R}{d\boldsymbol{\theta}^i} &= \frac{dE}{d\boldsymbol{\theta}^i} + \alpha(\boldsymbol{\theta}^i - \boldsymbol{\theta}^0) && (4.53) \\ \Delta\boldsymbol{\theta}^i &= -\eta^i \frac{dE}{d\boldsymbol{\theta}^i} - \eta^i\alpha(\boldsymbol{\theta}^i - \boldsymbol{\theta}^0) && (4.54) \end{aligned}$$

In the above equations, the adaptable learning rate, η^i, is updated based on the change in sign of the local gradient of the regularized error criterion, $\frac{dE_R}{d\boldsymbol{\theta}^i}$.

In back-propagation training of feed-forward networks, minimisation of the mean-squared error is normally interpreted as equivalent to maximum likelihood estimation of network weights with a Gaussian error model (Neal 1995, Lim & Oppenheim 1978).

Addition of the regularizing term, $(\alpha/2)||\boldsymbol{\theta}^i - \boldsymbol{\theta}^0||^2$, to the mean-squared error criterion, is equivalent to deriving a MAP estimate of the network weights, assuming that the network weights have a Gaussian prior distribution with mean $\boldsymbol{\theta}^0$ and covariance matrix $\boldsymbol{I}/\alpha$, where $\boldsymbol{I}$ is the identity matrix (Neal 1995, Sjöberg 1995). The update term obtained using this regularizing term differs from that for weight decay[6] by the additional term, $\eta^i\alpha\boldsymbol{\theta}^0$. Sjöberg (1995) has shown that the additional term allows training to continue when hidden units saturate. The effect of $\eta^i\alpha\boldsymbol{\theta}^0$ is analogous to the addition of a constant to $f'(.)$, which was proposed by Fahlman (1988) as a means to speed up training. The effect of this term can also be interpreted as an accumulated momentum term. Typically, a momentum term, $\beta\Delta\boldsymbol{\theta}^{i-1}$, is added to the mean-squared error criterion to speed training. The momentum term modifies the direction of the current weight update by taking the previous weight update into consideration

$$\Delta\boldsymbol{\theta}^i = -\eta^i \frac{dE}{d\boldsymbol{\theta}^i} + \beta\Delta\boldsymbol{\theta}^{i-1} \tag{4.55}$$

Using Eqn. (A.3) to rewrite $\Delta\boldsymbol{\theta}^{i-1}$, and taking the initial update to be $\boldsymbol{\theta}^1 = \boldsymbol{\theta}^0 + \Delta\boldsymbol{\theta}^0$ gives

$$\boldsymbol{\theta}^i = \sum_{k=0}^{i-1} \Delta\boldsymbol{\theta}^k + \boldsymbol{\theta}^0 \tag{4.56}$$

Replacing $\boldsymbol{\theta}^i$ in Eqn. (4.54) gives

$$\Delta\boldsymbol{\theta}^i = -\eta^i \frac{dE}{d\boldsymbol{\theta}^i} - \eta^i\alpha \sum_{k=0}^{i-1} \Delta\boldsymbol{\theta}^k \tag{4.57}$$

Comparison with Eqn. (4.55) shows that the regularization term can be interpreted in terms of momentum. The direction of the parameter update is modified according to the accumulated updates, rather than the previous update only.

4.7.2 Improved Performance Results

Table 4.3 shows the performance of models in synthesis, when trained with regularization and learning rate adaptation on the utterance "Germany's decision followed eight years later". The figures give the mean synthesis SNR (without the additional regularizing term) for the syntheses obtained from models trained using the regularized error criterion. For this reason, the performance of the models appears inferior to that from models trained without regularization.

Since the motivation for regularization is a perceptual one, α was selected based on best perceptual performance and a value of $\alpha = 5$ was found to be suitable for both feed-

[6] In weight decay, the regularizing term, $||\boldsymbol{\theta}^i||^2$, is used, resulting in a weight update, $\Delta\boldsymbol{\theta}^i = -\eta^i dE/d\boldsymbol{\theta}^i - \eta^i\alpha\boldsymbol{\theta}^i$. The regularizing term penalises large weights and is usually used with large architectures to prevent overfitting.

	FNN	**RNN** ($n_b = 2$)	**RNN** ($n_b = 9$)
Synthesis SNR (dB)	5.2 (6.5,1.3)	4.5 (5.8,1.7)	4.8 (6.1,2.0)
Linear Synthesis SNR (dB)	4.3 (5.4,0.9)	3.8 (5.0,1.4)	4.2 (5.4,1.7)
Mean Improvement (dB)	0.98 (1.16,0.43)	0.65 (0.85,0.23)	0.57 (0.73,0.27)
Max. Improvement (dB)	12.5 (12.5,1.8)	6.4 (6.4,1.8)	7.2 (7.2,1.8)
Last Frame Init. (%)	47	27	27
No Improvement in Synthesis(%)	20	40	41

Table 4.3: Performance of regularized networks for the utterance "Germany's decision followed eight years later". Performance figures show synthesis SNR averaged over all frames (voiced, unvoiced).

forward and recurrent networks. Perceptually, the performance of regularized models was improved over that of unregularized models and the background underwater effects were eliminated. With regularization, higher order initial ARX models for architecture RNN3 gave better quality synthesis, although the average synthesis SNR was comparable to that from networks initialised by lower order ARX models. The syntheses from architecture RNN3 were still more noisy than those from the feed-forward architecture, and were worse than those from the parallel implementation of linear ARX models.

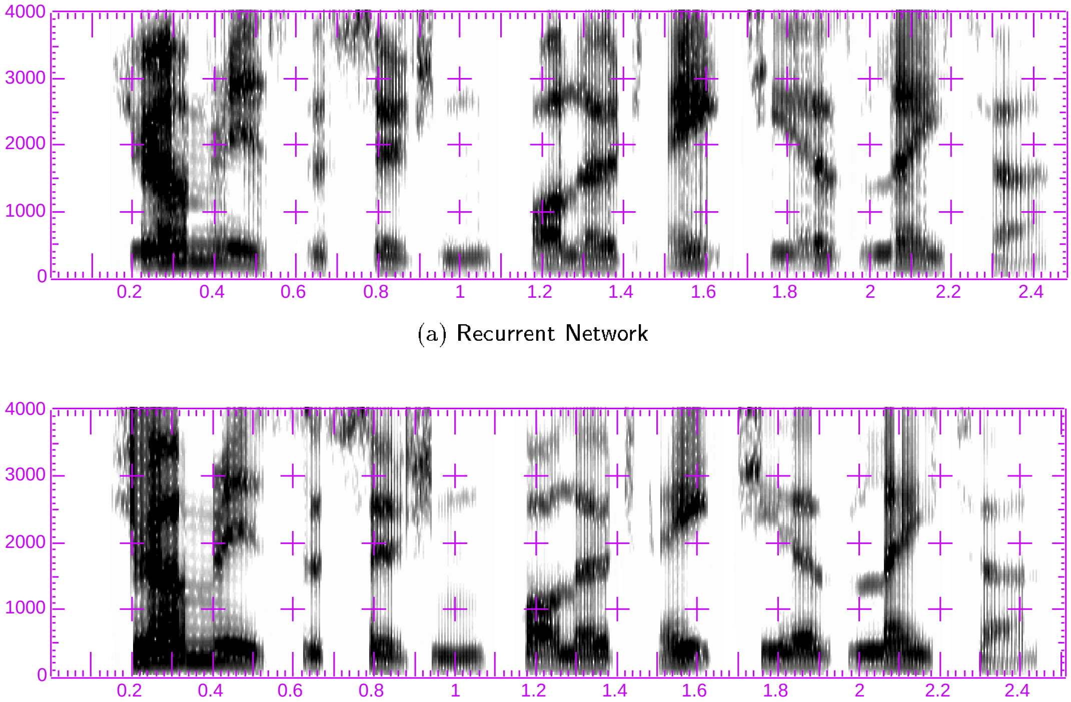
Fig. 4.10 shows spectrograms of the syntheses of the utterance "Germany's decision followed eight years later" by regularized neural networks. The spectrogram of the original utterance was shown in Fig. 3.23(a).

The tape demonstration includes examples of syntheses from feed-forward and recurrent neural networks to allow comparison of the performance of these models with black-box and linear prediction models.

4.8 Concluding Remarks

4.8.1 Summary

In this chapter, a nonlinear neural network based framework for speech synthesis was set up with the aim of overcoming the limitations of using linear models to capture the underlying dynamics of a nonlinear system. New methods for initialising the weights of feed-forward and recurrent architectures from linear models were developed, and a novel recurrent architecture was introduced which facilitates a linear initialisation of weights. The synthesis performance of this new architecture was compared with that of a feed-forward architecture. Feed-forward networks have the disadvantage that they define a static mapping between input and output and any correlation between the output and previous inputs and outputs must be explicitly modelled using an input window containing delayed samples of data. Recurrent networks provide a more compact representation of the system dynamics because correlations between the output and previous outputs and inputs is inherently incorporated into the internal state of the network. The average



(a) Recurrent Network

(b) Feed-forward Network

Figure 4.10: Spectrograms of syntheses of the utterance "Germany's decision followed eight years later". Networks were trained with learning rate adaptation and regularization. Horizontal axis shows time in seconds, vertical axis shows frequency in Hz.

synthesis SNR for recurrent and feed-forward networks was comparable but the synthesis from recurrent networks was audibly more noisy. This may be due to the parallel formant interpretation of the recurrent architecture. Regularization of the mean-squared error criterion improved the quality of the synthetic speech but the perceptual quality of syntheses from both architectures remained below that for linear ARX models.

4.8.2 Stability of Neural Network Models

Back-propagation training of recurrent networks minimises the output error (synthesis error) directly, thus a stable synthesis model is obtained provided training remains stable. For feed-forward networks, the parameters are estimated by equation error (prediction error) minimisation and are therefore not optimal for synthesis. Even when training remains stable, the resulting feed-forward network is not guaranteed to give a stable synthesis. This is potentially a problem as the degree of nonlinear operation of hidden units increases.

For linear models, stability constraints on parameters are well defined and models which produce an unstable (unbounded or oscillatory) synthesis can be stabilised by reflecting unstable poles back inside the unit circle. When calculating a linear initialisation for network weights, the weights were scaled to give approximately linear operation of all hidden units. Regularizing about the initial linear estimate of the weights biases the network toward a solution in which the operation of hidden units is localised in this region. While such operation persists, it was found that stability could be maintained by stabilising the equivalent linear model and re-mapping the stabilised parameters to network weights. Regularizing is thus beneficial for allowing stability of models in synthesis to be maintained.

With increased nonlinear operation of individual hidden units, stabilising the equivalent linearised network does not necessarily result in a stable nonlinear model. When nonlinear units operate close to saturation, the stabilising procedure may result in weight values which drive operation of all hidden units into saturation, thus causing saturation of the output of the network.

4.8.3 Usefulness of Linear Initialisation

For feed-forward networks, a linear initialisation assumes that a good linear ARX mapping between the input and output data exists. This is not the case when Φ_N, Eqn. (4.17), is ill-conditioned. The QR and SVD methods give rank n_h approximation to the best (in a mean-squared error sense) linear ARX model and are thus unsuitable for initialising small, single layer networks, for which $n_h << n_i$. The ARX-QR method was developed for such networks. Initialisations for networks with $n_i > n_h$ were not considered.

For recurrent architectures, new methods for initialisation of the network weights from state-space and output error models were considered. The advantage that initialising network weights from OE or state-space models offers over back-propagation training from random initialisations, is that second-order gradient descent techniques can be used to estimate the parameters of the initial linear model. Linear second order gradient techniques are less computationally intensive than the equivalent nonlinear procedure and give improved convergence over first-order techniques, such as back-propagation. Unfortunately, convergence to local minima may result in linear models which provide poor initial estimates for network weights.

Linear initialisation of network parameters is of possible use for small-scale system identification tasks, where an existing linear solution is quite good and it is expected that some nonlinearity will improve the performance. Omitted nonlinear dynamic terms in the estimated linear models can be detected by the application of correlation based model validity tests (Billings & Voon 1986).

4.8.4 Drawbacks of a Nonlinear Model

Despite their applicability to speech signals, nonlinear models in general have several drawbacks. The parameters are determined by iterative training techniques which are slow to converge and exhibit many local minima. The stability of the nonlinear system is undefined and limit cycles may occur. For nonlinear models, the principle of superposition no longer applies. There is no concept of a transfer function for the network, that is, no single function that describes the system's frequency response to any input. The additional frequency components can be accounted for by including higher order terms in a Taylor series expansion of the hidden node nonlinear function. This expansion is called a Volterra Series and can be used to set up generalised transfer functions for the nonlinear system (Priestley 1988). For small networks with hidden units operating in the approximately linear region of $f(.)$, the higher order terms may be small enough for a transfer function interpretation to approximate the frequency domain behaviour of the network. The lack of a transfer function means that the spectrum of synthetic speech cannot be calculated from the model parameters and the interpretation of the nonlinear model is difficult because it can no longer be related to the formant structure of the speech.

Part II

Classification of Speech Patterns

Chapter 5

Recurrent Networks for Context-Dependent Speech Classification

5.1 Introduction

Recurrent neural networks are widely used for context-dependent classification tasks such as identification of grammars (Giles, Miller, Chen, Chen, Sun & Lee 1992), phoneme recognition (Watrous & Shastri 1987, Etemad 1993, Hanes et al. 1994), large-scale speech recognition systems (Robinson & Fallside 1991) and hybrid connectionist-HMM speech recognition systems in which the recurrent network outputs are used to estimate posterior class probabilities for use with HMM models (Robinson 1994). Their popularity is due to the potential to exploit context in the classification decision by implicitly encoding past events within the network architecture. In this chapter, the way in which recurrent connections make use of previous context is investigated and the relevance of this to the performance of recurrent neural networks for context-dependent pattern classification tasks is discussed.

5.1.1 Context-dependent Pattern Classification Tasks

In this chapter, a context-dependent pattern classification task is regarded as one in which each class example is a finite labelled sequence, or trajectory, of correlated feature vectors. At a particular instant, the current feature vector is correlated with previous and future vectors of the trajectory (within-class context). At class boundaries, there may also be correlation between feature vectors from adjacent classes (between-class context). Training data consists of an uncorrelated set of class examples (no between-class context) or a correlated sequence of class examples (between-class context).

Acoustic speech data provides a rich source of context-dependent pattern classification tasks. Phone recognition from continuous speech utterances is a typical example, and exhibits both between-class and within-class context effects. As illustrated in Fig. 5.1, each

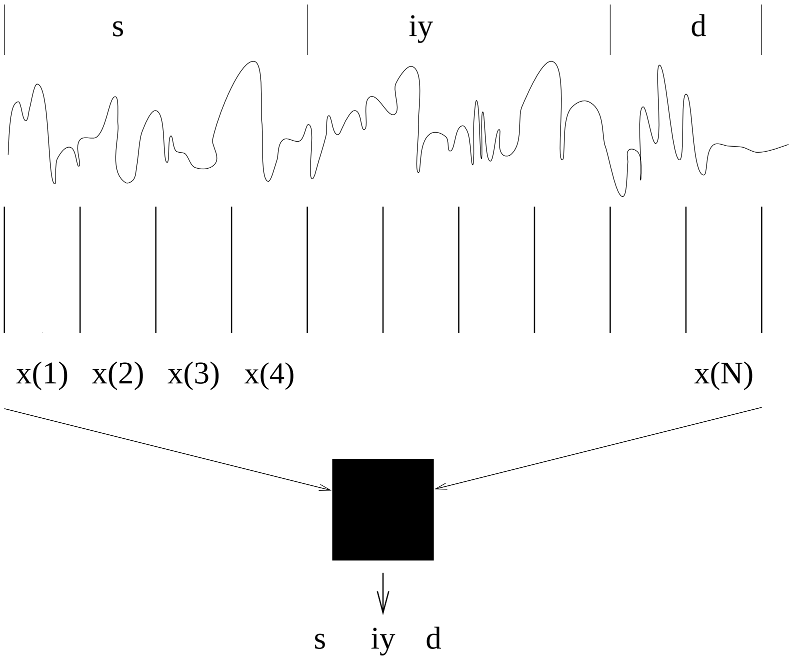


Figure 5.1: Phone recognition - a context-dependent classification task.

phone is characterised by one or more feature vectors which show noticeable correlation to the phone label. Feature vectors typically contain a spectral representation of the acoustic data, such as formant frequency contours (Hanes et al. 1994), channel energies from a filter bank spectrum analyser (Watrous & Shastri 1987) or mel-frequency cepstral coefficients derived form linear prediction (Niles 1991) or perceptual linear prediction analysis (Renals, Hochberg & Robinson 1994).

For each phone, there is an underlying configuration of the vocal tract required to produce that phone in isolation. In continuous speech, the physical constraints of the vocal tract means that each realisation of a phone is dependent on the previous and future phone to be uttered. There are therefore correlations both between feature vectors for the same phone (within-class context) and at phone boundaries, between feature vectors from adjacent phones (between-class context). In recognition, the task is to determine the sequence of phone labels from the sequence of feature vectors for the entire utterance. Human perception of speech relies quite strongly on the co-articulation effects between adjacent phones and the performance of automatic recognition systems is improved by efficient use of contextual effects. Recurrent neural networks offer one method for exploiting these effects. Several other methods which have been used in speech classification are reviewed in the following section and their advantages and disadvantages discussed.

5.1.2 Hidden Markov Models vs Neural Networks

Two popular approaches to the speech recognition problem are based on Hidden Markov Models and neural networks. The Hidden Markov Model (HMM) approach uses a statisti-

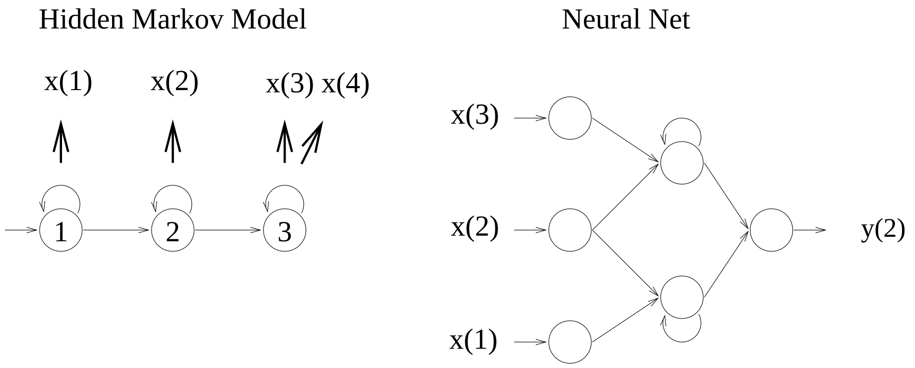


Figure 5.2: HMM and neural net approaches to phone recognition.

cal framework to fit models to the training data. For phone recognition, for example, each phone is represented by an HMM with one or more states and models are concatenated to form word models. Fig. 5.2 illustrates a three-state phone model. For each frame of an utterance, a state transition is made and an observation (feature vector) generated based on a probability density function associated with the new state. The parameters of the models are derived using a probabilistic training criterion, typically maximum likelihood. In recognition, a phone model is selected based on maximising the likelihood that the chosen model generated the observed sequence of feature vectors. A sequence of phone models that classifies the entire utterance is selected by maximising the likelihood that the chosen model sequence generates all feature vectors for the utterance. The disadvantage of the HMM approach is the underlying assumption that successive feature vectors (observations) are independent. For observations generated from the same state, for example vectors $\boldsymbol{x}(3)$ and $\boldsymbol{x}(4)$ in Fig. 5.2, the order in which the observations are produced has no effect on the overall likelihood of the model and there is no concept of within-class context for these observations. Limited within-class context can be accounted for by the state transition probabilities of models with more than one state. Phonetic (between-class) context is represented by the transition probabilities between models or can be modelled explicitly using diphone or triphone HMMs of all distinct phonetic contexts. The transition probabilities between models are given by a language model which describes the syntactic and phonological constraints on possible model sequences. These constraints do not however take account of the local acoustic context of a particular feature vector. Local acoustic context (for both within-class and between-class context effects) can be introduced by augmenting each feature vector to include first (and second) order differences of the vector elements calculated over several consecutive frames. Renals, Hochberg & Robinson (1994) use delta cepstral and delta energy parameters, for example.

