HELSINKI UNIVERSITY OF TECHNOLOGY
Department of Computer Science and Engineering
Laboratory of Computer and Information Science

**Juho Kontio**

# Neuroevolution Based Artificial Bandwidth Expansion of Telephone Band Speech

Master's Thesis submitted in partial fulfillment of the requirements for the degree of Master of Science in Technology.

Espoo, Jun 1, 2004

Supervisor: Professor Olli Simula
Instructor: Professor Paavo Alku

HELSINKI UNIVERSITY OF TECHNOLOGY

ABSTRACT OF THE MASTER'S THESIS


| | |
|---|---|
| **Author:** | Juho Kontio |


| | |
|---|---|
| **Name of the thesis:** | Neuroevolution Based Artificial Bandwidth Expansion of Telephone Band Speech |
| **Date:** | Jun 1, 2004 **Number of pages:** 103+6 |
| **Department:** | Department of Computer Science and Engineering |
| **Professorship:** | T-115 |
| **Supervisor:** | Prof. Olli Simula |
| **Instructor:** | Prof. Paavo Alku |


Most of the current telecommunication systems transmit narrowband speech, the frequecies of the speech signal ranging from about 300 Hz to 3.4 kHz. This leads to reduced quality and intelligibility of telephone band speech. To improve the speech quality, artificial bandwidth expansion can be used to generate the missing higher frequencies.

This thesis presents a neural network based algorithm for artificial bandwidth expansion, and a neuroevolution system to evolve the neural network. The expansion algorithm works by generating an initial expansion band using controlled aliasing, and then shaping the magnitude spectrum of these new frequencies using a cubic spline based magnitude shaping function. The spline shape is controlled by a set of control points, which are generated from a set of features by the neural network. The training system uses neuroevolution to evolve such neural network weights that produce an expansion that resembles an original wideband reference speech in a cepstral mean square sense. Analysis of the expansion algorithm behavior and listening test results are presented. The tests show that listeners clearly prefer speech expanded by the method over ordinary narrowband speech.



Keywords: speech processing, speech quality, artificial bandwidth expansion, neuroevolution, neural network, genetic algorithm

TEKNILLINEN KORKEAKOULU DIPLOMITYÖN TIIVISTELMÄ

| | | |
|---|---|---|
| **Tekijä:** | Juho Kontio | |
| **Työn nimi:** | Neuroevoluutiopohjainen puhelinpuheen taajuuskaistan keinotekoinen laajentaminen | |
| **Päivämäärä:** | 1.6.2004 | **Sivuja:** 103+6 |
| **Osasto:** | Tietotekniikka | |
| **Professuuri:** | T-115 | |
| **Työn valvoja:** | Prof. Olli Simula | |
| **Työn ohjaaja:** | Prof. Paavo Alku | |


Pääosa nykyisistä telekommunikaatiojärjestelmistä välittää kapeakaistaista puhetta, jonka puhesignaalin taajuuskaista on noin 300-3400 Hz. Tämä johtaa puhelinpuheen laadun ja ymmärrettävyyden heikkenemiseen. Keinotekoista taajuuskaistan laajennusta voidaan käyttää puuttuvien ylätaajuuksien luomiseen, ja siten parantaa puheen laatua.

Tässä työssä esitellään neuroverkkoon perustuva algoritmi keinotekoiseen taajuuskaistan laajennukseen, sekä neuroevoluutiojärjestelmä algoritmin neuroverkon opettamiseen. Laajennusalgoritmi luo alustavan laajennuskaistan käyttäen hallittua laskostumista ja muokkaa sen spektriä kuutiolliseen splini-käyrään pohjautuvalla magnitudinmuokkausfunktiolla. Muokkauskäyrän muotoa ohjataan joukolla ohjauspisteitä, jotka neuroverkko laskee kapeakaistaisesta signaalista laskettujen piirteiden pohjalta. Opetusjärjestelmä säätää neuroevoluution avulla neuroverkon painoja siten, että niiden tuottama laajennus muistuttaa neliölliseltä kepstrivirheeltään alkuperäistä laajakaistaista puhesignaalia. Työssä analysoidaan laajennusmenetelmän toimintaa sekä esitetään kuuntelutestituloksia. Testit osoittavat, että kuuntelijat pitävät menetelmällä keinotekoisesti taajuuskaistaltaan laajennettua puhetta selkeästi kapeakaistaista puhetta miellyttävämpänä.



Avainsanat: puheenkäsittely, puheen laatu, keinotekoinen taajuuskaistan laajennus, neuroevoluutio, neuroverkko, geneettinen algoritmi

# Acknowledgements

This thesis was written in the Laboratory of Acoustics and Signal Processing at Helsinki University of Technology. The thesis work was a part of a collaboration project with the Audio-Visual Systems Laboratory of Nokia Research Center.

First and foremost, I would like to thank Professor Paavo Alku for his excellent leadership and guidance during this project. His insights and relentless attention to detail helped to refine this thesis into what it is today. I would also like to thank my thesis supervisor Professor Olli Simula who, despite his busy schedule, went the extra mile to review and grade this thesis on short notice. In addition, I would like to thank everyone in the Speech and Audio DSP group at the Audio-Visual Systems Laboratory I had the pleasure of working with, especially Laura Laaksonen and Päivi Valve, whose ideas, comments and suggestions helped to bring this work together. Finally, I would like to thank my listening test subjects, who took time from their busy schedules to participate in the test.