Neural network based approaches to phone recognition use a neural network to perform a nonlinear mapping between a sequence of input feature vectors and a sequence of network outputs, as shown in Fig. 5.2. The network outputs encode the phone labels for the sequence of input feature vectors. Typically a 1-out-of-M encoding is used for M-class classification, in which the class targets are set (close) to ±1 for the correct class and zero for all others and remain constant for the duration of a phone. The network weights are trained to optimise classification of the training data by minimisation of the mean-squared error at the network outputs. This has the advantage over maximum likelihood training of HMMs that the training is discriminative and improves the output for the correct class while reducing the outputs for incorrect classes. The disadvantage of this method of training is the need to define suitable class targets.

Feed-forward networks approach the classification task by defining a static nonlinear mapping between input feature vectors and network outputs. The disadvantage of this architecture over HMM based models is that temporal variation in the feature vectors and the context in which they occur is not represented within the architecture. As illustrated in Fig. 5.2, past and future context can be introduced explicitly by replacing the single feature vector input by an input window containing several feature vectors from adjacent frames. Future context can also be provided by delaying the desired output decision for the central feature vector of the current input window.

Other approaches to incorporating context into neural network based recognition systems have used feed-forward networks as predictors to predict the current feature vector from an input window of previous feature vectors. A word recognition system was constructed by Iso & Watanabe (1990), in which each word in the training data was represented by a distinct sequence of feed-forward networks. Back-propagation training of the networks was embedded in a dynamic programming framework for correct time alignment of the sequence of network models for each example of the word in the training data. At a given instant, only the weights of the network selected by the optimum dynamic programming path were updated. A similar approach was used by Tebelskis & Waibel (1990) for building a word recognition system based on phone models comprised of three feed-forward prediction networks.

The limitation of these methods for introducing context into neural network models is that they only integrate the contextual effects for a finite duration of the input sequence. The depth of the context is fixed by the number of frames spanned by the input window. The networks fail to model dynamics of the feature vectors with a longer duration than that of the input window and cause smoothing of features that change rapidly within that duration. The depth of context is increased by increasing the size of the input window but this considerably increases the number of weights to be trained. A method for integrating context over longer time periods without substantially increasing the number of model parameters is required. Recurrent neural networks appear suited to this task.

5.1.3 Recurrent Networks for Context-Dependent Pattern Classification

Recurrent neural networks provide an efficient architecture for increasing the depth of contextual effects. They provide a more compact representation of the dynamics of a problem (in terms of number of network weights) than a feed-forward network with a finite duration input window because a shorter duration input window, or a single feature vector, can be used. Recurrent connections (delayed feedback paths) implicitly encode past context into the network architecture by introducing a memory of the past which is similar to that of an infinite impulse response filter. The hidden nodes operate like internal states to encode past events and the current network output is dependent on this internal state. In theory, the depth of past context potentially extends to the beginning of the training sequence and allows for the accumulation of contextual effects over many frames of data. In practice, the depth is determined by the relative size of the recurrent connection weights and can be estimated by a local linearisation of the hidden node nonlinear functions (Etemad 1993). The depth of past context is also influenced by the choice of training algorithm. In Back-Propagation Through Time (BPTT) (Werbos 1990), the entire training data sequence is divided into a number of windows and the network trained sequentially on each window. The learning algorithm is only capable of capturing features with dynamics which are represented within the current window and the duration of the window must therefore be long enough to capture the longest temporal dependency in the input sequence. Real Time Recurrent Learning (RTRL) (Williams & Zipser 1989*b*) does not suffer from this drawback. Although Williams & Zipser (1989*a*) have shown that RTRL is a more powerful learning algorithm, it has the disadvantages that it is more computationally intensive and requires more storage than BPTT. The computation and storage requirements are independent of the dynamics of the task but increase with network size. The following study does not consider the training algorithm itself, but focuses on the operation of the final network solution. Only RTRL was used in training network examples.

5.2 The Recurrent Network Decision Boundary

To determine how recurrent connections make use of context during the classification of time-varying patterns, the operation of a single hidden node with a unit delay recurrent connection, Fig. 5.3, is considered in this section. It is assumed that each class example is represented by a correlated sequence of feature vectors, $\boldsymbol{x}(t)$, and that sequences are concatenated to form one long time-varying pattern for classification. In training and recognition, the feature vectors are presented to the network one at a time (no input window), so that the only contextual effects are those due to the recurrent connection. The analysis focuses on two-class classification problems which permit the use of a single linear output node. The output weight, u, is set to 1 and it is assumed that networks are trained by back-propagation using class targets of -1 and $+1$, which remain constant for the duration of a class. In classification, a class decision is made for each feature vector of a sequence by an output threshold (bias weight). For the nonlinear function $f(\sigma) = \tanh(\sigma)$, the output threshold is assumed to be zero.

For the network of Fig. 5.3, the equations for the output of the hidden node, $h(t)$, and the network output, $y(t)$, are

$$h(t) \quad = \quad f\left(\boldsymbol{v}^{\mathrm{T}}\boldsymbol{x}(t) + wh(t-1) + \theta\right) \tag{5.1}$$

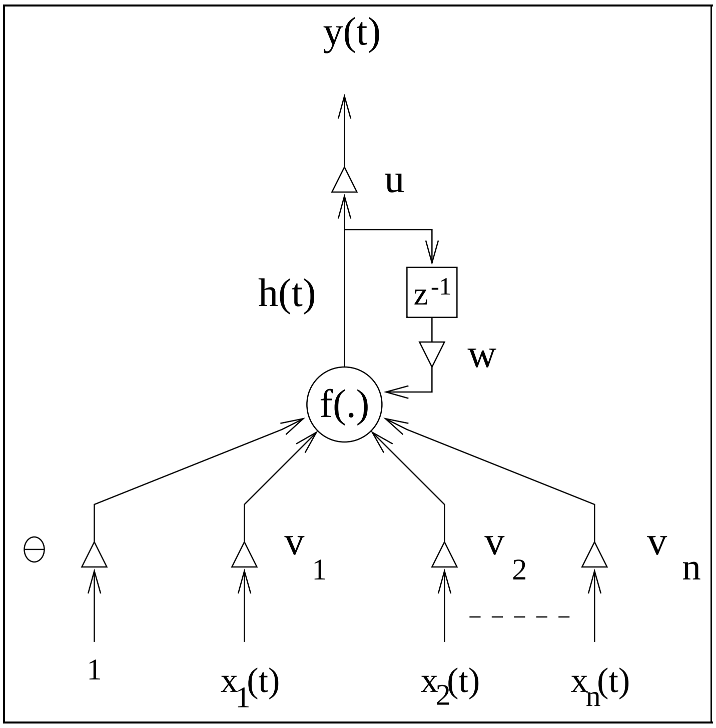


Figure 5.3: Single hidden node with recurrent connection.

$$y(t) \quad = \quad uh(t) \tag{5.2}$$

where $\boldsymbol{v}$ is a vector of input weights, $\boldsymbol{x}(t) = [x_1(t) \ldots \; x_n(t)]^{\mathrm{T}}$, θ is a bias term and $(.)^{\mathrm{T}}$ denotes transpose. The decision boundary is given by

$$\boldsymbol{v}^{\mathrm{T}}\boldsymbol{x}(t) + wh(t-1) + \theta = 0 \tag{5.3}$$

The contribution $\boldsymbol{v}^{\mathrm{T}}\boldsymbol{x}(t) + \theta$, represents a static linear decision boundary which can be interpreted as the decision boundary of an *extracted* feed-forward network which has the same weights u, $\boldsymbol{v}$ and θ. The term $wh(t-1)$ represents a variable bias term which causes the decision boundary to move in a direction parallel to $\boldsymbol{v}$. This is illustrated in Fig. 5.4 for a two-class problem in which $\boldsymbol{x}(t)$ is a two-dimensional feature vector and in training, fixed targets of -1 and $+1$ are used for class 0 and class 1 respectively.

The direction of decision boundary movement is determined by the sign of $wh(t-1)$. For the single hidden node of Fig. 5.3, the sign of $h(t-1)$ determines the previous class decision. Using a class target of $+1$ for class 1, $h(t-1)$ is positive when the previous class decision is class 1. Fig. 5.4 shows how the direction of decision boundary movement depends on the sign of w when $h(t-1)$ is positive. For positive w, the decision boundary moves away from class 1, the region of the feature space corresponding to the previous class decision. The region of the feature space for which feature vectors are classified as class 1 is increased, which biases the current decision towards that of the previous class decision. This is similar to the way in which the log prior ratio biases the decision function

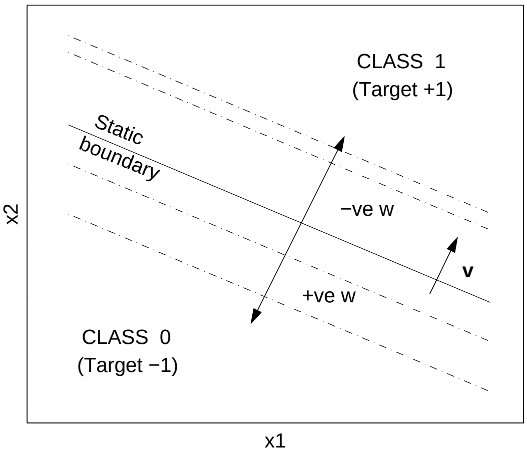


Figure 5.4: An illustration of a two-dimensional, two-class problem showing the effect of a recurrent connection, w, on the position of the decision boundary in feature space.

of a Bayes optimal classifier to favour the most probable class. For negative w, the decision boundary moves in the opposite direction and favours a change in class decision. With fixed class targets, training cannot converge to a stable solution with negative w. Positive w is therefore required and will be assumed in all further illustrations of the two-class problem introduced in Fig. 5.4.

The effect of the recurrent connection is thus to form a dynamic decision boundary which can change position in the feature space. Decision boundary movement potentially allows the current class decision to take account of the past context of a feature vector because the classification depends on the current position of the decision boundary. However, when saturation of the nonlinear function persists, movement of the decision boundary is inhibited. As discussed in the following sections, this impedes the ability of the network to make use of context in the classification procedure.

5.3 The Effect of Saturation - Context-Sensitivity

The effect of training the single hidden node of Fig. 5.3 with fixed class targets of ± 1 is that a sequence of feature vectors from the same class tends to drive the operation of the nonlinear function into saturation. Saturation causes maximum displacement of the decision boundary from the static position. For feedback w and class targets of ± 1, the positions of maximum displacement are at $\pm w$ (in the $\boldsymbol{v}$ direction) on either side of the static boundary, and divide the feature space into context-sensitive and context-insensitive regions. For the simple two-dimensional, two-class illustration considered previously, this is shown in Fig. 5.5, where the positions of maximum displacement are labelled class 0 and class 1 limit respectively.

For feature vectors from context-insensitive regions of the feature space, classification can be made without any prior knowledge of the sequence of feature vectors from which the

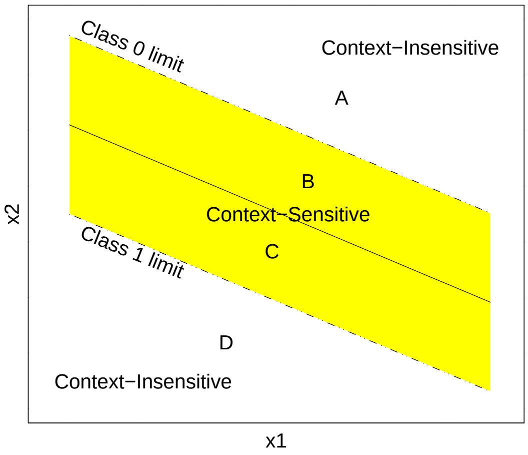


Figure 5.5: Effect of saturation on position of the decision boundary in the feature space.

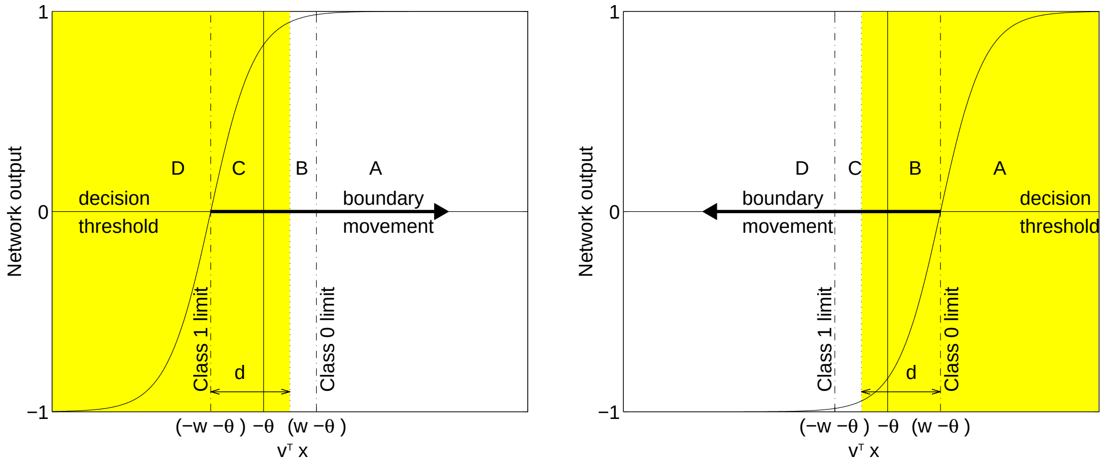
current example originates. Context-insensitive regions are those which are not spanned by movement of the decision boundary (A and D in Fig. 5.5). The classes assigned to these feature vectors are unaltered by a change in the position of the decision boundary (within the class limits) and remain the same when the feature vectors occur in a different context.

For feature vectors from context-sensitive regions of the feature space (B and C in Fig. 5.5), the class decision depends on the current position of the decision boundary which is determined by the history of previous class decisions. The class assigned to a feature vector from these regions is thus dependent on the past context in which the feature vector occurs.

5.4 The Effect of Decision Boundary Movement - Trajectory-Sensitivity

When saturation of the nonlinear function occurs, part of the context-sensitive region also becomes insensitive to local past context. This is because some of the context-sensitive feature vectors lie within the region of saturated operation of the hidden node nonlinearity and cannot initiate movement of the decision boundary. This is illustrated for the two-dimensional example in Fig. 5.6, which shows the feature space projections, $\boldsymbol{v}^{\mathrm{T}}\boldsymbol{x}(t)$, for which classification of a feature vector can cause movement of the decision boundary from the class limit (shaded regions). The figures also show the nonlinear (decision) function, $f(\sigma) = \tanh(\sigma)$, and the decision threshold.

Classification of a feature vector from region D changes the sign of feedback and gives large movement of the decision boundary. For feature vectors in regions B and C, movement of the decision boundary from a class limit only occurs when classification of a feature vector causes a significant change in the feedback from the saturation values of ± 1. This occurs for the region in which the nonlinear function operates significantly out of



(a) Movement from class 1 limit for $\boldsymbol{v}^{\mathrm{T}}\boldsymbol{x}(t) < -\theta - w + d$ (shown shaded)

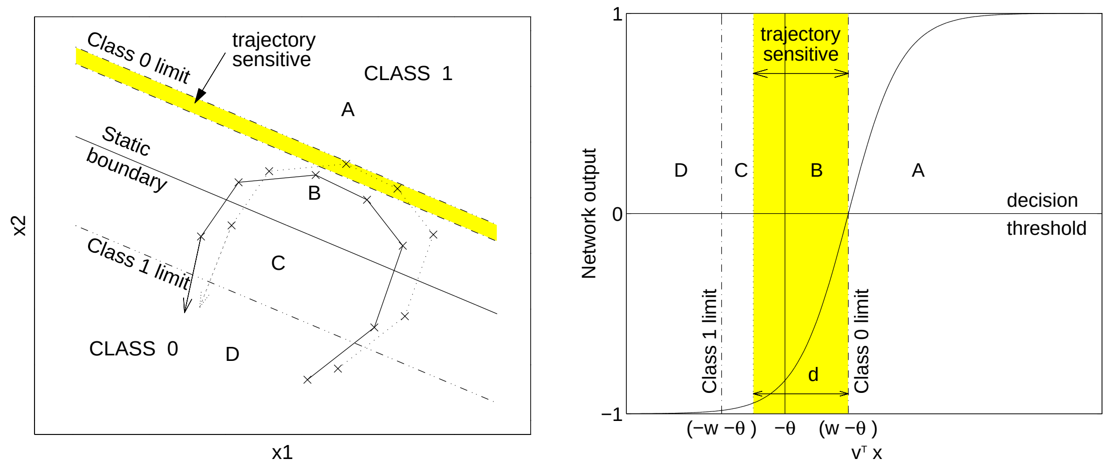
(b) Movement from class 0 limit for $\boldsymbol{v}^{\mathrm{T}}\boldsymbol{x}(t) > -\theta + w - d$ (shown shaded)

Figure 5.6: Feature space projections $(\boldsymbol{v}^{\mathrm{T}}\boldsymbol{x}(t))$ for which classification causes movement of the decision boundary (shown shaded). $2d$ defines the non-saturated region of $f(\sigma) = \tanh(\sigma)$.

saturation, shown by d in Fig. 5.6, where the width, or span, of the non-saturated region of operation of $f(\sigma) = \tanh(\sigma)$ is given by $2d$. The width of the non-saturated region of operation increases as the gradient of the nonlinear function decreases, narrow nonlinear functions corresponding to those with steep gradients. For a wider nonlinearity, d extends further into the context-sensitive region giving a larger number of feature vectors for which classification causes movement of the decision boundary (compare Figs. 5.9(c) and 5.9(d) in section 5.5.2).

This has implications for the ability of the network to account for context in trajectories of feature vectors from the same class. Consider the classification of a trajectory of class 0 feature vectors, shown by the solid line trajectory in Fig. 5.7(a), with all feature vectors located in the region BCD. Feature vectors from region A are not considered because they are always classified as class 1. Fig. 5.7(b) shows the feature space projection $(\boldsymbol{v}^{\mathrm{T}}\boldsymbol{x}(t))$ in the case where saturation at the class 0 limit has occurred (for example after classification of a feature vector from region D or classifications of feature vectors from the initial part of the trajectory).

The decision boundary will only begin to move when a feature vector is classified which is located in the trajectory-sensitive region shown (shaded) in Fig. 5.7. This corresponds to the non-saturated region of operation of the nonlinear function, which is shown by d in Fig. 5.7(b). Most feature vectors lie in the trajectory-insensitive region, where saturation of the nonlinearity persists and prevents movement of the decision boundary from the current class limit. As illustrated in Fig. 5.7(a), a trajectory (or part of a trajectory) of feature vectors which fails to enter the trajectory-sensitive region is effectively classified



(a) Examples of class 0 trajectories

(b) Feature space projection ($\boldsymbol{v}^{\mathrm{T}}\boldsymbol{x}(t)$) showing trajectory-sensitive region for a decision boundary at class 0 limit

Figure 5.7: An example of trajectory-sensitivity for trajectories of class 0 feature vectors. Only those trajectories which pass into trajectory-sensitive regions (shown shaded) give rise to classifications which are dependent on the order of presentation of feature vectors in the trajectory.

by a static boundary at the current class limit. Classifications of feature vectors on such trajectories are unaffected by the order in which the feature vectors occur and no within-class context is taken into account. It is also worth noting that any trajectory of vectors which is parallel to the decision boundary will not cause movement of the decision boundary ($\boldsymbol{v}^{\mathrm{T}}\boldsymbol{x}(t)$ is constant) and is therefore trajectory-insensitive, even if all feature vectors on the trajectory lie within the trajectory-sensitive region of the feature space. Any trajectory which crosses from one context-insensitive region to another will cause movement of the decision boundary and will result in a change in class decision (misclassification).

When the decision boundary is located at a class limit, the trajectory-sensitive region of the feature space is defined by d, the width of the non-saturated region of operation of the nonlinear function, Fig. 5.7(b). The entire context-sensitive region will be trajectory-sensitive if it is completely spanned by d. In contrast, a narrower nonlinear function with a larger maximum gradient gives much less trajectory-sensitivity. In the limit $d = 0$, the tanh nonlinearity is replaced by a step function. In this case, the only possible positions of the decision boundary are at the respective class limits because the hidden unit always operates in saturation. All feature vectors in the context-sensitive region are trajectory-insensitive because their classification does not move the decision boundary from its position at the current class limit. Context-sensitive feature vectors are therefore always classified as belonging to the same class as the previous feature vector. The class decision can only be

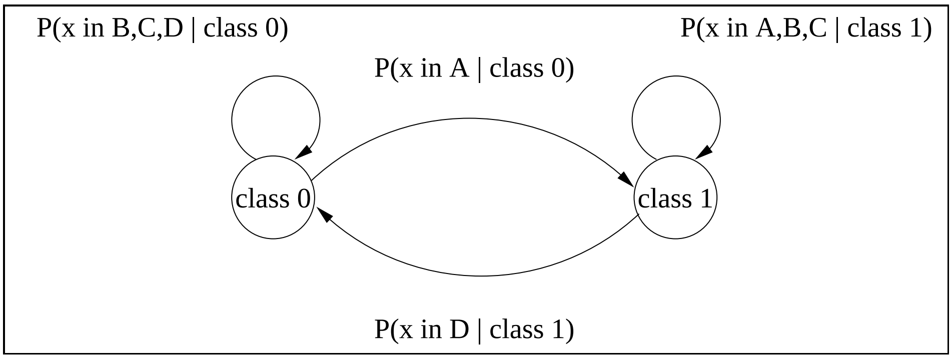


Figure 5.8: Two-state HMM equivalent to a single node recurrent net with step nonlinearity.

changed by classification of feature vectors from context-insensitive regions of the feature space. The classifier acts like a two-state HMM, as illustrated in Fig. 5.8, in which the two states represent the two possible positions of the decision boundary at each class limit. The state transition probabilities reflect the probability of the next feature vector being in a particular region of the feature space, given the current state. Within a state, all feature vectors are treated as independent, in the same way that the network treats all feature vectors which do not move the decision boundary as independent.

Fig. 5.7 illustrates that when saturation occurs, only a narrow region of the feature space is trajectory-sensitive. When trained with fixed class targets, the hidden node tends to operate in saturation for the duration of a class and only enters non-saturated regions of operation when switching state at class boundaries. Under these training conditions, the resultant network will have a limited ability to account for the within-class context exhibited by trajectories of feature vectors from the same class. A possible solution to this problem is to ensure that the operation of hidden units remains out of saturation. This can be enforced by training with non-constant class targets. A suitable target might also reflect the increasing confidence in a decision as more evidence is accumulated. Etemad (1993) proposed using Gaussian targets as a means of representing the instant of recognition within the duration of each class. The use of other types of target function was discussed in section 1.2.2.

The limited trajectory-sensitivity of networks trained with fixed class targets is illustrated in section 5.7, for a synthetic two-class problem, and in section 5.8, for the voiced-unvoiced classification of speech utterances. The use of ramp and exponential target functions for improving sensitivity to within-class context is also investigated in these experiments.

5.5 Implications for Classifier Performance

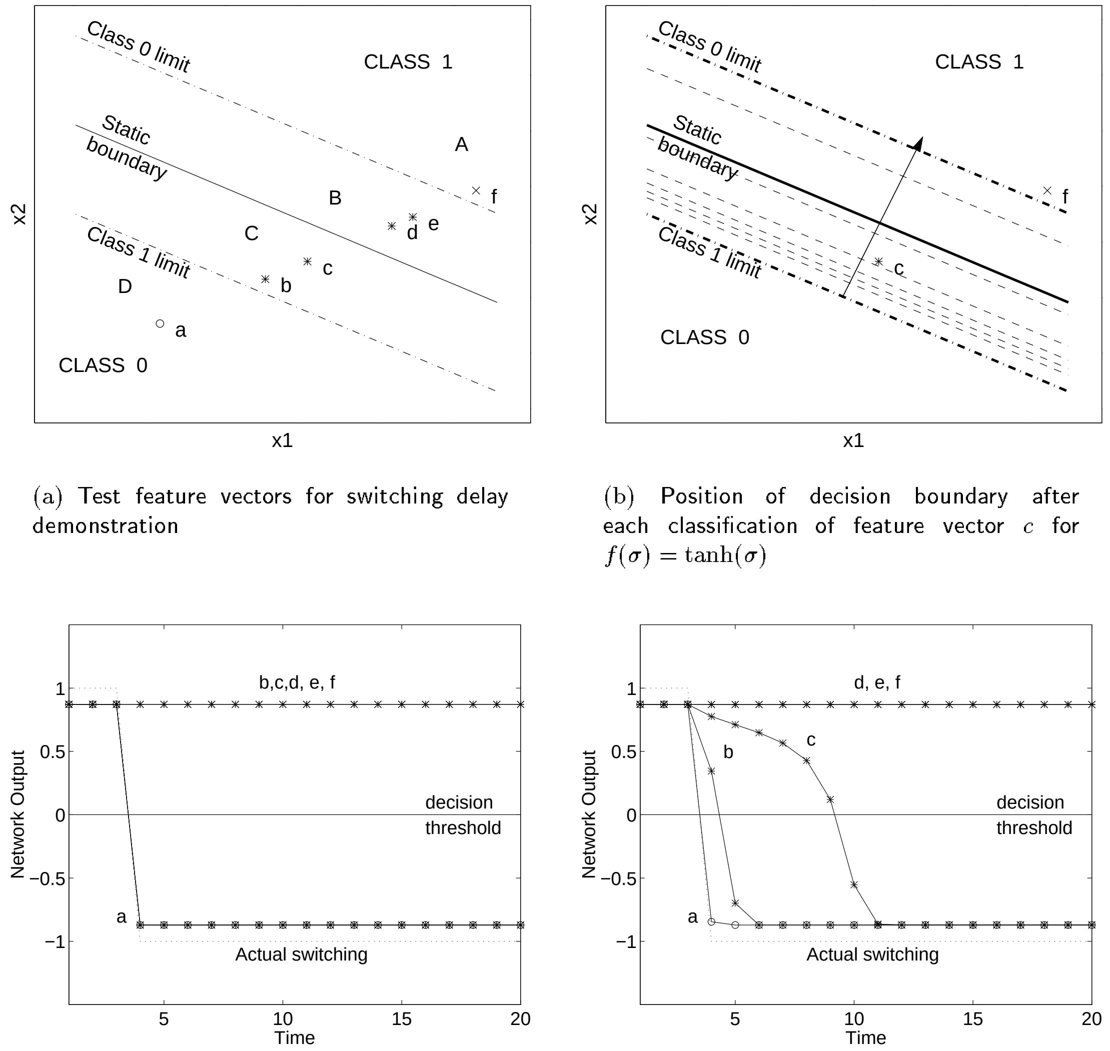
5.5.1 Misclassifications

Errors in the classification of feature vectors from the context-insensitive regions of the feature space are the same for the static and dynamic decision boundary. Classification of a feature vector from a context-sensitive region of the input space is dependent on the current position of the decision boundary. The errors in classifying context-sensitive feature vectors by the dynamic boundary are not predetermined. The movement of the decision boundary can cause both correction of, and addition to, the errors in classification by the static decision boundary. The following sections show that correction of misclassifications is a result of the smoothing action of the recurrent connection on output decisions and that additional errors are made as a result of the switching delay introduced at class boundaries.

5.5.2 Between-Class Context - Switching Delay

When trained with fixed class targets, the equilibrium configuration for a class occurs when the hidden unit operates in saturation and the decision boundary is located at a class limit. Due to the positive feedback connection, w, which favours previous class decisions, it takes a finite time for the decision boundary to move from one class limit to the other and as a result the classifier output exhibits a finite switching delay at class boundaries. This is illustrated for the two-dimensional two-class problem by considering how the output of the network varies when the same feature vector is repeatedly presented for classification. In Fig. 5.9(a), a set of test feature vectors are illustrated, which are located in the different regions of the feature space. These feature vectors are used to form trajectories, for example $fbbb\ldots$, $fccc\ldots$, $fddd\ldots$, in which each feature vector is repeated. The initial presentation of feature vector f locates the decision boundary at the class 1 limit, as illustrated in Fig. 5.9(b). Figs. 5.9(c) and 5.9(d) compare the different outputs obtained for classifiers with a narrow nonlinearity, $f(\sigma) = \tanh(10\sigma)$, and a wider nonlinearity, $f(\sigma) = \tanh(\sigma)$. The outputs show that not all feature vectors can cause switching. For the narrower nonlinearity, none of the context-sensitive feature vectors cause switching because they all lie in regions of saturated operation of the hidden node nonlinearity. For the wider nonlinearity, feature vectors which cause switching are those which lie in the regions of the feature space where the nonlinear function operates out of saturation (the shaded region d in Fig. 5.6(a)). For a wider nonlinear function, the non-saturated region extends further into the context-sensitive region so more points switch.

For the classification of feature vector c in Fig. 5.9(d), the corresponding positions of the decision boundary after each classification are shown in Fig. 5.9(b). It can be seen that the switching delay is a direct result of the finite time taken by the decision boundary to move from the class 1 to class 0 limit. This suggests that a longer switching delay would be observed for a wider context-sensitive region (larger $|w|$).



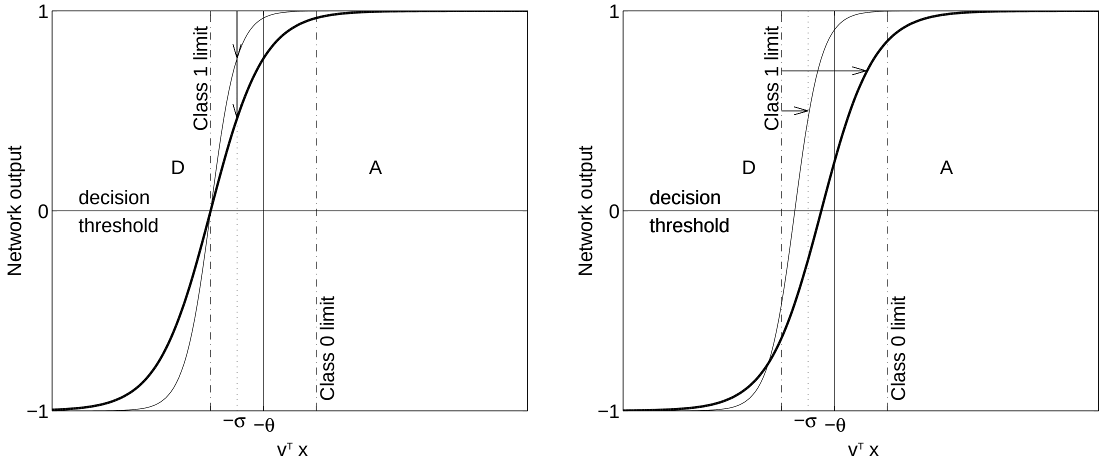
(a) Test feature vectors for switching delay demonstration

(b) Position of decision boundary after each classification of feature vector c for $f(\sigma) = \tanh(\sigma)$

(c) Output for narrow nonlinear function, $f(\sigma) = \tanh(10\sigma)$

(d) Output for wide nonlinear function, $f(\sigma) = \tanh(\sigma)$

Figure 5.9: Effect of gradient of nonlinear function on switching delay. The classifier outputs are those which result from repeated presentation of each of the feature vectors shown in Fig. 5.9(a).



(a) Position of decision boundary after classification of a feature vector from class 1

(b) Position of decision boundary after presentation of feature vector $\boldsymbol{x}$, where $\boldsymbol{v}^{\mathrm{T}}\boldsymbol{x} = \sigma$

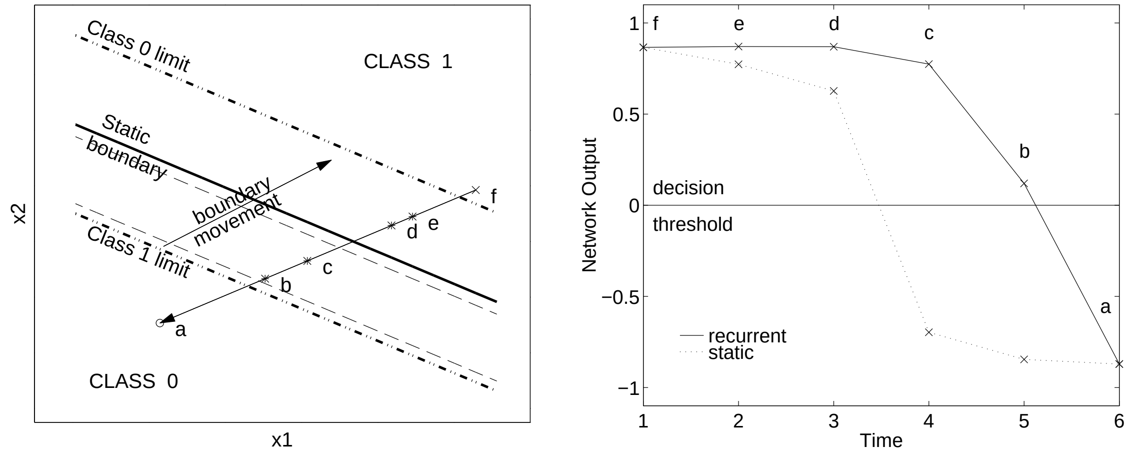
Figure 5.10: Effect of gradient of nonlinear function on switching speed.

For a feature vector that causes switching with either a wide or narrow nonlinear function, the speed of switching is faster for the wider nonlinearity because the change in feedback is greater. This causes a larger initial movement of the decision boundary from the initial class limit. This is illustrated in Fig. 5.10 for a feature vector $\boldsymbol{x}$ ($\boldsymbol{v}^{\mathrm{T}}\boldsymbol{x} = \sigma$), in which it can be seen that classification using a wider nonlinear function moves the decision boundary further toward the class 0 limit than the corresponding classification using a narrower nonlinear function, Fig. 5.10(b). In this illustration, switching by the wider nonlinear function occurs on the second presentation of vector $\boldsymbol{x}$, whereas further presentations are required before switching occurs with the narrower nonlinearity.

The implication for classifier performance is that the switching delay from one class to another can cause classification errors at class boundaries. However, for a trajectory of vectors from the same class, the switching delay can aid in correcting misclassifications by smoothing the classifier output decisions over a number of consecutive feature vectors as illustrated in the following section.

5.5.3 Within-Class Context - Output Smoothing

The delay in switching from one class to another can be beneficial in preventing misclassifications of feature vectors from class trajectories which span the context-sensitive region of the feature space. This is illustrated in Fig. 5.11, for the class 1-0 trajectory shown in Fig. 5.11(a). The slow movement of the decision boundary, from the class 1 limit to the class 0 limit, is illustrated in Fig. 5.11(a), which shows successive positions of the decision boundary (dashed line) after classification of each feature vector on the trajectory. The



(a) Decision boundary movement for class 1-0 trajectory

(b) Comparison of recurrent network output with that of static decision boundary defined by $\boldsymbol{u}$ and $\boldsymbol{v}$

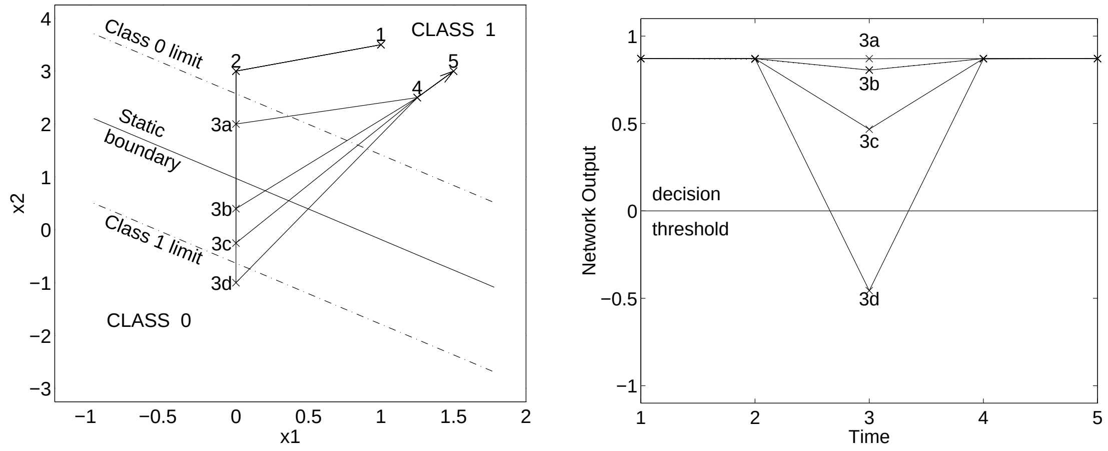
Figure 5.11: Operation of classifier on a class 1-0 trajectory. Fig. 5.11(a) shows successive positions of the decision boundary (dashed line) after classification of each feature vector f, e, d, c, b, a. Fig. 5.11(b) shows recurrent decisions lag those of the static boundary.

slow movement causes the output of the recurrent network to lag that given by the static decision boundary defined by $\boldsymbol{u}$ and $\boldsymbol{v}$. As a result, feature vectors 'c' and 'b' are correctly classified by the recurrent network, although the misclassification of feature vector 'a' is not corrected.

The lagged output of the recurrent network can smooth the class decisions from the corresponding static boundary and prevent spurious misclassifications of feature vectors on a trajectory of examples from the same class. This is demonstrated by Fig. 5.12. In Fig. 5.12(a), several trajectories are illustrated, for which it is assumed that all feature vectors belong to class 1. The feature vectors $3b$, $3c$, and $3d$ are misclassified by the static boundary. Fig. 5.12(b) shows the output of the network in classifying these trajectories. Due to the delay in switching, caused by the recurrent connection, the incorrect classification of feature vectors $3b$ and $3c$, by the static boundary, is corrected by the dynamic boundary. The misclassification of $3d$ is not corrected because this feature vector lies in the context-insensitive region of the feature space.

5.6 Larger Networks

The analysis so far has considered a simple recurrent network with a single hidden unit and can be extended to multiple hidden unit networks with self-feedback. With multiple hidden units, the decision boundaries become nonlinear (for examples, see Fig. 5.13) and result as a combination of local decisions by each unit. With cross-terms in the feedback



(a) Examples of class 1 trajectories

(b) Output decisions for recurrent network and static boundary showing output smoothing by recurrent network

Figure 5.12: Effect of boundary movement on output smoothing for class 1 trajectories.

matrix, $\boldsymbol{W}$, feedback between hidden units determines how previous local decisions in other regions of the feature space affect the current local decision. The effect of changing the pattern of feedback in hidden units is to change both the shape and position of the decision boundary. The limiting shapes and positions of the boundary are defined by the pattern of saturation of hidden units for each class. Although there are 2^{n_h} possible saturation patterns, usually there are only as many patterns as classes in a well trained network solution. When considering the effect of saturation, it is necessary to take into account the magnitude and sign of the output weights, since these determine the relative contribution that each hidden unit makes to the final decision. When viewing hidden unit outputs, this contribution can be reflected by scaling the output of each hidden unit by a scale factor k

$$k = \frac{h_i(t)u_i}{\sum\limits_{j=1}^{n_h} |u_j|} \qquad\qquad i = 1 \ldots n_h \tag{5.4}$$

Each permutation of operation of the hidden units defines context-sensitive and context-insensitive regions of the feature space (regions where the classification of feature vectors is always independent of previous classifications). The resulting region of the feature space which can be considered context-sensitive is given by the union of these regions. Therefore, as the number of hidden units increases, the context-sensitive region is also likely to increase in size.