Otaniemi, June 1, 2004

Juho Kontio

# Contents

# Abbreviations

| | |
|---|---|
| $a(k)$ | Autoregressive coefficients |
| $f_c$ | Cutoff frequency |
| $f_s$ | Sampling frequency |
| $f_s b$ | Stopband edge |
| I(x;y) | Mutual information |
| ABE | Artificial Bandwidth Expansion |
| ANN | Artificial Neural Network |
| BGA | Breeder Genetic Algorithm |
| CCR | Comparative Category Rating |
| CMOS | Comparison Mean Opinion Score |
| DSP | Digital Signal Processing |
| EA | Evolutionary Algorithm |
| EC | Evolutionary Computation |
| EM | Expectation Maximization |
| ESP | Enforced Subpopulations |
| FFT | Fast Fourier Transform |
| GA | Genetic Algorithm |
| GMM | Gaussian Mixture Model |
| GSM | Global System for Mobile Communications |
| HMM | Hidden Markov Model |
| HUT | Helsinki University of Technology |
| IFFT | Inverse Fast Fourier Transform |
| LBG | Linde Buzo Gray (algorithm) |
| LP | Linear Prediction |
| LPC | Linear Predictive Coding |
| LSP | Line Spectral Pairs |
| MLP | Multilayer Perceptron |
| MMSE | Minimum Mean Square Error |
| MOGA | Multi-Objective Genetic Algorithm |

| | |
|---|---|
| MPGA | Multi-Population Genetic Algorithm |
| MSE | Mean Square Error |
| NE | Neuroevolution |
| NEABE | Neuroevolution of Artificial Bandwidth Expansion |
| NEAT | Neuroevolution of Augmented Topologies |
| NN | Neural Network |
| PDF | Probability Density Function |
| PSTN | Public Switched Telephone Network |
| SNN | Spiking Neural Network |
| SUS | Stochastic Universal Sampling |
| TWEANN | Topology and Weight Evolving Artificial Neural Network |
| VQ | Vector Quantization |
| WSS | Wide Sense Stationary |

# List of Figures

# List of Tables

# Chapter 1

# Introduction

Most of the current telecommunication systems, such as PSTN and GSM, sample speech with 8 kHz sampling frequency. According to the Nyqvist theorem, the maximal frequency range that can be transmitted with a sampling frequency of 8 kHz is limited to 4 kHz. Partly for historical reasons, the speech transmitted in telephone networks has even smaller bandwidth, containing frequencies from 300 Hz to 3.4 kHz.

Human voice contains frequencies that are much higher than those transmitted by telecommunication systems. Speech sounds can contain energy at frequencies beyond 10 kHz, but for most purposes considering a much smaller frequency range is sufficient [31]. A readily intelligible signal can be transmitted in less than 5 kHz of bandwidth.

Unfortunately, the *narrowband* speech transmitted by the telephone systems does not have even such frequency range. While the narrowband speech signal contains enough information to preserve intelligibility to some degree [30], the lack of higher frequencies makes the speech sound unnatural and muffled. Especially the sibilants like /s/ suffer, as the lowest resonance frequency for them can easily fall outside the narrowband[1]. In the case of /s/, this leads to some speakers sounding like they are speaking with a lisp.

In addition, while the narrowband speech is intelligible, the listening effort required from the listener is high. Higher bandwidth speech signals can be comprehended more easily and they sound more natural. This has lead to development of wider bandwidth speech codecs to be used especially in the third generation mobile communication systems. For example the Adaptive Multi-Rate Wideband (AMR-WB) transmits frequencies from 70 Hz to 7 kHz, providing natural sounding speech with good intelligibility.

In the future, wideband speech transmission will become more common, but it will take time before all parts of the telephone system support wideband transmission. To enable smooth transition from narrowband to wideband systems it would be desirable to be able to *artificially* create the missing frequencies of the narrowband transmitted speech in wideband equipment,

[1] The lowest resonance frequency in /s/ is around 4 kHz for a male speaker [31].

so that it would more closely resemble the wideband transmitted speech. This way it would be possible to improve speech quality and reduce the perceived dissimilarities, when operating in a mixed bandwidth environment containing both narrowband and wideband speech channels and communication devices. This kind of **artificial bandwidth expansion** (ABE) would provide an economic way to enhance the perceived speech quality.

As the narrowband signal and the missing highband signal are created by the same speech production process, there is a reason to believe that the spectral envelope of the lower and higher frequency bands of the speech signal are dependent. If a model of this dependency could be made, it could be used to map the narrowband speech signal into a corresponding wideband speech signal.

Unfortunately, while the basics of the human speech production are known considerably well, the wide variety of aspects affecting it has made developing accurate models of the process difficult. Moreover, the physical properties and speaking style of each speaker are unique, leading to high differences in speech signals even when the spoken phrase is the same.

## 1.1 Overview of the Thesis

This thesis was written within an artificial bandwidth expansion project that was a collaboration with the Laboratory of Acoustics and Audio Signal Processing at Helsinki University of Technology and the Audio-Visual Systems Laboratory at Nokia Research Center. The original project aim was to improve an existing expansion algorithm that was developed in a previous collaboration project. During the development, a new idea of using artificial evolution for learning artificially bandwidth expansion emerged. The idea was first tested by using genetic algorithms to tune different parameters of the old algorithm. As it was successful, the work on a system that would produce neural network based artificial bandwidth expanders using neuroevolution was launched.

During the last decade, several artificial bandwidth expansion algorithms have been proposed by various authors. The main goal for all of them has been the same, to improve the quality of the speech by adding new spectral components to bandlimited speech. An overview of the existing work is given in **chapter 2**.

Neuroevolution has shown great promise in complex learning tasks and has been used for a wide variety of control tasks ranging from game playing to finless rocket guidance. It represents an alternative to statistical techniques that attempt to estimate the utility of particular actions in particular states. In neuroevolution, artificial evolution is used to train neural networks capable of controlling complex systems. An introduction to neural networks, artificial evolution and neuroevolution is given in chapter **chapter 3**.

The developed algorithm, NeuroEvolution of Artificial Bandwidth Expansion (NEABE), that was invented, is presented in chapter **chapter 4**. The chapter discusses the design and

implementation of the algorithm and the various choices made.

To assess the algorithm quality, various tests and experiments were made. The goal of the experiments was on one hand to analyze the behavior of the algorithm and on the other hand to get an idea of the subjective quality of the expansions produced by it. These experiments and their results are presented in **chapter 5**.

**Chapter 6** completes the thesis with conclusions of the topic and presents some ideas for future work.

# Chapter 2

# Methods of Artificial Bandwidth Expansion

Artificial bandwidth expansion can be defined as the process of expanding the signal bandwidth by artificially generating the missing frequencies of the signal, using only the information contained in the narrowband signal.