For a single hidden unit, stability requires the feedback weight to be positive (Frasconi, Gori & Soda 1992). The effect of the recurrent connection is thus to favour previous class decisions. With multiple hidden units, both positive and negative feedback weights are possible and the pattern of the sign of weights which is developed in training is determined by stability. For a linear network, stability of the linear dynamical system is achieved when all the eigenvalues of $\boldsymbol{W}$ are real and of magnitude less than 1. Analysing the stability of the nonlinear system is a difficult task, however. Preliminary work in this area by Tiño, Horne & Giles (1995) has studied the stability, in training, of two interconnected units with recurrent connections. They divided the feature space into regions according to the nature of the equilibrium points of the dynamical system exhibited in each region, which is determined by the pattern of the sign of the feedback weights. For a trained network, these regions could be used to determine which feature vectors permit switching of the network output when they are repeatedly presented to the network for classification.

In the following sections, two small-scale experiments are presented to illustrate how the observations made in this chapter extend to networks with multiple hidden units and to classification tasks using higher dimension feature vectors.

5.7 Classification of 2D Vector AR processes

In this section, the operation of small recurrent network classifiers is illustrated on a synthetic 2-class problem. Two-dimensional feature vectors are used to allow visualisation of decision boundaries. To assess the relevance of the observations of this work for speech pattern classification tasks, a synthetic classification problem was set up, simulating the correlation between feature vectors which is a characteristic of speech pattern classification tasks. Each class example consisted of a trajectory of vectors which was generated from two vector AR processes, and the 2 classes were distinguished by the clockwise or anti-clockwise rotation of the trajectories. The process was defined by

$$\boldsymbol{X}(t) = \boldsymbol{M}_i\boldsymbol{X}(t-1) + e(t) \tag{5.5}$$

where the vectors $\boldsymbol{X}(t)$ define a vector trajectory in feature space and $\boldsymbol{M}_i$ is a matrix of parameters for class $i \in \{0, 1\}$. M_i rotates and scales previous vectors. $e(t)$ is zero mean, Gaussian noise term. The variance of the noise was low, so that successive feature vectors show a high degree of correlation. Training and testing sequences were set up by forming an alternating sequence of class 0 and class 1 trajectories, in the pattern $[0, 1, 0, 1 \ldots]$ Successive $[0, 1]$ pairs were generated from the same random seed vector, $\boldsymbol{X}(0)$. The classification problem is not linearly separable because the $\boldsymbol{X}(0)$ for each $[0, 1]$ pair is classified as belonging to either class, depending on context. Although the feature vector sequences simulate the correlation between feature vectors within a class, they do not represent between-class context as well because there are discontinuities in the value of feature vectors across class boundaries.

In Fig. 5.13, examples of vector trajectory pairs making up a typical training sequence are shown, and were generated by the matrices M_0 (clockwise) and M_1 (anti-clockwise)

	TRAIN		TEST	
FIXED TARGET	MSE	% Correct	MSE	% Correct
$n_h = 1$	0.42	87	0.45	86
$n_h = 2$	0.23	91	0.32	87
$n_h = 3$	0.27	92	0.28	90
EXP TARGET	**TRAIN**		**TEST**	
$n_h = 2$	0.12	87	0.18	87
RAMP TARGET	**TRAIN**		**TEST**	
$n_h = 2$	0.13	81	0.17	76

Table 5.1: Performance results for networks trained to classify vector AR processes.

$$\boldsymbol{M}_0 = \begin{bmatrix} 0.85 & -0.2 \\ 0.1 & 0.9 \end{bmatrix} \qquad \boldsymbol{M}_1 = \begin{bmatrix} 0.85 & 0.2 \\ -0.1 & 0.9 \end{bmatrix}$$

Single output, single hidden layer recurrent networks, Fig. 4.1, with 1, 2 and 3 hidden units, were trained to classify the training sequence using fixed class targets $(-1, +1)$, ramp targets $(-1 \rightarrow +1, +1 \rightarrow -1)$, and exponential targets given by $y(t)$

$$y(t) = \pm\, 2 \left[1 - e^{\left(\frac{-t}{T}\right)} \right] - 1 \tag{5.6}$$

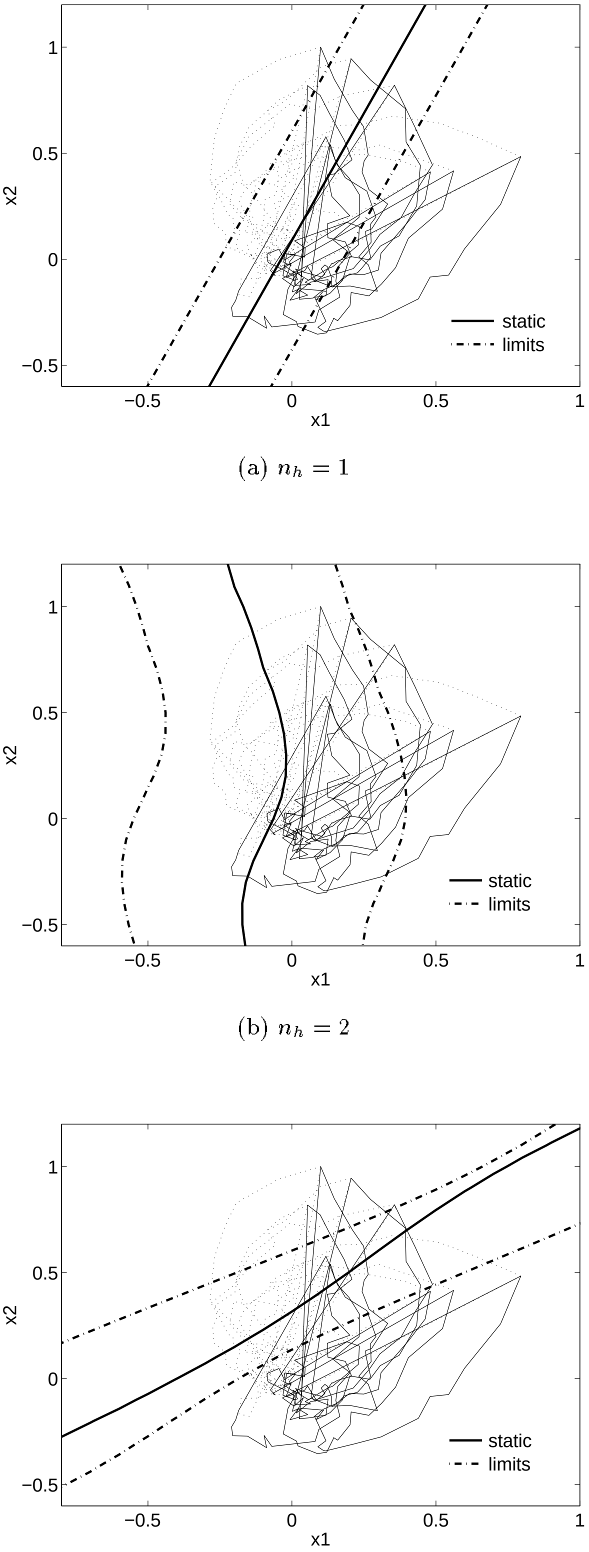
where T was set equal to $N/3$ and N is the duration of the current class[1]. Training was performed by Real Time Recurrent Learning using small, monotonically decreasing learning rates and network weights were initialised from a zero mean, Gaussian distribution.

Fig. 5.13 shows examples of the limiting positions of the decision boundary formed by networks with $n_h = 1$, 2 and 3. The decision boundaries shown by the dot-dash lines are those obtained when the hidden units adopt each pattern of saturation for the two classes. The static boundary (zero feedback) is illustrated by the solid line. These figures illustrate how the shape and position of the decision boundary is altered by changes in feedback, for networks with $n_h > 1$.

The performance of these networks in training and testing is shown in Table 5.1, where mean-squared error (MSE) and percentage correct classifications (% Correct) are given. Percentage correct classifications were evaluated by classifying each feature vector of the input sequence using a zero threshold at the network output. The performance in training and testing is similar which shows that suitable decision boundaries are learnt. The improvement in performance obtained by increasing the number of hidden units from $n_h = 1$, is due to the improved classification capabilities of a nonlinear class boundary.

For the single hidden node network, Fig. 5.14(a) compares the network output with that of the extracted feed-forward network (static boundary), for classification of the trajectories shown in Fig. 5.13(a). At class boundaries, the delayed switching of the network output, when compared with the output of the feed-forward network, is observed (for ex-

[1] As T is increased, the exponential targets become more like ramp targets and as T is reduced, they become more like fixed targets.



(a) $n_h = 1$

(b) $n_h = 2$

(c) $n_h = 3$

Figure 5.13: Feature space for vector AR processes, showing limiting positions of the decision boundary when hidden units saturate. The vector trajectory pairs in the training sequence are also shown (clockwise by solid lines, anti-clockwise by dotted lines).

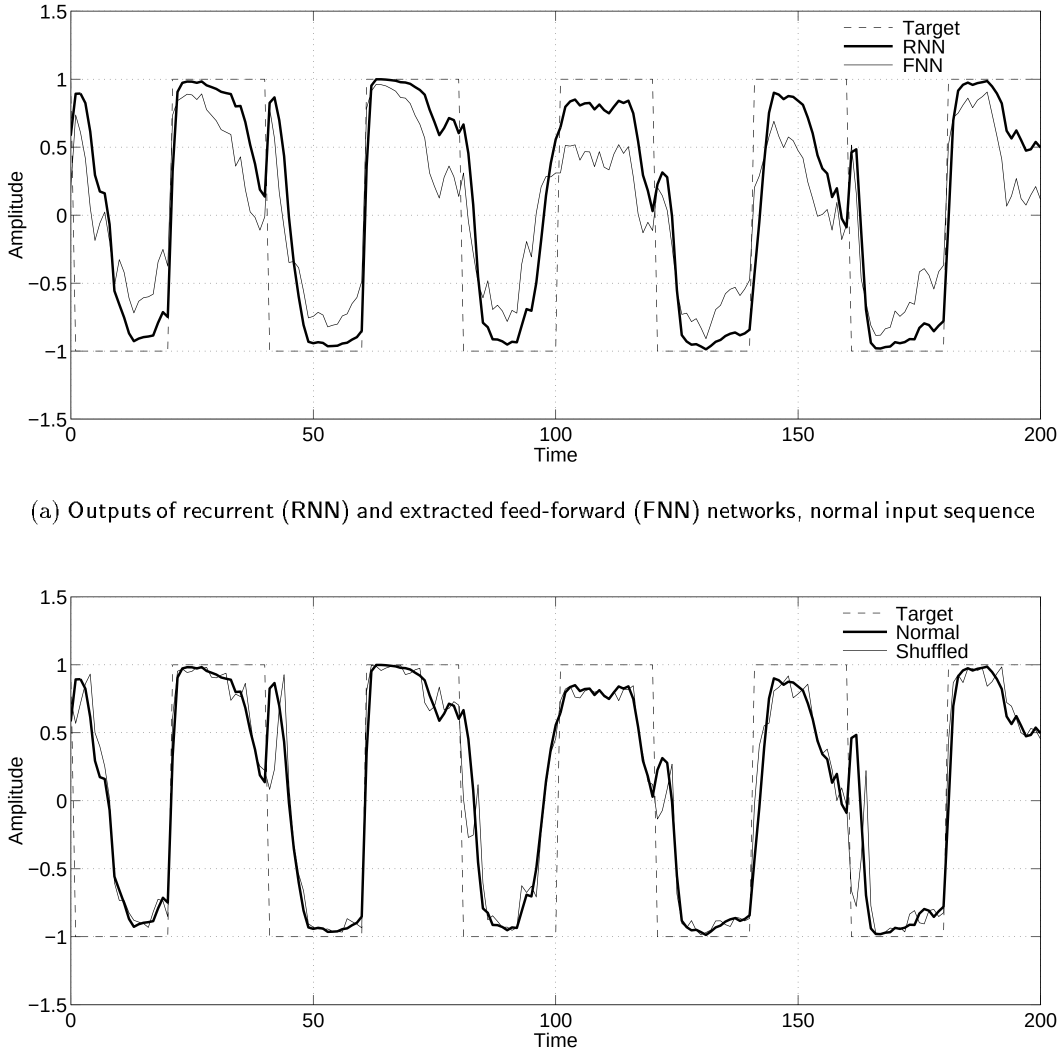
ample around frame number 100), and the smoothing effect of the recurrent connection is also illustrated (for example around frame numbers 40 and 120). It can be seen that the hidden unit operates close to saturation within a class. To demonstrate that the network is relatively insensitive to the order of presentation of feature vectors while saturation of the hidden unit persists, groups of 5 consecutive feature vectors within a class were reversed in order, and the shuffled sequence of feature vectors was re-classified. Fig. 5.14(b) shows the outputs obtained in both cases. It can be seen that, within a class, where the hidden unit operates close to saturation, the network is insensitive to the order of presentation of the feature vectors. Most degradation in the output is observed at class boundaries, where the hidden unit switches, and therefore enters the non-saturated region of operation.

In Fig. 5.15(a), similar results are shown for the network with 2 hidden nodes. The scaled outputs from the hidden nodes[2], corresponding to the normal input sequence, are illustrated in Fig. 5.15(b). The hidden units adopt a two-state pattern of saturation which corresponds to the limiting positions of the decision boundaries illustrated by dot-dash lines in Fig. 5.13(b). Comparing Fig. 5.15(a) and 5.15(b), it can be seen that in regions where both hidden nodes operate close to saturation, the network is relatively insensitive to the order of presentation of the feature vectors, for example around frame numbers 40 and 70. However, when at least one node enters a non-saturated region of operation, the network becomes sensitive to the context of the input feature vectors, and degradation of the network output is observed (around frames 90 and 170, for example).

In section 5.4, using non-constant class targets was proposed as a means of forcing hidden units to operate out of saturation, so that sensitivity to within-class context is improved. Figs. 5.16(a) and 5.16(b) show the network output and corresponding operation of hidden units for a two hidden unit network trained using exponential class targets, in classifying the trajectories illustrated in Fig. 5.13(b). The output obtained from the network when classifying the shuffled sequence of input feature vectors is also shown. It can be seen that the effect of training with non-constant class targets is to force at least one hidden node to operate out of saturation. As a result, the network is more sensitive to changes in the order of the feature vector sequence, especially in regions where both units operate out of saturation (for example around frame 170).

The performance figures for the network trained with exponential class targets are shown in Table 5.1. Although a lower MSE was obtained, the percentage correct classifications from this network was lower than that obtained from the corresponding network ($n_h = 2$) trained with fixed class targets. This is due to the use of a threshold decision, which has the effect of amplifying errors in the network output. For networks trained with exponential targets, small changes in network output at class boundaries only cause a small change in MSE, but can cause a change in class decision. Comparing MSE performance can also be misleading because the absolute magnitude of exponential targets is smaller than that of fixed class targets for most of the duration of a class. For comparison purposes, the performance of a two hidden unit network trained with ramp targets is also

[2] see section 5.6.



(a) Outputs of recurrent (RNN) and extracted feed-forward (FNN) networks, normal input sequence

(b) Output of RNN for normal and shuffled input sequence

Figure 5.14: Operation of the recurrent network ($n_h = 1$), trained with fixed class targets, in classifying the vector AR trajectories shown in Fig. 5.13(a). Fig. 5.14(a) illustrates smoothing and switching delay of RNN compared to FNN. Fig. 5.14(b) compares the outputs of RNN for a normal and shuffled input sequence, illustrating the insensitivity to trajectories when the hidden node saturates, for example around frames 55 and 110. Class decisions are by a zero threshold on the output.

shown in Table 5.1. Although the ramp and exponential targets are quite similar for the choice of T used, and gave network solutions with similar performance in terms of MSE, it proved more difficult to train networks successfully with ramp targets. This illustrates that the choice of class target can influence the ability to learn a suitable solution, and confirms similar observations made by other researchers, for example Etemad (1993).

This section has demonstrated the effect of saturation, switching delay, and the effect of training with variable class targets for a 2-dimensional classification problem. The following section investigates the effect of increasing the input dimension and number of hidden units for the 2-class classification of speech utterances into voiced and unvoiced segments.

5.8 Voiced-Unvoiced Classification of Speech Utterances

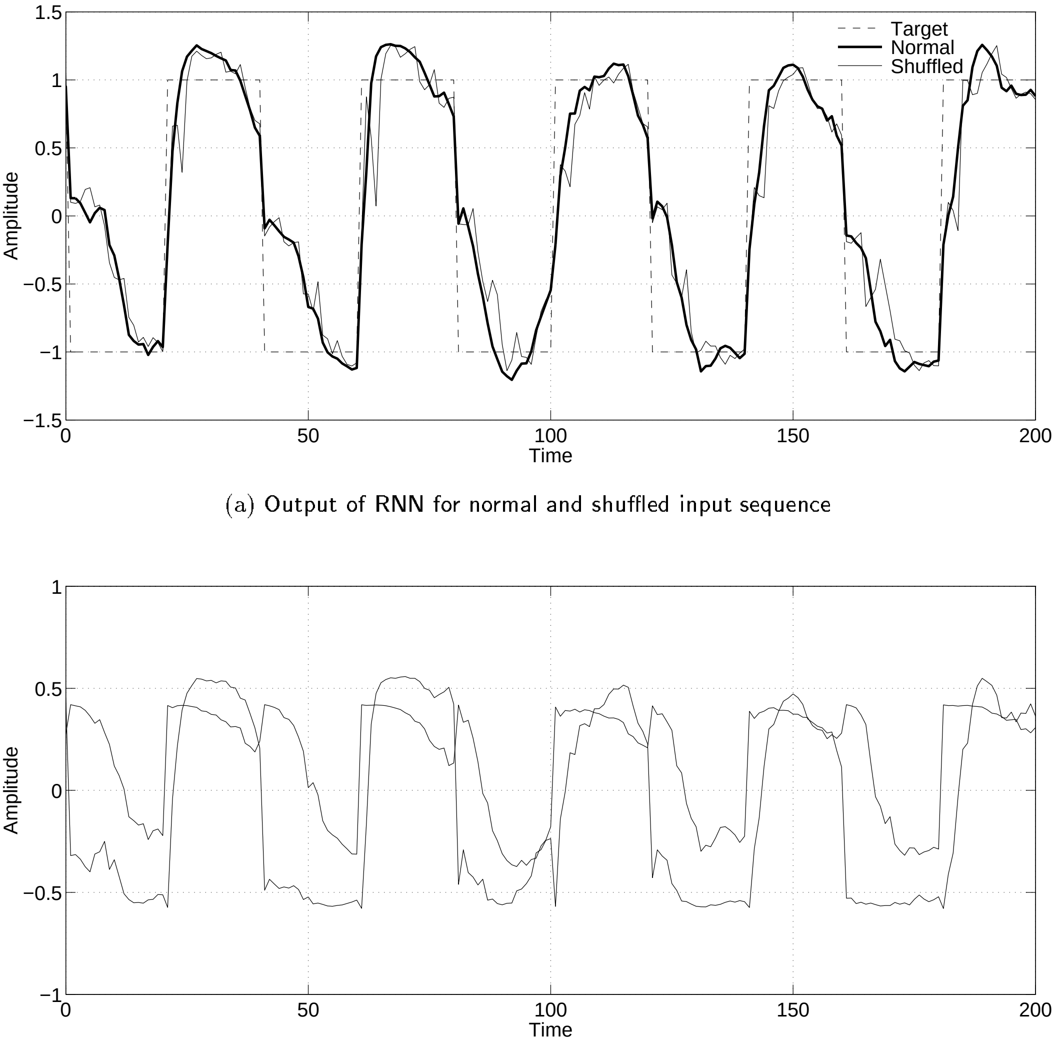
The classification of speech into voiced and unvoiced segments provides a 2-class problem on which to analyse the operation of small recurrent network classifiers, using higher dimension feature vectors obtained from real speech data. A random selection of 10 sentences from dialect region 1 of the TIMIT database (Garofolo 1988) were chosen as the source of training data and a further 4 sentences were selected for testing. The distribution of voiced and unvoiced classes was approximately 72% to 28% in both training and test data sets. The acoustic waveforms were converted to a sequence of feature vectors of cepstral coefficients (Rabiner & Schafer 1978), using a 10^{th} order analysis on 20ms windows of data, at a 10ms frame rate. Each feature vector was assigned a voiced or unvoiced class label, according to the mapping in Table 5.2. The TIMIT phone label for each frame was evaluated as the most common phone label within the corresponding analysis window.

PHONE TYPE	VOICED	UNVOICED
Vowels	aa,ae,ah,ao,aw,ax,ax-h,axr,ay eh,er,ey ih, ix, iy oh,oy uh,uw,ux	- - - - -
Stops	b,d,g,dx	p,t,k,q
Closures	-	bcl,dcl,gcl,pcl,tck,kcl,tcl
Fricatives	z,zh,v,dh	s,sh,f,th
Glides	l,r,w,y,el,hv	hh
Affricated	-	jh,ch
Silence	-	epi,pau,h#

Table 5.2: Mapping between TIMIT phone labels and voiced-unvoiced classes.

The single hidden layer, recurrent architecture of Fig. 4.1 was used, with 5 hidden nodes and 11 inputs (feature vectors were augmented to include a bias term). 20 networks were trained using fixed class targets and 20 using exponential targets, from different random initialisations of the weights and an average performance was calculated.

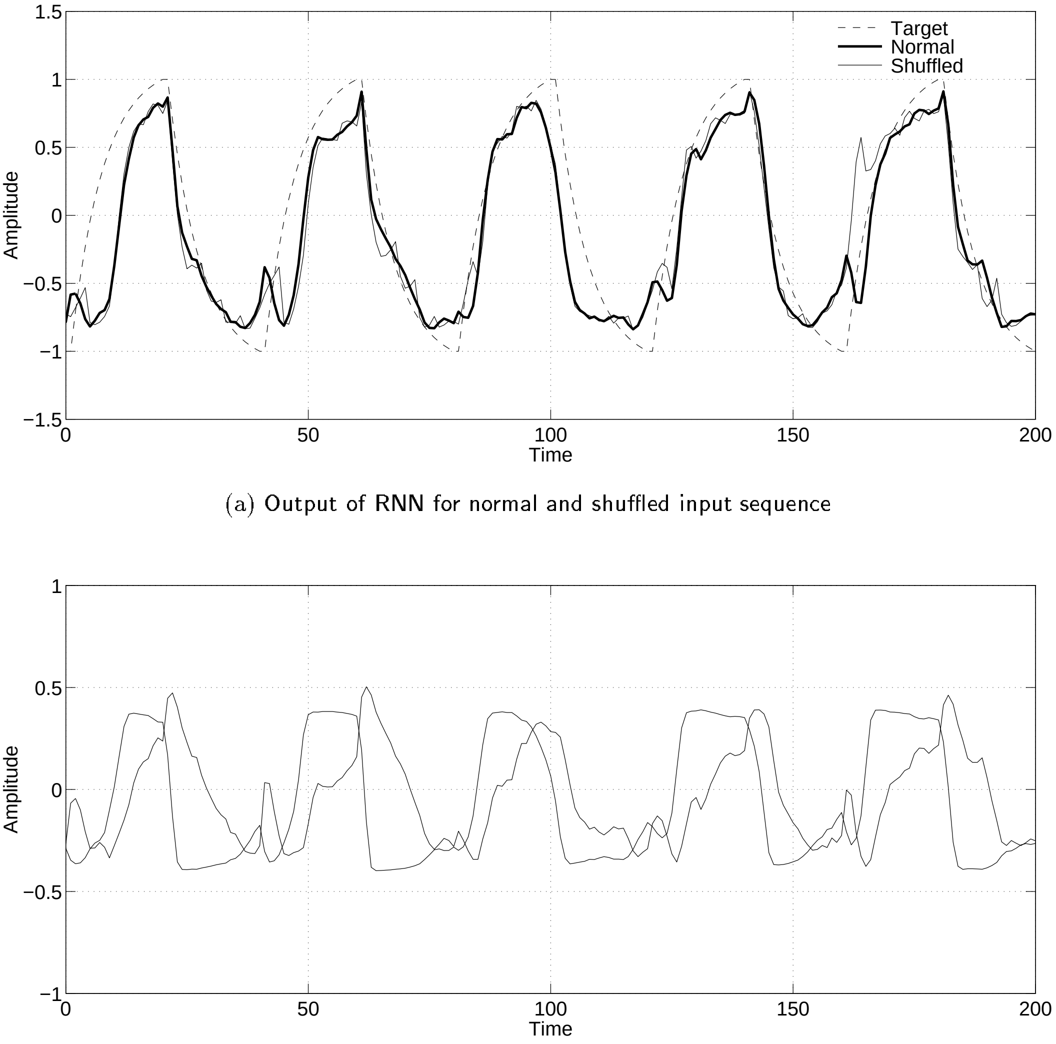
For networks trained with fixed class targets of ±1, Table 5.3 shows the resulting per-



(a) Output of RNN for normal and shuffled input sequence

(b) Scaled outputs of hidden nodes for normal input sequence

Figure 5.15: Operation of the recurrent network ($n_h = 2$), trained with fixed class targets, in classifying the vector AR trajectories shown in Fig. 5.13(b). Fig. 5.15(a) compares the outputs of RNN for a normal and shuffled input sequence, and illustrates trajectory insensitivity when both hidden nodes operate close to saturation. Fig. 5.15(b) shows the corresponding operation of individual hidden nodes. Class decisions are by a zero threshold on the output.



(a) Output of RNN for normal and shuffled input sequence

(b) Scaled outputs of hidden nodes for normal input sequence

Figure 5.16: Operation of recurrent network ($n_h = 2$), trained with exponential class targets, in classifying the vector AR trajectories shown in Fig. 5.13(b). Fig. 5.16(a) compares the outputs of RNN for a normal and shuffled input sequence, and illustrates trajectory sensitivity when both hidden nodes operate out of saturation. Fig. 5.16(b) shows the corresponding operation of individual hidden nodes. Class decisions are by a zero threshold on the output.

	TRAIN	TEST
n_d	1.7	3.6
n_s	2.6	3.3
% Correct (RNN)	88	85
% Correct (FNN)	87	85
MSE (RNN)	0.37	0.45
MSE (FNN)	0.47	0.53

Table 5.3: Comparison of performance of recurrent networks (RNN) and extracted feed-forward network (FNN) with equivalent weights $\boldsymbol{U}$ and $\boldsymbol{V}$. Fixed class targets were used in training and the performance was averaged over 20 networks classifying all sentences.

formance on training and test data, in terms of mean-squared error (MSE) and percentage correct classifications. In classification, the feature vectors were presented to the network and a class decision for each vector was made based on an output threshold at zero. The performance figures do not reflect state-of-the-art performance in voiced-unvoiced classification of speech, but verify that the networks have extracted some information in the learning procedure. To evaluate the effect of recurrent connections on classifier performance, the performance of the extracted feed-forward network (FNN) defined by $\boldsymbol{U}$, $\boldsymbol{V}$, is also quoted. Two comparative measures, n_s and n_d, are used in Table 5.3. n_d measures the classification errors made by the recurrent network due to switching delay. n_d was evaluated relative to the FNN performance, and is given by

$$n_d = \frac{\sum \text{RNN incorrect and FNN correct}}{N_{\text{total}}}$$

where N_{total} is the total number of feature vectors in the training or test data. n_s measures the corrections by the recurrent network (of classification errors by FNN) due to smoothing of output decisions, and is given by

$$n_s = \frac{\sum \text{RNN correct and FNN incorrect}}{N_{\text{total}}}$$

A measure of switching delay is made relative to the performance of FNN, rather than by counting the misclassifications made by the recurrent network at class boundaries because simple counting would not reflect the errors that are due to switching delay alone. In speech, there are no clearly defined class boundaries and transitions from one class to another occur gradually. As a result, classification errors are likely to occur at class boundaries. In calculating a measure of switching delay which is dependent on the performance of FNN, errors which are also made by FNN are disregarded.

Table 5.3 shows a higher percentage of corrections by smoothing than additional errors from switching delay. On average, the recurrent connections were beneficial to the classifier performance. This is illustrated by the improved performance of RNN, both in terms of MSE (lower) and percentage correct classifications (higher). Comparison of MSE and percentage correct classifications for test data shows the same percentage correct for FNN

TARGET	% Correct		MSE	
	TRAIN	TEST	TRAIN	TEST
FIXED	88	85	0.37	0.45
EXP	83	81	0.26	0.31

Table 5.4: Comparison of performance of recurrent networks trained with fixed and exponential targets for voiced-unvoiced classification of speech. Performance was averaged over 20 networks for each target type, in classifying all sentences.

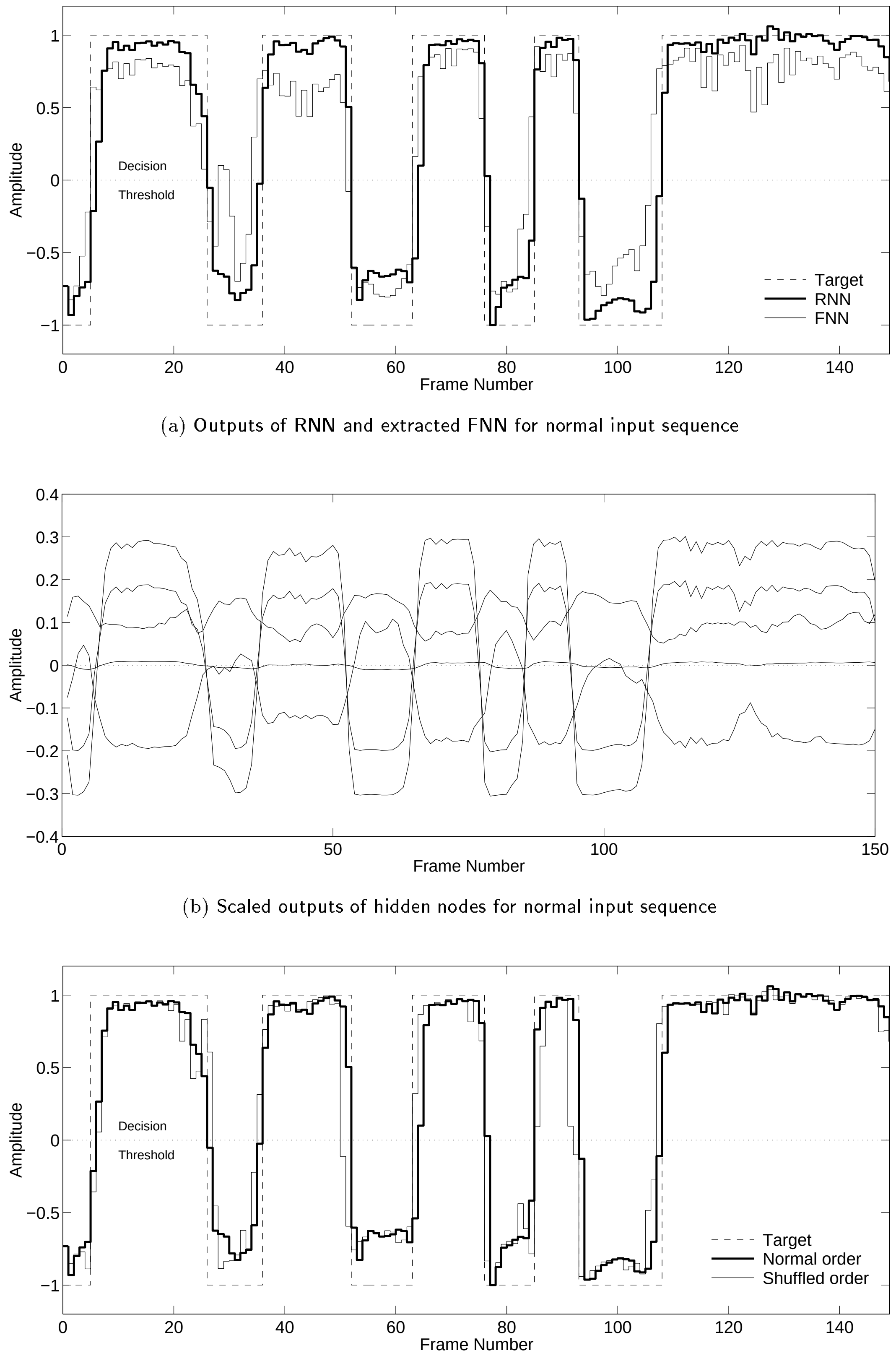
and RNN, despite a higher MSE for FNN. This highlights an attribute of using a threshold criterion to evaluate performance; it does not reflect how closely the network output matches the class targets.

Fig. 5.17(a) compares the output of one of the trained networks with its corresponding FNN (static boundary), for classification of the sentence "John cleans shellfish for a living". Corrections due to the smoothing action of the recurrent connections are observed around frame 30, and delay in switching is seen at class boundaries, both relative to FNN (for example, around frame 110), and relative to the actual class targets (for example, around frames 10 and 65). Fig. 5.17(b) shows the corresponding scaled outputs[3] of hidden nodes, and illustrates that training with fixed class targets tends to drive the operation of hidden nodes into saturation for the duration of a class. For classification of the same sentence, Figs. 5.18(a) and 5.18(b) show the output and operation of hidden units for one of the networks trained with exponential targets, given by Eqn. (5.6). T was set to $N/4$, where N was the duration of the current class. For this network, Fig. 5.18(a) shows poor performance from the extracted feed-forward network (FNN). The operation of the hidden units, Fig. 5.18(b), shows that the effect of training with an exponential target function is to force at least one hidden unit to operate out of saturation. The saturation of all dominant hidden units around frames 120 to 150 results in a network output which fails to track the target function accurately.

Table 5.4 shows a comparison of performance of networks trained with fixed and exponential class targets. For exponential targets, there is a fall in percentage correct classifications compared to networks trained with fixed class targets, despite a lower average MSE from these networks. This highlights the difficulty in comparing the performance of networks trained with different class targets, which was discussed previously in section 5.7.

The sensitivity to within-class context of networks trained with fixed and exponential targets was estimated by reversing the order of groups of 3 feature vectors within a class and presenting the shuffled input sequence to the network to be re-classified. The effect on the network output is shown in Figs. 5.17(c) and 5.18(c), for fixed and exponential targets respectively. These figures show the outputs of the network when classifying both the normal and shuffled sequence of input vectors. For networks trained with fixed class targets, it is seen that despite more than one hidden node, the network is generally insen-

[3] see section 5.6.

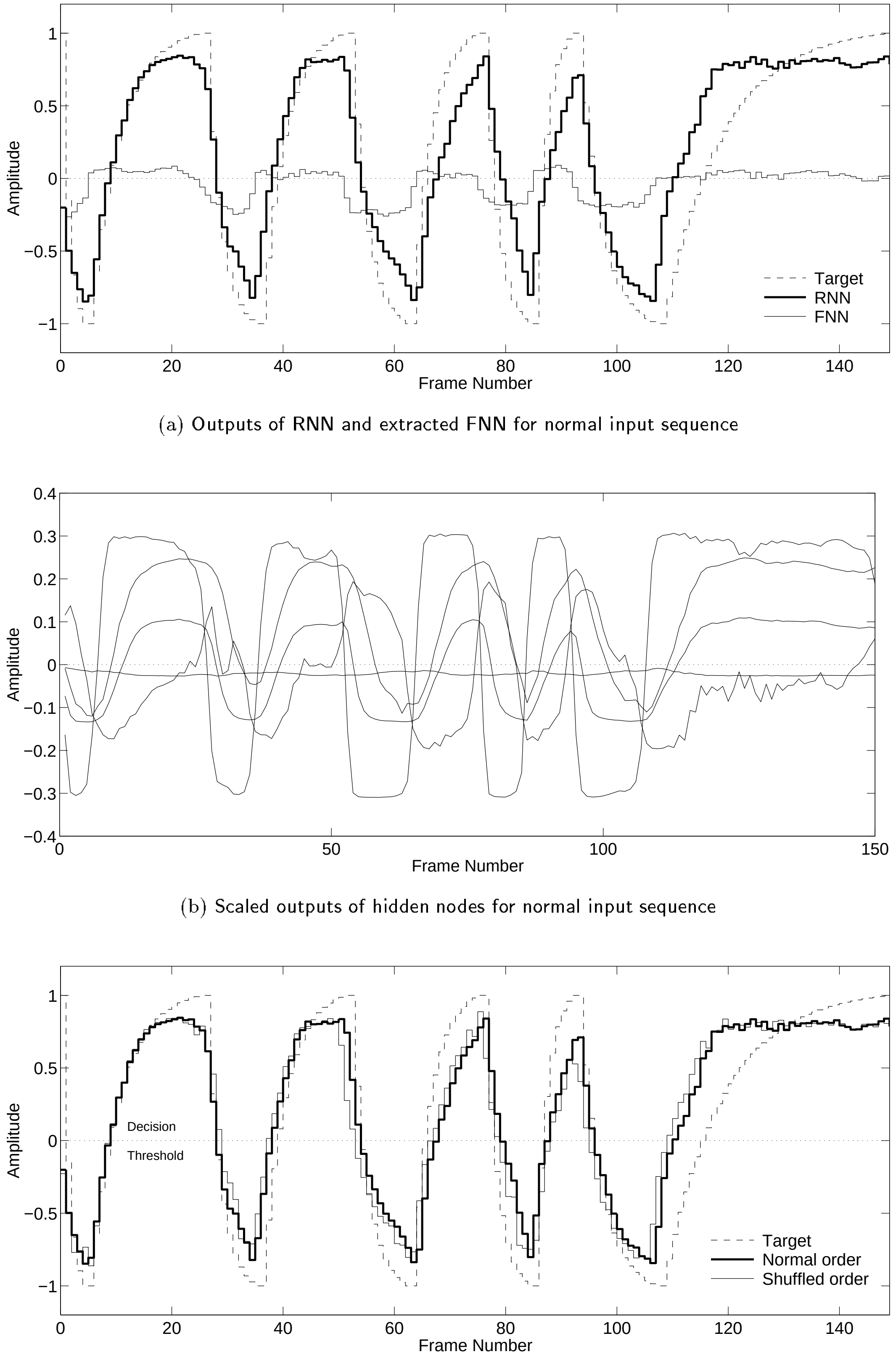


(a) Outputs of RNN and extracted FNN for normal input sequence

(b) Scaled outputs of hidden nodes for normal input sequence

(c) Output of RNN for normal and shuffled input sequence.

Figure 5.17: Operation of recurrent network ($n_h = 5$), trained with fixed class targets, in voiced-unvoiced classification of the sentence "John cleans shellfish for a living".