Most, if not all, of the artificial bandwidth expansion methods that have been developed are expanding *speech* signals. The most common application for the expansion methods is expanding narrowband telephone speech of bandwidth 300–3400 Hz to enhance the speech quality. There is, however, no reason why some of the (statistical) artificial bandwidth expansion methods could not be used to expand any signal that has enough mutual information between its base band and expansion band.

In this chapter various methods for artificial bandwidth expansion of telephone band (300–3400 Hz) speech to span frequencies up to 7 kHz or 8 kHz are described. Some of the methods may also add spectral components to cover the frequency band below 300 Hz, but the main emphasis is on the expansion towards high frequencies.

The methods have been divided into two groups. The first group, described in section 2.1, utilizes envelope aliasing, nonlinear processing or artificial neural network to produce the frequency components of the exansion band. The methods of the second group, discussed in section 2.2, are based on a linear model of human speech production. They typically use a linear prediction [35] to separate the signal into a spectral envelope model and a spectrally (approximately) flat excitation signal and expand these separately.

## 2.1 Methods without Speech Production Model

The artificial bandwidth expansion algorithms described in this section, as opposed to the ones described in the next section, are not based on linear model of human speech production. In

section 2.1.1, nonlinear processing based methods are explained. Section 2.1.2 covers methods based on envelope aliasing and 2.1.3 outlines a method which uses envelope aliasing and frame classification. Section 2.1.4 describes an artificial neural network based method.

### 2.1.1 Nonlinear Processing

The general idea of the nonlinear processing methods is to use a suitable nonlinear function to distort the upsampled and lowpass filtered narrowband signal to produce the high frequencies into the expansion band. The generated artificial expansion band is then shaped with a spectral shaping filter to produce a more natural sounding highband. After that, the expansion band signal is highpass filtered and combined with the original narrowband signal to produce the bandwidth expanded signal.

Suitable distortion functions that have been used for nonlinear processing include half- and full rectification [34, 55, 65], and quadratic, cubic or saturation functions [27].

Yasukawa [65] proposes a nonlinear processing based expansion method, which utilizes rectification to produce the expansion band spectral components. Block diagram of this method is shown in Figure 2.1. Soon *et al.*[55] use a similar basic system, but add a zero crossing rate based adaptation to scale the highband gain, as shown in Figure 2.2. This scaling is used to increase the highband gain during the unvoiced sounds and decrease it during the voiced sounds. According to their experiments, this adaptation improves the log spectral error and reduces artefact severity. However, a mere highband gain alteration doesn't correct the tilt of the spectrum. Thus the spectral shape must still be incorrect, which most likely leads to reduced subjective quality.

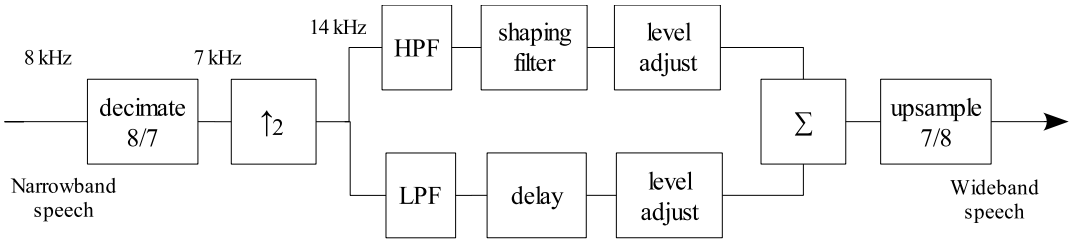



**Figure 2.1:** Block diagram of the frequency expansion algorithm based on rectification introduced by Yasukawa [65].

The main benefits of the nonlinear processing are low computational cost of the implementation and robustness. Quadratic distortion has the additional benefit of generating only harmonic distortions, thus the tonal components of the expansion band will always match the harmonic structure of the bandlimited signal during voiced sounds [27].

The major drawback of the nonlinear methods is the expansion quality. Since most of the methods aim for low computational complexity, they typically contain only one shaping filter, which is usually optimized for voiced speech frames. This leads to a wrong kind of behavior

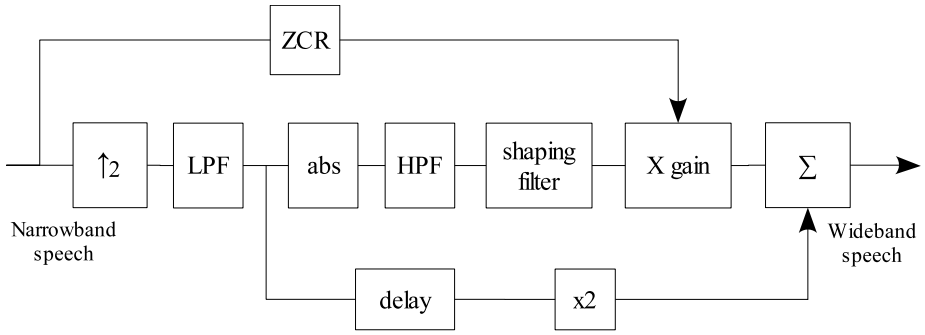



**Figure 2.2:** Block diagram of the nonlinear processing based frequency expansion system proposed by Soon *et al.* [55]. ZCR block evaluates the zero crossing rate feature, which controls the gain of the signal highband.

for fricatives, which would require highband amplification instead of the attenuation used for shaping the highband spectra of voiced frames.

### 2.1.2 Envelope Aliasing

Envelope aliasing methods are based on a simple idea of using controlled aliasing to produce a mirrored copy of the lowband spectrum as a new highband spectrum. The new highband spectrum is then shaped with a shaping filter to produce a highband that more closely resembles the highband of the original speech.

Typically, the signal is $1:2$ upsampled from 8 kHz to 16 kHz by inserting zeroes between the samples. The aliasing caused by this operation is not, however, corrected with lowpass filtering. Instead, the aliased highband portion is shaped with a shaping filter to produce a suitable artificially expanded speech signal highband. The highband is then combined with the original narrowband signal, for example by combining the lowpass filtered version of the upsampled narrowband signal with the highpass filtered version of the frequency shaped upsampled narrowband signal.

In [64], Yasukawa describes the implementation of this kind of bandwidth expansion system. In his method the signals are delay-adjusted to counter the different delays introduced on low- and highband processing paths and then both paths are level adjusted separately before combining the signals. The amount of level adjustment is tuned by trial and error, using subjective expansion quality as a guide. Yasukawa's method is illustrated in Figure 2.3.

One of the benefits of the aliasing methods in comparison to the nonlinear processing methods is the reduced need for level adjustment. The high- and lowband have a matching gain, so in principle the shaping filter could contain all the level adjustment needed, as long as the extra power brought by aliasing is pre- or post-compensated. The envelope aliasing is also computationally efficient and allows simple time domain implementation.

The aliasing methods have two drawbacks. Since the fundamental frequency of the speech

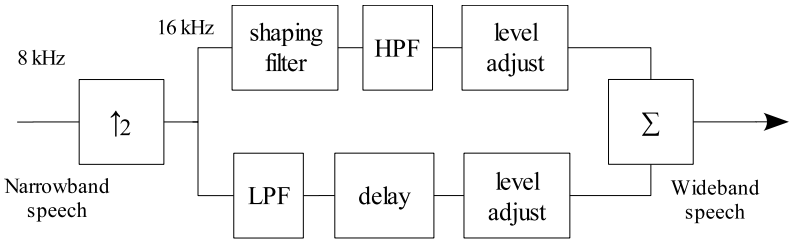



**Figure 2.3:** Block diagram of the frequency expansion algorithm based on envelope aliasing introduced by Yasukawa [64].

is not considered, the reproduced discrete structure of the expanded frequency band is inconsistent during voiced sounds, as the discrete frequency components are not correctly placed at integer multiples of fundamental frequency. In addition, if the aliasing is done directly to a telephone signal sampled with 8 kHz rate, it will result in a spectral gap in the middle frequencies (from 3.4 kHz to 4.6 kHz) of the expanded signal, as the telephone signal is typically strongly attenuated from 3.4 kHz on. The expansion will also be bandlimited in the high end to 7.7 kHz due to the telephone bandpass filtering, which usually has a passband starting from 300 Hz.

However, according to for example [27], the spectral gaps of moderate width are almost inaudible to human ear. Therefore the second disadvantage of the envelope methods is not a serious one.

### 2.1.3 Envelope Aliasing with Classification

In [30], Kallio describes an artificial bandwidth expansion method which uses frame classification to change the shaping filter according to the phoneme type expressed in the frame. Three types of shaping filters are proposed. One is for shaping voiced frames (frames containing /a/, /e/ or /i/, for example), one for shaping stop consonant frames (/k/, /p/ or /t/, among others) and the last for sibilant frames (/s/, /z/, /sh/, /zh/, ...). In addition, sibilant processing uses a more conservatively amplifying shaping filter for the first frame of the sibilant to avoid abrupt changes and reduce artefacts caused by misclassification.

Using multiple shaping filters makes it possible to considerably amplify fricatives, without introducing severe artefacts in shaping of voiced and especially stop consonant frames. This leads to clearer, more natural sounding speech. However, it also requires classification of the frames to decide when to use which filter. This classification is not easy to implement robustly and misclassifications usually lead either to over-amplification, which causes artefacts, or to over-attenuation, which leads to lower subjective speech quality[1].

The ABE algorithm in [30] computes spectral tilt estimate which is used both to classify the frame, and to adapt the shaping filter. The filter adaptation reduces the over-attenuation

[1]For example, overattenuating a sibilant frame causes the speaker to sound like he/she is speaking with a lisp.