(a) Outputs of RNN and extracted FNN for normal input sequence

(b) Scaled outputs of hidden nodes for normal input sequence

(c) Output of RNN for normal and shuffled input sequence

Figure 5.18: Operation of recurrent network ($n_h = 5$), trained with fixed class targets, in voiced-unvoiced classification of the sentence “John cleans shellfish for a living”.

sitive to changes in the order of presentation of feature vectors in regions where all hidden nodes operate close to saturation (for example, around frames 15, 70 and 110 to 150). At class boundaries, however, the output of hidden units changes from one pattern of saturation to another and during the transition, the output of the network becomes sensitive to the shuffling of the input sequence (for example, around frame 25). For networks trained with exponential target functions, there is a significant contribution to the network output from non-saturated hidden nodes and as a result, the network output is more sensitive to changes in the order of presentation of input feature vectors. However, in regions where the dominant contributions to the output are from units operating close to saturation, for example for frames 110 to 150, the output of the network is still insensitive to trajectories of feature vectors within a class.

5.9 Concluding Remarks

This chapter has shown how the operation of a single hidden unit recurrent network classifier can be interpreted in terms of the decision boundary formed in feature space. Two of the main characteristics of this operation were shown to be switching delay at class boundaries and smoothing of output decisions for sequences of feature vectors within the same class. These characteristics were shown to be caused by the effect of feedback, which was to favour previous class decisions by moving the decision boundary. Movement of the decision boundary allows the recurrent network to account for the correlation between successive feature vectors, provided hidden units operate out of saturation. When trained with fixed class targets, the effect of presenting the network with a long sequence of feature vectors from the same class is to force hidden units into saturation. It was demonstrated that when a segment of the feature vector trajectory maintains saturation, the output of the network is no longer sensitive to the order of presentation of feature vectors within that segment. Thus each feature vector is regarded as independent, similar to a Hidden Markov Model state. These observations were verified for small recurrent networks trained to classify speech data into voiced and unvoiced segments, and trained to classify trajectories of feature vectors with opposite rotations, which were generated by two vector AR processes.

The observations of this chapter have several implications. The switching delay effect implies that the order of presentation of the training classes is important. It may also explain the fall in performance of recurrent networks at higher frame rates that was reported by Renals, Hochberg & Robinson (1994). At higher frame rates, there are more feature vectors for a given class duration but the feature vectors vary much less on a frame to frame basis than at a lower frame rate, causing saturation of the recurrent network.

A further implication concerns the interpretation of network outputs in classification. Santini & Del Bimbo (1995) have shown that the outputs of a recurrent architecture, trained with fixed class targets and a mean-squared error criterion, approximate the posterior probability of class membership dependent on the entire history of input feature vectors up to the current time. This interpretation may no longer be valid for long duration classes in which saturation of hidden units is maintained, because the network output becomes insensitive to the order of presentation of feature vectors.

Chapter 6

Conclusions and Further Work

6.1 Conclusions

6.1.1 Vocal Tract Modelling

Part I of this dissertation presented a study of parametric modelling of the vocal tract. Two novel contributions arising from this work are

- a framework, using linear black-box models of the vocal tract, to generate high quality synthetic speech, which retains naturalness when the pitch of the excitation is varied from that of the original speech.
- a study of neural network based speech synthesis, including methods for initialising specific neural network architectures from linear models of the vocal tract, and a new architecture which is analogous to a parallel formant implementation of the vocal tract model, and allows initialisation from ARX models.

ARX models, excited by a stylised representation of glottal volume velocity waveforms, produced synthetic speech of very high quality. This was attributed, in part, to the effect of auditory masking in reducing the perceived loudness of the signal distortion. For ARX models, minimisation of the mean squared equation error imposes the constraint that the spectrum of the synthesis error match the formant structure of the synthetic speech. The noise is concentrated into the formant regions of the synthetic signal where it is effectively masked. Using the ARX framework, it was shown possible to manipulate the pitch of the synthetic speech, without loss of quality. The pitch manipulation capabilities were demonstrated for a number of utterances and the performance was shown to be comparable to that of the PSOLA technique, which is typically used in state-of-the-art speech synthesis systems.

With linear OE models and neural network models, objective criteria, such as synthesis SNR, were improved over ARX models, but the resulting synthetic speech was demonstrated to be inferior in quality. This was attributed to the convergence of the learning algorithm to local minima, which resulted in abrupt changes in perceptually significant features, such as formant positions and bandwidths. It was shown that regularization could be used to impose continuity constraints on parameter values for adjacent frames and that this improved the quality of synthetic speech from these models.

For neural network models, large improvements in synthesis SNR over linear ARX models were achieved on some frames of data, which indicates the potential benefit of using a nonlinear framework. However, it was shown that the average improvement in synthesis SNR over all frames of utterances was low, and as a result, the quality of synthetic speech from the existing nonlinear framework was poor.

Several disadvantages of this implementation for vocal tract modelling are:

- simplified representation of vocal tract excitation
- sensitivity of model parameter estimation to misalignment between the instants of glottal closure in the excitation signal and the true instants of glottal closure in the original speech
- the use of an external signal from a laryngograph to locate instants of glottal closure.

6.1.2 Classification of Speech Patterns

The popularity of recurrent neural networks for context-dependent classification tasks, such as those which arise in speech processing, is due to their potential to integrate the context of a feature vector into the current class decision. Part II of this dissertation studied the operation of small recurrent architectures on two-class problems and showed that the main effects of including recurrent connections at hidden units are:

- formation of moving decision boundaries
- switching delay at class boundaries
- smoothing of output decisions for sequences of feature vectors from within the same class.

In theory, movement of decision boundaries gives context-dependent classifications, because the class decision for a feature vector is made with respect to current boundary positions, which are determined by the trajectory of previous feature vectors. However, it was shown that this is not the case when hidden units operate in saturation for prolonged periods of time. Saturation of hidden units results in the following:

- context-sensitive and context-insensitive regions of the feature space
- limited sensitivity to the order of presentation of feature vectors.

This has implications for the ability of the network to perform context-dependent classification of feature vectors within a class and for interpretation of the outputs as estimates of posterior probabilities dependent on the previous history of feature vectors. Saturation of hidden units during presentation of long sequences of feature vectors from the same class is a direct result of training networks with constant class targets. Training networks with ramp-like class targets was shown to reduce insensitivity to within-class context by forcing hidden units to operate out of saturation. For speech applications, such as phone recognition, non-constant target functions may be perceptually motivated as reflecting an increasing confidence in the identity of a phone as more information becomes available.

Typical operation of small recurrent architectures was illustrated for two-class classification of speech utterances into voiced and unvoiced segments, and for a synthetic two-class problem, in which trajectories of vectors generated by AR processes were classified by their clockwise or anti-clockwise rotation.

6.2 Further Work

6.2.1 Linear Models of the Vocal Tract

Evaluation of Relative Loudness of Synthesis Error

Based on a mathematical formulation of hearing, Schroeder, Atal & Hall (1979*b*) have derived a measure of the relative loudness of noise in the presence of a masking signal. The measure is defined in terms of the spectra of the noise and masking signal. In speech synthesis, the noise represents the synthesis error and the masking signal is the synthetic speech. The spectrum of the synthetic speech can be calculated from the frequency response of the vocal tract model and the spectrum of the vocal tract excitation. For a pulse based representation of the vocal tract excitation of voiced speech, the spectrum of the excitation is dependent on the pitch of the excitation and the shape of pulse and can be calculated. Assuming that parameter estimation results in prediction and synthesis errors with zero mean, Gaussian distributions, with measurable variances, the spectrum of the synthesis error for ARX and OE models can be approximated. Thus, an estimate of the relative loudness of the synthesis error for each model can be calculated and may be useful in estimating how significant the effect of auditory masking is to the perceived quality of the synthetic speech generated by each family of models.

Improving Perceptual Quality of Synthesis

The real vocal tract system is more complex than the models used for identification. For a finite model order, the transfer function estimates from all families of models are biased, in the sense that the true underlying system is not within the family of models under consideration. The distribution of bias in the estimation of the transfer function is determined by the frequency bias function, which is affected by a number of factors. The dependence of the frequency bias function on the spectrum of the vocal tract excitation, the model noise, and the choice of pre-filter was considered in chapter 3. The frequency bias function also depends on the sampling interval and prediction horizon (Wahlberg & Ljung 1986). Investigation of the effects of varying these parameters, with specific relevance for speech applications, would be of interest. Further study of the optimal forms for pre-filter, synthesis error spectrum and frequency bias of transfer function estimate for speech synthesis applications would also be of potential benefit for improving the perceptual quality of synthetic speech generated from linear black-box models.

Bayesian Evidence Framework for Estimation of Model Parameters

For OE and neural network models, regularization of the mean-squared error criterion was used to enforce continuity between parameters of adjacent frames. For ARX and LP models, the regularizing term used in this work results in Maximum a Posteriori estimates (MAP) of the model parameters, assuming a Gaussian prior distribution with mean $\boldsymbol{\theta}^0$ and variance I/α, where $\boldsymbol{\theta}^0$ is the initial estimate of the model parameters, and α is the regularizing constant (Lim & Oppenheim 1978, Saleh et al. 1994). Saleh et al. (1994) used a regularizing term of this form for the estimation of linear prediction parameters, using a Bayesian evidence framework to optimally select model order, based on the evidence for models of different orders. Although a Bayesian evidence framework would increase the complexity of the parameter estimation procedure, its use for estimation of ARX parameters in this application would allow the variances of prior distributions, and the relative importance of the priors with respect to the data, to be optimised along with the model parameters. In addition, it would allow selection of the optimal model order for each speech segment.

Text-to-speech Synthesis

The natural sounding synthetic speech generated by ARX models and the ability to manipulate the pitch of the synthetic speech without severe distortion makes them potentially more useful than linear prediction models for text-to-speech synthesis applications. Using a pulse based model of the vocal tract excitation is particularly suitable for text-to-speech synthesis systems because it provides a parametric representation of the excitation which can be stored efficiently, and offers the potential to adapt voicing style by changing the pulse shape (Rutledge et al. 1995). Adaptation of the poles of ARX models (Slifka & Anderson 1995) would also allow modification of speaker type, giving a flexible voice manipulation system. For ARX models to be incorporated into such systems, several relevant areas of further work are the robustness of models to quantisation of parameters and methods for smoothing parameter transitions between concatenated models.

6.2.2 Neural Network Models of the Vocal Tract

Improved Back-propagation Training

Despite improvements in convergence of back-propagation training by using learning rate adaptation, the average improvement in performance over linear models was still low. Investigation of other training methods, such as conjugate-gradients, RProp or Quick-Prop (Schiffmann et al. 1992), may be worthwhile.

Improving Performance on Unvoiced Speech

For unvoiced frames, a codebook of Gaussian excitation sequences was used as excitation. To reduce computation, the codebook entry was selected based on minimising the mean-squared synthesis error from the initial network. Using a larger codebook, and re-optimising the codebook entry after network training would improve the performance on unvoiced data.

Reducing Synthesis Noise for Recurrent Networks

The synthetic speech generated from recurrent network models was perceptually very noisy. Suitable shaping of the spectrum of the synthesis error may permit the perceived loudness of the noise to be reduced, by exploiting the effect of auditory masking. The synthesis error spectrum can be altered by applying a perceptual weighting filter to the error which is minimised in back-propagation. The ideal perceptual weighting filter has a spectrum which is the inverse of the masking threshold, and is related to the spectrum of the synthetic speech. For linear models of the vocal tract, the spectrum of synthetic speech can be calculated from the model parameters. For nonlinear models, there is no equivalent concept of a transfer function. The spectrum of the synthetic speech cannot be derived from the model parameters, and an alternative method for calculating a perceptual weighting filter is needed. Schroeder et al. (1979*b*) and Drogo De Iacovo, Montagna & Sereno (1990) have developed approximate masking thresholds for speech. These could be used to derive a weighting filter with a frequency response that approximates the inverse of the masking threshold of synthetic speech (Drogo De Iacovo & Montagna 1991). Such filters may prove effective in reducing the noise of the synthesis error for recurrent neural networks.

Quantisation of Network Parameters

The effects of quantisation of network weights, applicable for speech coding, has not been studied. Wu & Fallside (1992) suggest that direct quantisation of network weights is not suitable, due to its effect on network stability. A possible solution would be to use a codebook of representative network models.

6.2.3 Classification of Speech Patterns

Larger Architectures

The analysis and experimental work of chapter 5 focused on small recurrent architectures. Further work using multiple output, multilayer architectures is needed, to evaluate the significance of this work for the large architectures typically used for speech recognition. Studies of the minimum class duration which permits switching, the maximum class duration for which trajectory-sensitivity can be maintained, and the implications of these for the choice of frame rate used in speech processing, are of potential interest.

Statistical Interpretation of Network Outputs

Under certain assumptions, the outputs of a feed-forward network trained with constant class targets have been shown to approximate the posterior probabilities of class membership (Richard & Lippman 1991, Ruck et al. 1990, Wan 1990). Santini & Del Bimbo (1995) have extended these proofs to the outputs of a recurrent architecture, showing that the outputs approximate the posterior probability of class membership, dependent on the entire history of input feature vectors up to the current time. The statistical interpretation of the outputs of networks trained with variable class targets has not been studied.

Statistical Interpretation of Recurrent Network Weights

A statistical interpretation of the feedback weights of a single hidden layer recurrent network architecture would be useful. For a single hidden unit with a recurrent connection, the effect of the feedback weight is to bias the current decision towards that of the previous decision. This is similar to the way in which the log prior ratio biases the decision of a Bayes optimal discriminant function towards the most probable class (Duda & Hart 1973). This suggests that recurrent connections may encode the prior probability of classes, or class trajectories, in some way. Possible directions of investigation may lie in the links between Hidden Markov Models and neural networks (Bourlard & Wellekens 1990) or in the application of compound decision theory (Abend 1968, Raviv 1967) to recurrent network architectures.

Adaptation of Networks to Different Data Sets

When there is a mismatch between the class priors for training and test data sets, a network may not be suitable for use on the test data because the network weights inherently encode the class priors (Webb & Lowe 1990). Richard & Lippman (1991) accounted for the variation in priors by scaling network outputs and recent work on feed-forward networks has shown that adjustment of the output biases is sufficient to adapt the classifier to new data (Gish & Siu 1994). If a link between recurrent weights and class priors exists, this suggests that modifications to both recurrent connection weights and biases are necessary to adapt a recurrent network to a new data set with different class priors.

Appendix A

Back-propagation Training For Multi-Delay Recurrent Neural Networks

The following algorithm is for back-propagation training of single output, multi-delay, single hidden layer recurrent neural networks. The architecture has no inter-connections between nodes in the hidden layer and the network input is given by Eqn. (4.5). Fig. 4.7 shows the architecture RNN3, which corresponds to a specific case, where $n_b = 2$ and the total number of delays around hidden units is $n_q = 2$.

The error criterion minimised by the algorithm is the mean-squared (synthesis) error between target values, $y(t)$, and network outputs, $\hat{y}(t)$

$$\begin{aligned} E &= \frac{1}{2}\sum_{t=0}^{N-1} e^2(t) && \text{(A.1)} \\ &= \frac{1}{2}\sum_{t=0}^{N-1} (y(t) - \hat{y}(t))^2 && \text{(A.2)} \end{aligned}$$

Grouping all weights into a parameter vector $\boldsymbol{\theta} = \{\boldsymbol{U}, \boldsymbol{V}, \boldsymbol{W}\}$, the current vector, $\boldsymbol{\theta}^i$, is updated according to

$$\boldsymbol{\theta}^{i+1} = \boldsymbol{\theta}^i + \Delta\boldsymbol{\theta}^i \qquad \text{(A.3)}$$

The update direction is determined by the gradient $dE/d\boldsymbol{\theta}^i$, and the step-size is given by the learning rate, η.

$$\Delta\boldsymbol{\theta}^i = -\eta\frac{dE}{d\boldsymbol{\theta}^i} \qquad \text{(A.4)}$$

$$\begin{aligned}\frac{dE}{d\boldsymbol{\theta}^i} &= \sum_{t=0}^{N-1} e(t)\frac{de(t)}{d\boldsymbol{\theta}^i} \\ &= -\sum_{t=0}^{N-1} e(t)\frac{d\hat{y}(t)}{d\boldsymbol{\theta}^i} \qquad \text{(A.5)}\end{aligned}$$

For stochastic update of weights, after every training pair $[y(t), \boldsymbol{x}(t)]$, the weight update is given by

$$\Delta\boldsymbol{\theta}^t = \eta e(t)\frac{d\hat{y}(t)}{d\boldsymbol{\theta}^t} \qquad \text{(A.6)}$$

The network output, $\hat{y}(t)$, is given by

$$\begin{aligned}\hat{y}(t) &= \sum_{j=1}^{p} u_j h_j(t) \qquad \text{(A.7)} \\ h_j(t) &= f(\sigma_j(t)) \qquad \text{(A.8)} \\ \sigma_j(t) &= \sum_{k=0}^{n_b-1} v_{jk}x(t-k) + \sum_{q=1}^{n_q} w_{jq}h_j(t-q) \qquad \text{(A.9)}\end{aligned}$$

where $f(.)$ is the nonlinear function at the hidden nodes. Superscripts t on weights will be neglected in the following, for simplicity. For output weight u_m,

$$\begin{aligned}\frac{\partial\hat{y}(t)}{\partial u_m} &= h_m(t) \qquad \text{(A.10)} \\ \Delta u_m &= \eta e(t)h_m(t) \qquad \text{(A.11)}\end{aligned}$$

For input weight v_{mn},

$$\begin{aligned}\frac{\partial\hat{y}(t)}{\partial v_{mn}} &= u_m\frac{\partial h_m(t)}{\partial v_{mn}} \qquad \text{(A.12)} \\ \frac{\partial h_m(t)}{\partial v_{mn}} &= g_m\left[x(t-n) + \sum_{q=1}^{n_q} w_{mq}\frac{\partial h_m(t-q)}{\partial v_{mn}}\right] \qquad \text{(A.13)} \\ \frac{\partial h_j(t)}{\partial v_{mn}} &= 0 \qquad\qquad j \neq m \qquad \text{(A.14)}\end{aligned}$$

$$\Delta v_{mn} = \eta e(t) u_m g_m \left[x(t-n) + \sum_{q=1}^{n_q} w_{mq} \frac{\partial h_m(t-q)}{\partial v_{mn}} \right] \tag{A.15}$$

where $g_m = \partial f(\sigma_m)/\partial \sigma_m$ and σ_m is the activation at node m.

For feedback weights, w_{rs},

$$\frac{\partial \hat{y}(t)}{\partial w_{rs}} = u_r \frac{\partial h_r(t)}{\partial w_{rs}} \tag{A.16}$$

$$\frac{\partial h_r(t)}{\partial w_{rs}} = g_r \left[h_r(t-s) + \sum_{q=1}^{n_q} w_{rq} \frac{\partial h_r(t-q)}{\partial w_{rs}} \right] \tag{A.17}$$

$$\frac{\partial h_j(t)}{\partial w_{rs}} = 0 \qquad\qquad j \neq r \tag{A.18}$$

$$\Delta w_{rs} = \eta e(t) u_r g_r \left[h_r(t-s) + \sum_{q=1}^{n_q} w_{rq} \frac{\partial h_r(t-q)}{\partial w_{rs}} \right] \tag{A.19}$$

Appendix B

Tape Demonstration

Evaluation of the performance of vocal tract models was based on subjective assessments of the quality of synthetic speech produced by different models. To allow the reader to verify the reported performance results, an audio tape is included with this thesis. The following outlines the syntheses that will be heard.

B.1 Introduction

This tape demonstrates the perceptual quality of synthetic speech generated by the different types of model studied in this thesis. First, you will hear a comparison of the synthetic speech produced by linear black-box models.