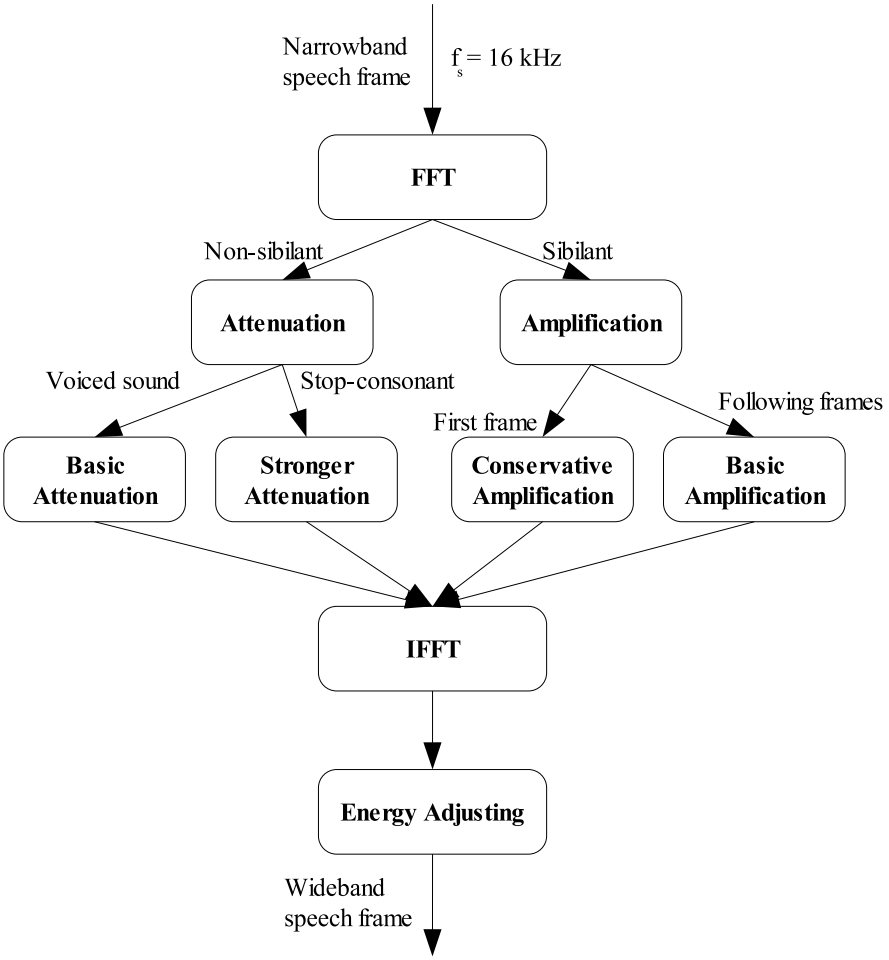


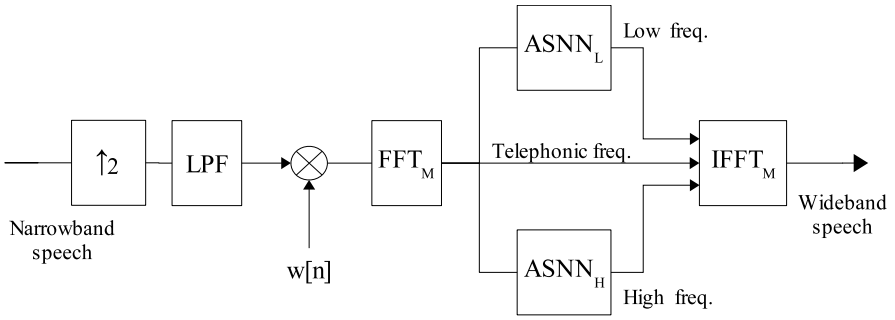

**Figure 2.4:** Block diagram of the frequency expansion algorithm based on envelope aliasing with classification, originally described in [30].

problems and smooths the transitions between phoneme classes.

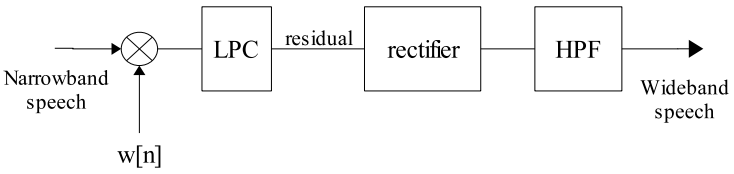
A block diagram of the ABE method by Kallio is given in Figure 2.4.

### 2.1.4 Adaptive Spline Neural Network

Uncini *et al.* [60] describe an artificial bandwidth expansion method based on Adaptive Spline Neural Networks (ASNN). The architecture of the system is illustrated in Figure 2.5(a). The system generates the missing low and high frequency components of the signal by using separate adaptive spline neural networks for both. The networks take telephone band (300 Hz - 3400 Hz) FFT coefficients as inputs and produce FFT coefficients as outputs. For highband expansion of unvoiced sounds, an additional scheme utilizing nonlinear processing, illustrated in Figure 2.5(b) is used instead of the $\mathrm{ASNN_H}$.




(a) The architecture of the bandwidth expander. $M$ is the length of the window $w[n]$ used for the STFT and LPF is the interpolation filter.




(b) Additional scheme used for expansion of high frequency part of the unvoiced sounds. HPF is a highpass filter identical to the one in Yasukawa's nonlinear method [65].

**Figure 2.5:** Adaptive spline neural network based artificial bandwidth expander developed by Uncini *et al.* [60].

The implementation used in [60] uses 64 point complex FFT/IFFT, which is too short to contain even a single pitch period of wideband signal. According to Rabiner [49], too short frame length will cause the FFT components to fluctuate very rapidly depending on the exact positioning of the frame. The short frame length also leads to reduced spectral resolution, which might affect the expansion quality.

## 2.2 Methods Based on Linear Model of Speech Production

The artificial bandwidth expansion algorithms described in this section are based on a source-filter model of human speech production, illustrated in Figure 2.6 [49]. In the model, an excitation signal is shaped with a filter that models the vocal tract and lip radiation effects. The vocal tract is a part of the human voice production apparatus that starts from the vocal folds and ends to lips. Lip radiation effect corresponds to the changing of the volume velocity at lips into a speech pressure waveform in free field outside the mouth [49].

There are two excitation signals, one for voiced sounds and another for unvoiced sounds. They correspond to the speech excitation by glottal[2] pulses, and the turbulent noise generated

[2]named after the *glottis*, i.e. the orifice between the vibrating vocal folds

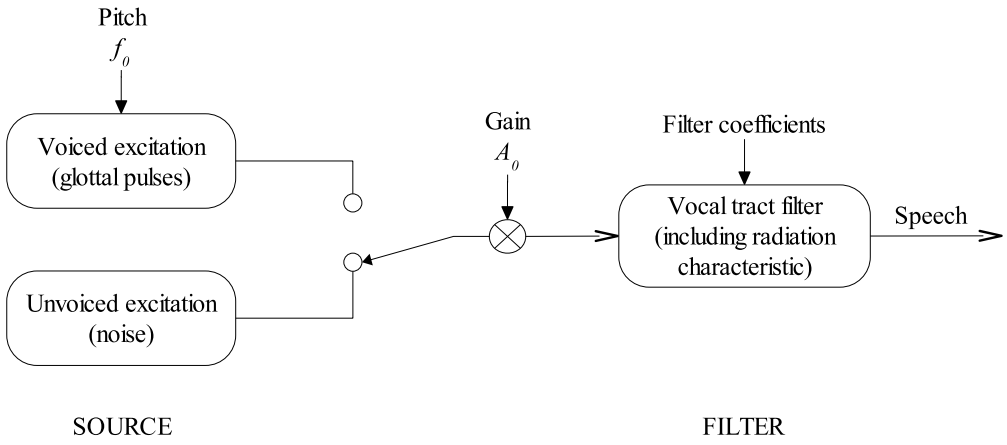



**Figure 2.6:** A source-filter model of speech production.

by forcing air at high velocity through a vocal tract constriction, respectively. The vocal tract filter modulates the frequency content of the excitation signal to produce the final utterance. By altering its parameters, realizations of different phonemes can be produced.

Typical way of using the linear speech production model in artificial bandwidth expansion can be described in six steps:

1. Extract auto-regressive(AR) coefficients from the narrowband speech frame.

2. Construct a linear prediction analysis filter which uses the extracted coefficients. Filter the narrowband frame with the analysis filter to get an estimate for the excitation signal.