Then, you will hear syntheses from ARX and LP models, in which the pitch contours are altered from that of the original speech. These will be compared with syntheses using the pitch-synchronous overlap-add method (PSOLA).

Finally, you will hear a demonstration of the synthetic speech generated by nonlinear neural network models.

B.2 Comparison of Different Linear Models

The following examples demonstrate the performance of linear black-box models for synthesising speech at the same pitch as the original utterance.

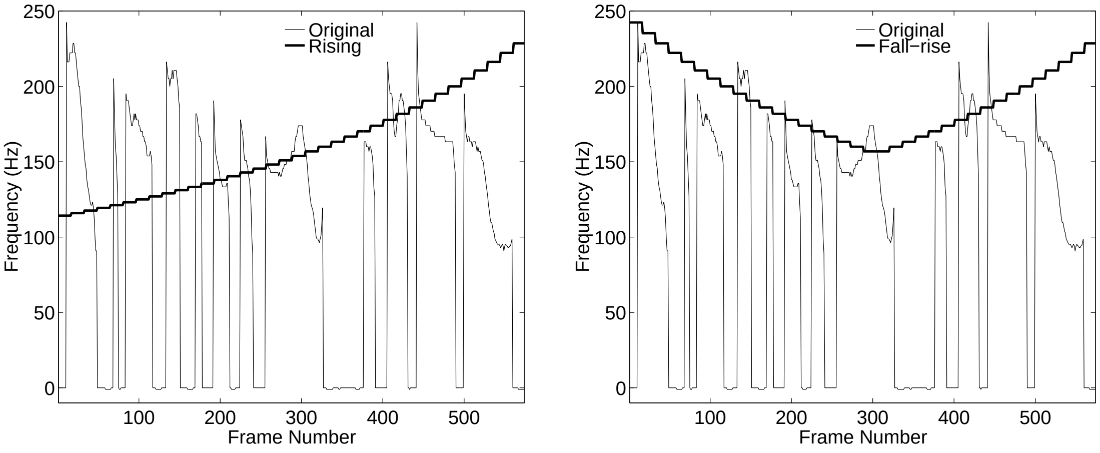
1. ***"Germany's decision followed eight years later"***
    - Original Speech
    - LP ($n_a = 12$, covariance analysis, with pre-emphasis)
    - ARX ($n_a = 10$, $n_b = 2$, input $dx(t)$, with pre-emphasis)
    - OE ($n_a = 10$, $n_b = 2$, input $dx(t)$, with pre-emphasis)
    - OE with continuity constraints ($n_a = 10$, $n_b = 2$, input $dx(t)$, with pre-emphasis)

B.3 Pitch Manipulation using ARX and LP Models

Now you will hear syntheses of several utterances by ARX models, in which new pitch contours are applied to the synthetic speech. For comparison, the syntheses from linear prediction (LP) models and the pitch synchronous overlap-add method will also be heard.

1. ***"France became the first decimal country in Europe, in 1799"***

   - Original Speech
   - ARX at original pitch
   - LP at original pitch
   - PSOLA, rising pitch contour, Fig. B.1(a)
   - ARX, rising pitch contour
   - LP, rising pitch contour
   - PSOLA, fall-rise pitch contour, Fig. B.1(b)
   - ARX, fall-rise pitch contour
   - LP, fall-rise pitch contour



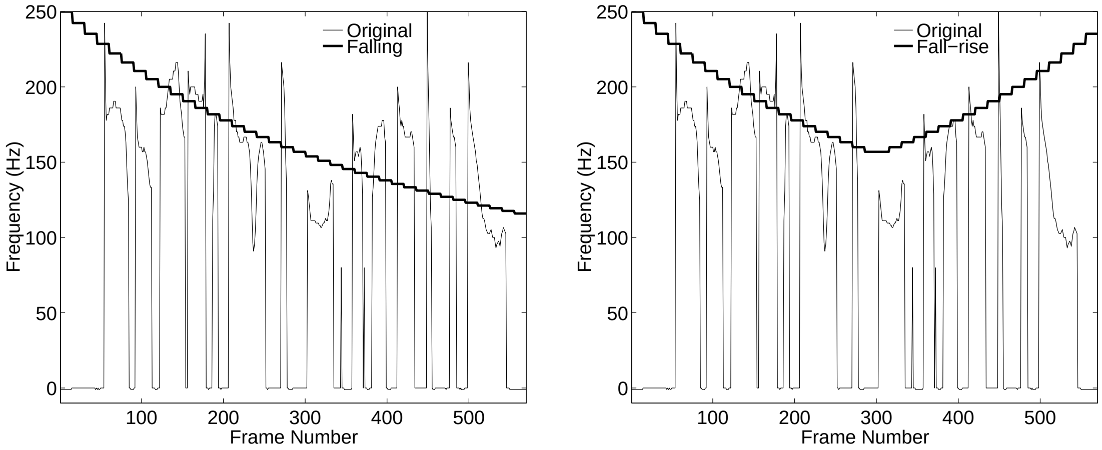
(a)

(b)

Figure B.1: Pitch contours applied to the utterance *"France became the first decimal country in Europe, in 1799"*.

2. *" . . . joined by Belgium, Italy and Switzerland, in 1865"*

    - Original Speech
    - ARX at original pitch
    - LP at original pitch
    - PSOLA, falling pitch contour, Fig. B.2(a)
    - ARX, falling pitch contour
    - LP, falling pitch contour
    - PSOLA, fall-rise pitch contour, Fig. B.2(b)
    - ARX, fall-rise pitch contour
    - LP, fall-rise pitch contour



(a)

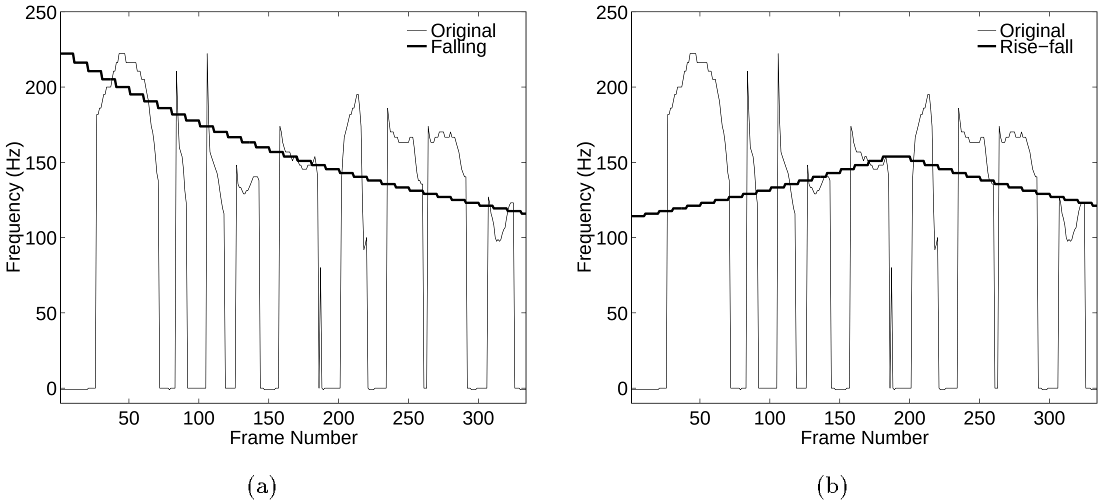
(b)

Figure B.2: Pitch contours applied to the utterance *" . . . joined by Belgium, Italy and Switzerland, in 1865"*.

3. *" Germany's decision followed eight years later"*

    - Original Speech
    - ARX at original pitch
    - LP at original pitch
    - PSOLA, falling pitch contour, Fig. B.3(a)
    - ARX, falling pitch contour
    - LP, falling pitch contour

- PSOLA, rise-fall pitch contour, Fig. B.3(b)
- ARX, rise-fall pitch contour
- LP, rise-fall pitch contour



(a)

(b)

Figure B.3: Pitch contours applied to the utterance *"Germany's decision followed eight years later"*.

B.4 Neural Network Models of the Vocal Tract

The following syntheses illustrate the different performance from feed-forward and recurrent neural network models of the vocal tract.

1. ***"France became the first decimal country in Europe, in 1799"***

    - Original Speech
    - ARX ($n_a = 10$, $n_b = 2$, input $dx(t)$, no pre-emphasis)
    - FNN ($n_h = 2$, $n_i = 13$, initial ARX models as above)
    - FNN with regularization ($n_h = 2$, $n_i = 13$, initial ARX models as above)
    - RNN3 ($n_h = 5$, $n_i = 2$, initial ARX models as above)
    - RNN3 with regularization ($n_h = 5$, $n_i = 2$, initial ARX models as above)

2. ***"Germany's decision followed eight years later"***

    - Original Speech
    - ARX ($n_a = 10$, $n_b = 2$, input $dx(t)$, no pre-emphasis)
    - FNN ($n_h = 2$, $n_i = 13$, initial ARX models as above)
    - FNN with regularization ($n_h = 2$, $n_i = 13$, initial ARX models as above)
    - RNN3 ($n_h = 5$, $n_i = 2$, initial ARX models $n_a = 10$, $n_b = 9$, input $dx(t)$, no pre-emphasis)
    - RNN3 with regularization ($n_h = 5$, $n_i = 2$, initial ARX models $n_a = 10$, $n_b = 9$, input $dx(t)$, no pre-emphasis)

Bibliography

Abend, K. (1968), Compound decision procedures for pattern recognition, *in* L. Kanal, ed., 'Pattern Recognition', Thomson Book Co., Washington D.C., pp. 207–249.

Alku, P. (1992), An automatic method to estimate the time-based parameters of the glottal pulseform, *in* 'Proceedings of the IEEE International Conference on Acoustics, Speech, and Signal Processing', Vol. 2, San Francisco, California, USA, pp. II–29–II–32.

Allen, J., Hunnicutt, M. & Klatt, D. (1987), *From Text to Speech: The MITalk System*, Cambridge University Press.

Ananthapadmanabha, T. & Yegnanarayana, B. (1979), 'Epoch extraction from linear prediction residual for identification of closed glottis interval', *IEEE Transactions on Acoustics, Speech, and Signal Processing* **ASSP-27**(4), 309–319.

Atal, B. & Remde, J. (1982), A new model of LPC excitation for producing natural-sounding speech at low bit rates, *in* 'Proceedings of the IEEE International Conference on Acoustics, Speech, and Signal Processing', Vol. 1, Paris, France, pp. 614–617.

Atal, B. & Schroeder, M. (1979), 'Predictive coding of speech signals and subjective error criteria', *IEEE Transactions on Acoustics, Speech, and Signal Processing* **ASSP-27**(3), 247–254.

Back, A. & Tsoi, A. C. (1991*a*), Analysis of hidden layer weights in a dynamic locally recurrent network, *in* T. Kohonen, K. Makisara, O. Simula & J. Kangas, eds, 'Artificial Neural Networks', Vol. 1, Elsevier Science Publishers B. V. (North Holland), pp. 961–966.

Back, A. & Tsoi, A. C. (1991*b*), 'FIR and IIR synapses, a new neural network architecture for time-series modelling', *Neural Computation* **3**, 375–385.

Baer, T., Löfqvist, A. & McGarr, N. (1983), 'Laryngeal vibrations: A comparison between high-speed filming and glottographic techniques', *Journal of the Acoustical Society of America* **73**(4), 1304–1308.

Barton, S. (1991), 'A matrix method for optimizing a neural network', *Neural Computation* **3**(3), 450–459.

Billings, S. & Voon, W. (1986), 'Correlation based model validity tests for non-linear models', *International Journal of Control* **44**(1), 235–244.

Billings, S., Jamaluddin, H. & Chen, S. (1992), 'Properties of neural networks with applications to modelling non-linear dynamical systems', *International Journal of Control* **55**(1), 193–224.

Borden, G. & Harris, K. (1984), *Speech Science Primer*, Williams and Wilkins, USA.

Bourland, H. & Kamp, Y. (1988), 'Auto-association by multilayer perceptrons and singular value decomposition', *Biological Cybernetics* **59**, 291–294.

Bourlard, H. (1991), Neural nets and Hidden Markov Models: Review and generalizations, *in* 'Proceedings of the 2nd European Conference on Speech Communication and Technology', Vol. 2, Genova, Italy, pp. 363–369.

Bourlard, H. & Wellekens, C. (1990), 'Links between Markov models and multilayer perceptrons', *IEEE Transactions on Pattern Analysis and Machine Intelligence* **12**(12), 1167–1178.

Bridle, J. (1989), Probabilistic interpretation of feedforward classification network outputs, with relationships to statistical pattern recognition, *in* F. Fogelman Soulié & J. Hérault, eds, 'Neurocomputing: Algorithms, Architectures and Applications', Vol. F 68 of *NATO ASI Series*, Springer-Verlag, Berlin Heidelberg, pp. 227–236.

Bridle, J. (1990), 'ALPHA-NETS: A recurrent neural network architecture with a hidden Markov model interpretation', *Speech Communication* **9**(1), 83–92.

Burrows, T. (1992), Non-linear time series analysis of speech signals, Dissertation for M.Phil. in Computer Speech and Language Processing, Cambridge University Engineering Department, Cambridge, England.

Burrows, T. & Niranjan, M. (1993), The use of feed-forward and recurrent neural networks for system identification, Technical Report CUED/F-INFENG/TR158, Cambridge University Engineering Department, Cambridge, England.

Burrows, T. L. & Niranjan, M. (1994), The use of recurrent neural networks for classification, *in* J. Vlontzos, J.-N. Hwang & E. Wilson, eds, 'Proc. IEEE Workshop on Neural Networks for Signal Processing IV', Greece, pp. 117–125.

Burrows, T. L. & Niranjan, M. (1995), Vocal tract modelling with recurrent neural networks, *in* 'Proceedings of the IEEE International Conference on Acoustics, Speech, and Signal Processing', Vol. 5, Detroit, Michigan, USA, pp. 3315–3318.

Chandra, S. & Lin, W. (1974), 'Experimental comparison between stationary and nonstationary formulations of linear prediction applied to voiced speech analysis', *IEEE Transactions on Acoustics, Speech, and Signal Processing* **ASSP-22**(6), 403–415.

Charpentier, F. & Stella, M. (1986), Diphone synthesis using an overlap-add technique for speech waveforms concatenation, *in* 'Proceedings of the IEEE International Conference on Acoustics, Speech, and Signal Processing', Vol. 3, Tokyo, Japan, pp. 2015–2018.

Chen, S., Billings, S. & Grant, P. (1990), 'Non-linear system identification using neural networks', *International Journal of Control* **51**(6), 1191–1214.

Cheng, Y. & O'Shaughnessy, D. (1989), 'Automatic and reliable estimation of glottal closure instant and period', *IEEE Transactions on Acoustics, Speech, and Signal Processing* **37**(12), 1805–1815.

Cheng, Y. & O'Shaughnessy, D. (1993), 'On 450-600 b/s natural sounding speech coding', *IEEE Transactions on Speech and Audio Processing* **1**(2), 207–219.

Cybenko, G. (1989), 'Approximation by superpositions of a sigmoidal function', *Math. Contr. Signals. Syst.* **2**, 303–314.

Deller, Jr., J., Proakis, J. & Hansen, J. (1993), *Discrete-time Processing of Speech Signals*, Macmillan Publishing Company, Englewood Cliffs, N.J.

Denzler, J., Kompe, R., Kießling, A., Niemann, H. & Nöth, E. (1993), Going back to the source: Inverse filtering of the speech signal with ANNs, *in* 'Proceedings of the 3rd European Conference on Speech Communication and Technology', Vol. 1, Berlin, Germany, pp. 111–114.

Drogo De Iacovo, R. & Montagna, R. (1991), Some experiments in perceptual masking of quantizing noise in analysis-by-synthesis speech coders, *in* 'Proceedings of the 2nd European Conference on Speech Communication and Technology', Vol. 2, Genova,Italy, pp. 825–828.

Drogo De Iacovo, R., Montagna, R. & Sereno, D. (1990), Vector quantization and perceptual criteria in SVD-based CELP coders, *in* 'Proceedings of the IEEE International Conference on Acoustics, Speech, and Signal Processing', Vol. 1, Albuquerque, New Mexico, USA, pp. 33–36.

Duda, R. O. & Hart, P. E. (1973), *Pattern Classification and Scene Analysis*, Wiley, New York.

Elman, J. (1990), 'Finding structure in time', *Cognitive Science* **1**(14), 179–211.

Etemad, K. (1993), Phoneme recognition based on multi-resolution and non-causal context, *in* C. A. Kamm, G. M. Kuhn, B. Yoon, R. Chellappa & S. Y. Kung, eds, 'Proc. 1993 IEEE Workshop on Neural Networks for Signal Processing', pp. 343–352.

Fahlman, S. (1988), An empirical study of learning speed in back-propagation networks, Technical Report CMU-CS-88-162, CMU.

Fant, G. (1960), *Acoustic Theory of Speech Production*, Mouton, The Hague.

Fant, G. (1979), Glottal source and excitation analysis, Technical Report STL-QPSR 1/1979, Speech Transmission Lab.

Fant, G., Liljencrants, J. & Lin, Q. (1985), A four-parameter model of glottal flow, Technical Report STL-QTSR 4/1985, Speech Transmission Lab.

Flanagan, J. (1972), *Speech Analysis, Synthesis and Perception*, 2nd edn, Springer-Verlag, New York.

Fourcin, A. & Abberton, E. (1971), 'First applications of a new laryngograph', *Medical and Biological Illustration* **21**, 172–182.

Frasconi, P., Gori, M. & Soda, G. (1992), 'Local feedback multilayer networks', *Neural Computation* 4, 120–130.

French, N. & Steinberg, J. (1947), 'Factors governing the intelligibility of speech sounds', *Journal of the Acoustical Society of America* **19**, 90–119.

Fujisaki, H. & Ljungqvist, M. (1986), Proposal and evaluation of models for the glottal source waveform, *in* 'Proceedings of the IEEE International Conference on Acoustics, Speech, and Signal Processing', Tokyo, Japan, pp. 1605–1608.

Fujisaki, H. & Ljungqvist, M. (1987), Estimation of voice source and vocal tract parameters based on arma analysis and a model for the glottal source waveform, *in* 'Proceedings of the IEEE International Conference on Acoustics, Speech, and Signal Processing', Vol. 1, Dallas,Texas, pp. 637–640.

Funaki, K. & Mitome, Y. (1990), A speech analysis method based on a glottal source model, *in* 'International Conference on Spoken Language Processing', Vol. 1, Kobe, Japan, pp. 45–48.

Gallinari, P., Thiria, S., Badran, F. & Fogelman Soulié, F. (1991), 'On the relations between discriminant analysis and multilayer perceptrons', *Neural Networks* 4, 349–360.

Garofolo, J. (1988), *Getting Started with the DARPA TIMIT CD-ROM: An Acoustic Phonetic Continuous Speech Database*, National Institute of Standards and Technology NIST, Gaithersburgh, M.D.

Gerson, I. & Jasiuk, M. (1992), 'Techniques for improving the performance of CELP-type speech coders', *IEEE Journal on Selected Areas in Communications* **10**(5), 858–865.

Giles, C., Miller, C., Chen, D., Chen, H., Sun, G. & Lee, Y. (1992), 'Learning and extracting finite state automata with second-order recurrent neural networks', *Neural Computation* 4, 393–405.

Gish, H. & Siu, M. (1994), An invariance property of neural networks, *in* 'Proceedings of the IEEE International Conference on Acoustics, Speech, and Signal Processing', pp. 541–544.

Goggin, S., Gustafson, K. & Johnson, K. (1991), An asymptotic singular value decomposition analysis of nonlinear multilayer neural networks, *in* 'International Joint Conference on Neural Networks', Vol. 1, pp. 785–789.