3. Expand the excitation signal using one of the methods described in section 2.1. The expansion method is not so critical when expanding excitation signal, as its influence on the final expansion quality is not nearly as significant as that of the spectral envelope estimate.

4. Estimate wideband AR coefficients from the corresponding narrowband coefficients using some estimation method (for example HMM, GMM, NN and codebook methods have been used). Additional features calculated from the narrowband frame(s) may be utilized to improve the estimate accuracy.

5. Construct a linear prediction synthesis filter using the estimated wideband AR coefficients. Filter the expanded excitation signal with the synthesis filter.

6. If the method does not ensure the preservation of the narrowband information, highpass filter the expanded signal and combine it with the original narrowband signal to preserve the original lowband. Power adjust as needed before combining the signals.

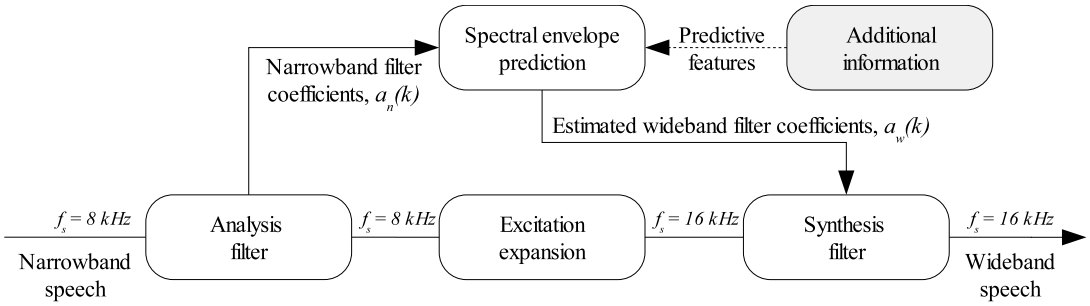



**Figure 2.7:** Block diagram illustrating generic linear prediction based artificial bandwidth expansion scheme.

This generic linear prediction based artificial bandwidth expansion scheme is illustrated in Figure 2.7.

The most important differences in the source-filter model based methods are in the spectral envelope estimation methods. In the following subsections, algorithms based on various spectral envelope prediction methods are overviewed.

### 2.2.1 Linear Mapping

While the spectral envelope prediction is generally considered to be a non-linear problem, there have been some attempts to use linear mapping to solve it. In [11], Epps and Holmes have explored using ordinary linear mapping and piecewise linear mapping based spectral envelope prediction methods for artificial bandwidth expansion.

In the linear mapping method, a vector of narrowband parameters $\mathbf{x} = [x_1, x_2, \ldots, x_m]$ is mapped into the output highband envelope parameter vector $\mathbf{y} = [y_1, y_2, \ldots, y_n]$ using a linear mapping

$$\mathbf{y} = \mathbf{W}\mathbf{x} \tag{2.1}$$

where elements of $\mathbf{W}$ are determined using least squares. Thus, matrices $\mathbf{X}$ and $\mathbf{Y}$, whose rows consist of narrowband and highband parameter vectors respectively, are formed from a large training database, and $\mathbf{W}$ is calculated as

$$\mathbf{W} = (\mathbf{X^T X})^{-1}\mathbf{X^T Y} \tag{2.2}$$

In the piecewise linear mapping method, the training data ($\mathbf{X}$ and $\mathbf{Y}$) is divided into $L$ partitions using some clustering technique. A separate $\mathbf{W}_l$, where $l \in 1, 2, \ldots, L$, is calculated for each of the clusters using equation 2.2. The prediction equation becomes:

$$\mathbf{y} = \mathbf{W}_l\,\mathbf{x} \tag{2.3}$$

where $l$ is the cluster the parameter vector $\mathbf{x}$ belongs to.

Avedano, Hermansky and Wan [1] map time trajectories of narrowband LPC-cepstral coefficients linearly into LPC-cepstral coefficients of the wideband signal. The used mapping is

$$\hat{C}_r^d(k) = \sum_{i=1}^{p} \sum_{l=-M}^{M} W_{i,r}(l) C_i(k-l) \tag{2.4}$$

where $\hat{C}_r^d(k)$ is the estimate of the $r$th cepstral coefficient, $C_i(k)$ is the $i$th narrowband LPC-cepstral coefficient and $W_{i,r}$ are FIR filter coefficients, selected in such a way that $\hat{C}_r^d$ becomes the least squares estimate of the original $C_r^d$.

The linear mapping methods are conceptually simple and usually quite efficient to implement. However, due to the nonlinear nature of the spectral envelope prediction the estimate quality is usually limited[3], which may lead to worse subjective expanded speech quality.

### 2.2.2 Codebook Mapping

Many of the artificial bandwidth expansion methods have used some variant of codebook mapping for spectral envelope prediction. In codebook mapping, the predicted spectral envelope is selected from a pre-designed codebook. The narrowband envelope is compared to wideband envelope entries in the codebook, and the entry closest to the original narrowband envelope is selected. The closeness is measured by a distance metric, which varies from one method to another. The wideband envelope corresponding to the selected entry is used as the spectral envelope estimate.

The entries for the codebook are selected by clustering the training data and selecting representative vectors for all clusters. This is typically done using the LBG algorithm.

Enbom and Kleijn [10] use a codebook containing narrowband mel-frequency cepstral coefficients (MFCCs) and the corresponding wideband LSP coefficients. For clustering and selecting the best matching codeword from the codebook they use the following distance measure:

$$D_n = \sum_{i=1}^{K} (c(i) - \hat{c}_n(i))^2 \tag{2.5}$$

where $c(i)$ is the MFCC of the current speech frame, $\hat{c}_n(i)$ is the MFCC of codeword $n$ and $K$ is the order of the MFCC.

Chan and Hui [6] use a single codebook which consists of wideband LSP vectors. For clustering and selecting the best matching codeword they measure the spectral distortion on the signal lowband:

$$SD_{AV} = \left[ \frac{1}{N} \sum_{n=1}^{N} \frac{1}{\pi} \int_{-\pi/2}^{\pi/2} \Big( 10 \log |H_n(\omega)| - 10 \log |\hat{H}_n(\omega)| \Big)^2 \mathrm{d}\omega \right]^{\frac{1}{2}} \tag{2.6}$$

[3]For example in [11], linear mapping had a highband distortion of 3.74 dB, piecewise linear mapping 3.62 dB and ordinary codebook mapping 3.35 dB.

where $H(\omega)$ is the signal wideband LPC spectrum and $\hat{H}(\omega)$ is the LPC spectrum of the codeword.

The benefit of using a single codebook entry instead of a (narrowband, wideband) pair, is that the coefficients for the analysis and synthesis filters are the same and their transfer functions are mutually inverse. Therefore, if the lowband of the excitation signal is retained, the lowband of the expanded signal will be identical to the original narrowband signal.

Codebook mapping has been applied to bandwidth expansion related spectral envelope prediction widely, and it can be seen as a benchmark method, against which other methods are compared. While it is capable of producing rather good results in the spectral distortion sense, it seems to have a tendency to create perceptually annoying signal highband power overestimates [10]. In addition, computational costs for the codebook methods are considerable, as usually every codebook codeword must be compared with the narrowband spectrum to determine the winning codeword.

### 2.2.3 Adaptive Codebook Mapping

In its basic form, the number of possible highband envelopes which can be predicted by codebook mapping is limited to the size of the codebook. Epps and Holmes [11] propose an interpolation method to reduce the distortion resulting from this limitation. In the interpolative codebook mapping, instead of selecting a single codeword closest to the narrowband spectral envelope, $N$ closest codewords are selected and their corresponding wideband code vectors are combined using weighted average.

In addition to interpolation, Epps and Holmes suggest using different codebooks for voiced and unvoiced speech frames. During training, a binary voicing detection algorithm is used to divide the training data into envelopes from voiced and unvoiced frames. The voiced and unvoiced codebooks are then trained separately. Because different degrees of voicing are associated with different spectral envelope shapes, this produces additional information for the expansion process.

### 2.2.4 Gaussian Mixture Models

Gaussian mixture models (GMM) have also been proposed for highband spectral envelope prediction. According to Park and Kim [45] there are two principal motivations for using the GMM as a representation of the acoustic space. Firstly, the empirical observation that a linear combination of Gaussian basis functions is capable of representing a large class of sample distributions. Secondly, the individual component densities can be intuitively interpreted to represent some broad acoustic classes.

In GMM based approaches, a gaussian mixture model for mapping the narrowband feature vector into a wideband spectral envelope is learned. Typically the model is learned by using

expectation maximization (EM) algorithm to obtain a maximum likelihood (ML) estimate for a given training set. The trained model consists of triplets $\lambda = \{\alpha_i, \mu_i, C_i\}, i = 1, \ldots, Q$; where $\alpha_i$ is the weight of the mixture component $i$, $\mu_i$ is the mean vector of the component, $C_i$ is the $n \times n$ covariance matrix and $Q$ is the number of mixture components. When estimating, the GMM produces a weighted combination of the mixture component mean vectors as the highband spectral envelope estimate. Thus, GMM approach can be considered to be an intelligent codebook interpolator, where the component mean vectors correspond to the codewords of the codebook and the mixture parameters define the way in which to combine them.

Park and Kim [45] describe a GMM based approach in which the narrowband feature vector consists of the spectral vector sequence of narrowband speech and the envelope estimate consists of the spectral vector sequence of wideband speech. The joint density of these vectors, $z = (x, y)^T$, is modeled as a mixture of $Q$ $2n$-variate Gaussian functions and a minimum MSE estimate for the wideband is calculated from the model.

Qian & Kabal [48] use a similar approach, except that in their model the feature vector consists of LSP coefficients and an additional pitch gain feature, and the highband spectral envelope is estimated as a vector of LSP coefficients.

Generally, the GMM based methods perform better than corresponding codebook mapping based methods. Both Park and Kim [45], and Qian and Kabal [48] report an improvement in comparison to the VQ codebook mapping method. Park and Kim also report a high preference for the GMM based method in a subjective preference test. The computational requirements for the GMM methods are around the same order as for the codebook mapping methods.

### 2.2.5 Hidden Markov Models

Recently, some artificial bandwidth expansion methods utilizing hidden Markov models (HMM) for spectral envelope prediction have been presented. The main benefit of using HMM for envelope prediction is its capability of implicitly exploiting information from the preceding signal frames to improve the estimation quality. Each state of the HMM usually corresponds to a specific speech sound, similar to the way each codeword in the codebook methods does. Therefore, the states of the HMM can be seen to relate to the possible positions of the articulatory organs, while the state transition probabilities can be seen to model the way the articulatory organs can be moved from the position.

Jax and Vary [29] use HMM to estimate a set of wideband linear prediction coefficients, which are then used for both analysis and synthesis filter. This way the lowband of the expansion will be identical to the original narrowband signal. Their HMM model takes an input vector consisting of 10 narrowband auto-correlation coefficients and five additional features[4], uses EM trained Gaussian mixture models to approximate the observation probability PDFs and produces a cepstral MMSE estimate of the wideband LP coefficients. The state and state

[4] zero-crossing rate, normalized frame energy, gradient index, local kurtosis and spectral centroid

transition probabilities are estimated from the wideband training data using the true state sequence.

Hosoki *et al.* [23] introduce a bandwidth expansion algorithm which uses subband HMMs (SHMMs) for envelope estimation. They divide the wideband speech into a number of subbands and extract their features independently. The relationships between spectral subbands are learned during the training phase. During the reconstruction phase, the lowband HMM decodes the state sequence, which the highband HMM uses for spectral envelope estimation. A separate HMM is used to model the energy ratio between the low- and highband.

Jax and Vary report that exploiting the transition probabilities of the HMM considerably reduced the occurence of artefacts [29]. The drawback of the HMM method is that it requires considerable amounts of data to reliably estimate the state models and transition probabilities. In [29], over 25 hours (4.5 million frames) of training data is used.