Grice, M. & Barry, W. (1989), Extension phase final report, Multi-lingual speech input/output : Assessment, methodology and standardization, Technical Report ESPRIT project 1541 (SAM), University College, London.

Hamon, C., Moulines, E. & Charpentiers, F. (1989), A diphone synthesis system based on time-domain prosodic modifications of speech, *in* 'Proceedings of the IEEE International Conference on Acoustics, Speech, and Signal Processing', Vol. 1, Glasgow, Scotland, pp. 238–241.

Hanes, M., Ahalt, S. & Krishnamurthy, A. (1994), 'Acoustic-to-phonetic mapping using recurrent neural networks', *IEEE Transactions on Neural Networks* **5**(4), 659–662.

Hansen, J. & Clements, M. (1994), 'Constrained iterative speech enhancement with application to speech recognition', *IEEE Transactions on Signal Processing* **39**(4), 795–805.

Hedelin, P. (1984), A glottal LPC-vocoder, *in* 'Proceedings of the IEEE International Conference on Acoustics, Speech, and Signal Processing', Vol. 1, San Diego, California, pp. 1.6.1–1.6.4.

Hirschauer, P., Larzabal, P. & Clergeot, H. (1994), 'Parametric estimation with neural networks : Taking the nonlinearity into account for backpropagation, initialization and network size', *IMACS International Symposium on Signal Processing, Robotics and Neural Networks SPRANN'94*.

Holmes, J. (1973), 'The influence of glottal waveform on the naturalness of speech from a parallel formant synthesizer', *IEEE Transactions on Audio and Electroacoustics* **AU-21**(3), 298–305.

Holmes, J. (1983), 'Formant synthesizers : Cascade or parallel', *Speech Communication* **2**, 251–273.

Howard, D. & Breen, A. (1989), Methods for dynamic excitation control in parallel formant speech synthesis, *in* 'Icassp89', Vol. 1, Glasgow, Scotland, pp. 215–218.

Iso, K. & Watanabe, T. (1990), Speaker-independent word recognition using a neural prediction model, *in* 'Proceedings of the IEEE International Conference on Acoustics, Speech, and Signal Processing', Vol. 1, Albuquerque, New Mexico, USA, pp. 441–444.

Jordan, M. (1986), Serial order: A parallel distributed processing approach, Technical Report 8604, UCSD.

Kershaw, D., Hochberg, M. & Robinson, A. (1995), Context-dependent classes in a hybrid recurrent network-HMM speech recognition system, Technical Report CUED/F-INFENG/TR217, Cambridge University Engineering Department, Cambridge, England, Cambridge, England.

Kiritani, S., Honda, K., Imagawa, H. & Hirose, H. (1986), Simultaneous high-speed digital recording of vocal fold vibration and speech signal, *in* 'Proceedings of the IEEE International Conference on Acoustics, Speech, and Signal Processing', Tokyo, Japan, pp. 1633–1636.

Klatt, D. (1980), 'Software for cascade/parallel formant synthesizer', *Journal of the Acoustical Society of America* **67**(3), 971–995.

Klatt, D. (1987), 'Review of text-to-speech conversion for english', *Journal of the Acoustical Society of America* **82**(3), 737–793.

Krishnamurthy, A. & Childers, D. (1986), 'Two-channel speech analysis', *IEEE Transactions on Acoustics, Speech, and Signal Processing* **ASSP-34**(4), 730–743.

Kroon, P. & Atal, B. (1988), Strategies for improving the performance of CELP coders at low bit rates, *in* 'Proceedings of the IEEE International Conference on Acoustics, Speech, and Signal Processing', Vol. 1, New York, USA, pp. 151–154.

Ku, C. & Lee, K. (1995), 'Diagonal recurrent neural networks for dynamic systems control', *IEEE Transactions on Neural Networks* **6**(1), 144–155.

Kulkarni, A. (1991), 'Solving ill-posed problems with artificial neural networks', *Neural Networks* **4**, 477–484.

Lee, C. (1988), 'On robust linear prediction of speech', *IEEE Transactions on Acoustics, Speech, and Signal Processing* **36**(5), 642–650.

Lim, J. & Oppenheim, A. (1978), 'All-pole modeling of degraded speech', *IEEE Transactions on Acoustics, Speech, and Signal Processing* **ASSP-26**(3), 197–210.

Linggard, R. (1985), *Electronic synthesis of speech*, Cambridge University Press, Cambridge.

Lippmann, R. & Gold, B. (1987), Neural-net classifiers useful for speech recognition, *in* 'Proceedings of the 1st International Conference on Neural Networks', Vol. IV, San Diego, CA., pp. 417–425.

Ljung, L. (1985), 'On estimation of transfer functions', *Automatica* **21**(6), 677–696.

Ljung, L. (1987), *System Identification : Theory for the User*, Prentice-Hall, Englewood Cliffs, N.J.

Ljung, L. (1991), *System Identification Toolbox User's Guide*, The MathWorks, Inc., Mass., USA.

Ljung, L. & Sjöberg, J. (1992), A system identification perspective on neural nets, *in* S. Kung, ed., 'Proc. IEEE Workshop on Neural Networks for Signal Processing', IEEE Science Center, N.J.

Lobo, A. & Ainsworth, W. (1992), Evaluation of a glottal ARMA model of speech production, *in* 'Proceedings of the IEEE International Conference on Acoustics, Speech, and Signal Processing', Vol. 2, San Francisco, California, USA, pp. II–13–II–16.

Lowe, D. & Webb, A. (1989), Adaptive networks, dynamical systems, and the predictive analysis of time series, *in* 'Proc. IEE Conference on Neural Networks', London, pp. 95–99.

Makhoul, J. (1975), 'Spectral linear prediction: Properties and applications', *IEEE Transactions on Acoustics, Speech, and Signal Processing* **ASSP-23**(3), 283–296.

Markel, J. & Gray, A. (1976), *Linear Prediction of Speech*, Springer-Verlag.

Markel, J. & Gray, Jr., A. (1974), 'A linear prediction vocoder simulation based upon the autocorrelation method', *IEEE Transactions on Acoustics, Speech, and Signal Processing* **ASSP-22**(2), 124–134.

Mathews, M., Miller, J. & David, Jr., E. (1961), 'Pitch synchronous analysis of voiced sounds', *Journal of the Acoustical Society of America* **33**(2), 179–186.

Milenkovic, P. (1986), 'Glottal inverse filtering by joint estimation of an AR system with a linear input model', *IEEE Transactions on Acoustics, Speech, and Signal Processing* **ASSP-34**(1), 28–41.

Morikawa, H. & Fujisaki, H. (1984), 'System identification of the speech production process based on a State-Space representation', *IEEE Transactions on Acoustics, Speech, and Signal Processing* **ASSP-32**(2), 252–262.

Moulines, E. & Charpentier, F. (1990), Pitch-synchronous waveform processing techniques for text-to-speech synthesis using diphones, *in* 'Speech Communication', Vol. 9, Elsevier Science Publishers B.V., North-Holland, pp. 453–467.

Neal, R. (1995), Bayesian Learning for Neural Networks, PhD thesis, University of Toronto.

Nerrand, O., Roussel-Ragot, P., Urbani, D., Personnaz, L. & Dreyfus, G. (1994), 'Training recurrent neural networks: Why and how? An illustration in dynamical process modeling', *IEEE Transactions on Neural Networks* **5**(2), 178–184.

Ney, H. (1984), 'The use of one-stage dynamic programming algorithm for connected word recognition', *IEEE Transactions on Acoustics, Speech, and Signal Processing* **32**, 263–272.

Nguyen, D. & Widrow, B. (1990), Improving the learning speed of 2-layer neural networks by choosing initial values of the adaptive weights, *in* 'Proceedings of the International Joint Conference on Neural Networks', San Diego, C.A.

Niles, L. (1991), TIMIT phoneme recognition using an HMM-derived recurrent neural network, *in* 'Proceedings of the 2nd European Conference on Speech Communication and Technology', Vol. 2, Genova, Italy, pp. 559–562.

Niranjan, M. (1990), CELP coding with adaptive output-error model identification, *in* 'Proceedings of the IEEE International Conference on Acoustics, Speech, and Signal Processing', Vol. 1, Albuquerque, New Mexico, USA, pp. 225–228.

Pearce, D. & Whitaker, L. (1986), Reference formant analysis, *in* 'International Conference on Speech Input/Output Techniques and Applications', number 258 *in* 'IEE Conference Publication', IEE, pp. 37–42.

Priestley, M. (1988), *Nonlinear and Nonstationary Time Series Analysis*, Academic Press, London.

Psichogios, D. & Ungar, L. (1994), 'SVD-NET: an algorithm that automatically selects network structure', *IEEE Transactions on Neural Networks* **5**(3), 513–515.

Rabiner, L. & Schafer, R. (1978), *Digital Processing of Speech Signals*, Prentice Hall.

Raviv, J. (1967), 'Decision making in markov chains applied to the problem of pattern recognition', *IEEE Transactions on Information Theory* **IT-3**(4), 536–551.

Renals, S., Hochberg, M. & Robinson, T. (1994), Learning temporal dependencies in connectionist speech recognition, *in* J. Cowan, G. Tesauro & J. Alspector, eds, 'Advances in Neural Information Processing Systems 6', Morgan Kaufmann, pp. 1051–1058.

Renals, S., Morgan, N., Bourlard, H., Cohen, M. & Franco, H. (1994), 'Connectionist probability estimators in HMM speech recognition', *IEEE Transactions on Speech and Audio Processing* **2**(1), 161–174.

Richard, M. & Lippman, R. (1991), 'Neural network classifiers estimate Bayesian *a posteriori* probabilities', *Neural Computation* **3**(4), 461–483.

Robinson, A. (1994), 'An application of recurrent nets to phone probability estimation', *IEEE Transactions on Neural Networks* **5**(2), 298–305.

Robinson, A. & Fallside, F. (1987), The utility driven dynamic error propagation network, Technical Report CUED/F-INFENG/TR.1, Cambridge University Engineering Department, Cambridge, England, Cambridge, England.

Robinson, T. & Fallside, F. (1991), 'A recurrent error propagation network speech recognition system', *Computer Speech and Language* **5**(3), 259–274.

Rosenberg, A. (1971), 'Effect of glottal pulse shape on the quality of natural vowels', *Journal of the Acoustical Society of America* **49**(2 (Part 2)), 583–590.

Rubin, P. & Baer, T. (1981), 'An articulatory synthesizer for perceptual research', *Journal of the Acoustical Society of America* **70**(2), 321–328.

Ruck, D., Rogers, S., Kabrisky, M., Oxley, M. & Suter, B. (1990), 'The multilayer perceptron as an approximation to a Bayes optimal discriminant function', *IEEE Transactions on Neural Networks* **1**(4), 296–298.

Rumelhart, D. E., Hinton, G. E. & Williams, R. J. (1986), Learning internal representations by error propagation, *in* D. E. Rumelhart & J. L. McClelland, eds, 'Parallel Distributed Processing: Explorations in the Microstructure of Cognition. Vol. I: Foundations.', Bradford Books/MIT Press, Cambridge, MA, chapter 8, pp. 318–362.

Rutledge, J., Cummings, K., Lambert, D. & Clements, M. (1995), Synthesizing styled speech using the Klatt synthesizer, *in* 'Proceedings of the IEEE International Conference on Acoustics, Speech, and Signal Processing', Vol. 1, Detroit, Michigan, USA, pp. 648–651.

Saleh, G., Niranjan, M. & Fitzgerald, W. (1994), The application of Bayesian inference to linear prediction of speech, Technical Report CUED/F-INFENG/TR.205, Cambridge University Engineering Department, Cambridge, England.

Santini, S. & Del Bimbo, A. (1995), 'Recurrent neural networks can be trained to be maximum a posteriori probability classifiers', *Neural Networks* **8**(1), 25–29.

Schiffmann, W., Joost, M. & Werner, R. (1992), Optimization of the backpropagation algorithm for training multilayer perceptrons, Technical report, University of Koblenz, Germany.

Schroeder, M. & Atal, B. (1985), Code-excited linear prediction (CELP): High-quality speech at very low bit rates, *in* 'Proceedings of the IEEE International Conference on Acoustics, Speech, and Signal Processing', Vol. 3, Tampa, Florida, pp. 937–940.

Schroeder, M., Atal, B. & Hall, J. (1979*a*), Objective measure of certain speech signal degradations based on masking properties of human auditory perception, *in* B. Lindblom & S. Ohman, eds, 'Frontiers of Speech Communication Research', Academic Press, London, pp. 217–229.

Schroeder, M., Atal, B. & Hall, J. (1979*b*), 'Optimizing digital speech coders by exploiting masking properties of the human ear', *Journal of the Acoustical Society of America* **66**(6), 1647–1652.

Sejnowski, T. & Rosenberg, C. (1986), NETtalk: A parallel network that learns to read aloud., Technical Report JHU/EECS-86/01, The Johns Hopkins University.

Silva, F. & Almeida, L. (1990), Speeding up backpropagation, *in* R. Eckmiller, ed., 'Advanced Neural Computers', pp. 151–158.

Singhal, S. & Atal, B. (1983), Optimizing LPC filter parameters for multi-pulse exctiation, *in* 'Proceedings of the IEEE International Conference on Acoustics, Speech, and Signal Processing', Vol. 2, Boston, U.S.A., pp. 781–784.

Sjöberg, J. (1995), Non-Linear System Identification with Neural Networks, PhD thesis, Linköping University, Sweden.

Sjöberg, J. & Ljung, L. (1992), Overtraining, regularization, and searching for minimum in neural networks, *in* 'Preprint 4th IFAC Symposium on Adaptive Systems in Control and Signal Processing', Grenoble, France, pp. 669–674.

Slifka, J. & Anderson, T. (1995), Speaker modification with LPC pole analysis, *in* 'Proceedings of the IEEE International Conference on Acoustics, Speech, and Signal Processing', Vol. 1, Detroit, Michigan, USA, pp. 644–647.

Teager, H. & Teager, S. (1990), Evidence for nonlinear sound production mechanisms in the vocal tract, *in* W. Hardcastle & A. Marchal, eds, 'Speech Production and Speech Modelling', Vol. D 55 of *NATO ASI Series*, Kluwer Academic Publishers, pp. 241–261.

Tebelskis, J. & Waibel, A. (1990), Large vocabulary recognition using linked predictive neural networks, *in* 'Proceedings of the IEEE International Conference on Acoustics, Speech, and Signal Processing', Vol. 1, Albuquerque, New Mexico, USA, pp. 437–440.

Thomson, M. (1992), A new method for determining the vocal tract transfer function and its excitation from voiced speech, *in* 'Proceedings of the IEEE International Conference on Acoustics, Speech, and Signal Processing', Vol. 2, San Francisco, California, USA, pp. II–37–II–40.

Tiño, P., Horne, B. & Giles, C. (1995), Fixed points in two–neuron discrete time recurrent networks: Stability and bifurcation considerations, Technical Report UMIACS-TR-95-51 and CS-TR-3461, Institute for Advance Computer Studies, University of Maryland, College Park, MD 20742.

Tishby, N. (1990), A dynamical systems approach to speech processing, *in* 'Proceedings of the IEEE International Conference on Acoustics, Speech, and Signal Processing', Vol. 1, Albuquerque, New Mexico, USA, pp. 365–368.

Townshend, B. (1991), Nonlinear prediction of speech, *in* 'Proceedings of the IEEE International Conference on Acoustics, Speech, and Signal Processing', Vol. 1, pp. 425–428.

Tuerk, C. & Robinson, T. (1993), Speech synthesis using artificial neural networks trained on cepstral coefficients, *in* 'Proceedings of the 3rd European Conference on Speech Communication and Technology', Vol. 3, Berlin, Germany, pp. 1713–1716.

Wahlberg, B. & Ljung, L. (1986), 'Design variables for bias distribution in transfer function estimation', *IEEE Transactions on Automatic Control* **AC-31**(2), 134–144.

Waibel, A., Hanazawa, T., Hinton, G., Shikano, K. & Lang, K. (1989), 'Phoneme recognition using time-delay neural networks', *IEEE Transactions on Acoustics, Speech, and Signal Processing* **37**(3), 328–339.

Wan, E. (1990), 'Neural network classification: A Bayesian interpretation', *IEEE Transactions on Neural Networks* **1**(4), 303–305.

Wang, R., Guan, Q. & Fujisaki, H. (1990), A method for robust GARMA analysis of speech, *in* 'International Conference on Spoken Language Processing', Vol. 1, Kobe, Japan, pp. 33–36.

Watrous, R. & Shastri, L. (1987), Learning phonetic features using connectionist networks: An experiment in speech recognition, *in* 'Proceedings of 1st International Conference on Neural Networks', pp. 318–388.

Watrous, R., Ladendorf, B. & Kuhn, G. (1990), 'Complete gradient optimization of a recurrent network applied to /b/,/d/,/g/ discrimination', *Journal of the Acoustical Society of America* **87**(3), 1301–1309.

Webb, A. & Lowe, D. (1990), 'The optimised internal representation of multilayer classifier networks performs nonlinear discriminant analysis', *Neural Networks* **3**, 367–375.

Weigend, A. & Rumelhart, D. (1991), Generalization through minimal networks with application to forecasting, *in* E. Keramidas, ed., 'INTERFACE '91– 23rd Symposium on the Interface: Computing Science and Statistics', Interface Foundation of North America, Seattle, WA, pp. 362–370.

Weigend, A. S. & Nix, D. A. (1994), Predictions with confidence intervals (local error bars), Technical Report CU-CS-724-94, Department of Computer Science and Institute of Cognitive Science, University of Colorado, Boulder, CO 80309-0439.

Werbos, P. J. (1990), 'Backpropagation through time: What it does and how to do it', *Proceedings of the IEEE* **78**(10), 1550–1560.

Wessels, L. & Barnard, E. (1992), 'Avoiding false local minima by proper initialization of connections', *IEEE Transactions on Neural Networks* **3**(6), 899–905.

White, H. (1989), 'Learning in artificial neural networks : A statistical perspective', *Neural Computation* **1**, 425–464.

Wigren, T., Bergström, A., Harrysson, S., Jansson, F. & Nilsson, H. (1995), Improvements of background sound coding in linear predictive speech coders, *in* 'Proceedings of the IEEE International Conference on Acoustics, Speech, and Signal Processing', Vol. 1, Detroit, Michigan, USA, pp. 25–28.

Williams, R. J. & Zipser, D. (1989*a*), 'Experimental analysis of the real-time recurrent learning algorithm', *Connection Science* **1**(1), 87–111.

Williams, R. J. & Zipser, D. (1989*b*), 'A learning algorithm for continually running fully recurrent neural networks', *Neural Computation* **1**, 270–280.

Wilson, E. & Tufts, D. (1993), 'Neural network design algorithm and multistage structure', unpublished.

Wu, L. & Fallside, F. (1992), Fully vector quantized neural network-based code-excited non-linear predictive speech coding, Technical Report CUED/F-INFENG/TR.94, Cambridge University Engineering Department, Cambridge, England.

Wu, L. & Fallside, F. (1994), 'Fully vector quantized neural network-based code-excited nonlinear predictive speech coding', *IEEE Transactions on Speech and Audio Processing* **2**(4), 482–489.

Xue, Q., Hu, Y. & Milenkovic, P. (1990), Analyses of the hidden units of the multi-layer perceptron and its application in acoustic-to-articulatory mapping, *in* 'Proceedings of the IEEE International Conference on Acoustics, Speech, and Signal Processing', Vol. 2, Albuquerque, New Mexico, USA, pp. 869–872.

Yam, Y. & Chow, T. (1995), 'Determining initial weights of feedforward neural networks based on least squares method', *Neural Processing Letters* **2**(2), 13–17.