### 2.2.6 Neural Network Based Methods

Iser and Schmidt [26] compare neural network based spectral envelope prediction method to a codebook method. Their neural network envelope predictor uses LPC coefficients which have been converted into cepstral domain as inputs. 12 LPC coefficients from two consecutive frames are transformed into 36 cepstral coefficients using the following equation

$$\check{c}_i(n) = \begin{cases} \check{a}_i(n) + \dfrac{1}{i}\displaystyle\sum_{k=1}^{i-1} k\,\check{c}_k(n)\,\check{a}_{i-k}(n), & \text{for } i = 1 \ldots p \\ \dfrac{1}{i}\displaystyle\sum_{k=1}^{i-1} k\,\check{c}_k(n)\,\check{a}_{i-k}(n), & \text{else} \end{cases} \tag{2.7}$$

where $\check{a}_i(n)$ is the $i$th LPC coefficient and $\check{c}_i(n)$ is the $i$th cepstral coefficient. The LPC is evaluated for a preprocessed signal to prevent the LPC from modeling the edges caused by the bandpass filtering of the signal. The preprocessing consists of reducing the sample rate so that the signal contains only frequency components under 3400 Hz, and setting the frequencies below 300 Hz to a mean value calculated out of a few samples above 300 Hz.

Network output is given as cepstral LPC coefficients. The output LPC order is 20, resulting in 30 cepstral coefficient outputs. The outputs are post processed to force the poles of the LPC filter inside the unit circle to ensure the model stability.

The neural network used by the method is a feedforward network, which is trained using the standard backpropagation algorithm. The distance measure in the learning process is the cepstral MSE between the network output and the cepstral LPC coefficients calculated from the original wideband signal. The network contains 36 hidden neurons in a single hidden layer.

For both the codebook method and the neural network method, the signals are power adjusted and phase manipulated after the initial expansion. The resulting signal is bandstop filtered and combined with the original narrowband signal.

According to the results reported by Iser and Schmidt, the neural network achieves better log spectral distortion and cepstral distortion, but slightly worse log area ratio (LAR) distortion. However, in the listening tests, 80% of the listeners prefer the codebook based method.

The computational costs of the neural networks are considerably lower than those of codebook methods.

# Chapter 3

# Neuroevolution

Neuroevolution (NE) can be defined as the artificial evolution of neural networks using Evolutionary Computation[1] (EC) algorithms. It has shown great promise in complex reinforcement learning tasks [56] and has been used for a wide variety of control tasks ranging from game playing [7] to finless rocket guidance [16]. Neuroevolution approach represents an alternative to statistical techniques that attempt to estimate the utility of particular actions in particular states of the world [56].

In this chapter, an introduction to the basics of neuroevolution is given. We begin by overviewing neural networks, in section 3.1, and genetic algorithms, in section 3.2. The two topics are then combined, as we explore the evolutionary approach to neural network training, in section 3.3. Finally, in section 3.4, the chapter is concluded by a look into specialized neuroevolution methods, which have been designed to overcome problems encountered when utilizing evolutionary algorithms for neuroevolution.

## 3.1 Neural Networks

In [20], an *artificial neural network*[2] (ANN) is defined as:

> A massively parallel distributed processor made up of simple processing units, which has a natural propensity for storing experiential knowledge and making it available for use. It resembles the brain in two respects:
>
> 1. Knowledge is acquired by the network from its environment through a learn-

[1] *Evolutionary Computation* is the general term for several computational techniques which are based to some degree on the evolution of biological life in the natural world [52]. The most widely used form of evolutionary computation are *Genetic Algorithms*. Others include *Genetic Programming* and *Evolution Strategies*, among others.

[2] While the correct term is *artificial* neural network, it is a quite common convention to refer to ANN's simply as neural networks (NN's). As this thesis only discusses artificial neural networks, this convention will be adopted here.

ing process.

2. Interneuron connection strengths, known as synaptic weights, are used to store the acquired knowledge.

### 3.1.1 Architecture

A NN consists of a set of processing elements, also known as neurons or nodes, which are interconnected. It can be described as a directed graph in which each node $i$ performs a transfer function $f_i$ of the form:

$$y_i = f_i\left(\sum_{j=1}^{n} w_{ij}x_j + \theta_i\right) \tag{3.1}$$

where $y_i$ is the output of the node $i$, $x_j$ is the $j$th input to the node, $w_{ij}$ is the connection weight between nodes $i$ and $j$, and $\theta_i$ is the bias (or threshold) of the node. Usually, $f_i$ is nonlinear, such as a heaviside, sigmoid, or Gaussian function.

NN's can be divided into feedforward and recurrent classes according to their connectivity. A recurrent NN has feedback connections in its graph. More formally, a neural network is feedforward, if there exists a method which numbers all the nodes in the network such that there is no connection from a node with a larger number to a node with a smaller number [63].

Perhaps the most widely used neural network type is the fully connected feedforward multilayer perceptron (MLP), which consists of separate layers of neurons, in which every neuron is connected to all neurons in the next layer and receives input from all neurons in the previous layer. The signal is input into the first layer, called the input layer, and propagated through one or more hidden layers into the output layer, from which the output signal can be read. A single hidden layer is often used, and is, in theory, sufficient to model any continuous function [8]. An illustration of a single hidden layer fully connected MLP is given in Figure 3.1.

### 3.1.2 Learning

Learning in NN's is typically accomplished using a set of examples. The network learns to produce desired results for the learning set and generalizes the results for other inputs. Learning in NN's is also called training, because the learning is achieved by adjusting the connection weights iteratively to make the network perform a desired task. Types of learning that are used with the NN's can be divided to three groups: supervised, unsupervised and reinforcement learning.

Supervised learning is based on direct comparison between the actual output of NN and the desired output. It is often formulated as the minimization of a differentiable error function, such as MSE over whole teaching sample set. A gradient-descent based optimization algorithm such as backpropagation (BP) can then be used to adjust the connection weights in the NN iteratively in order to minimize the error. [63]

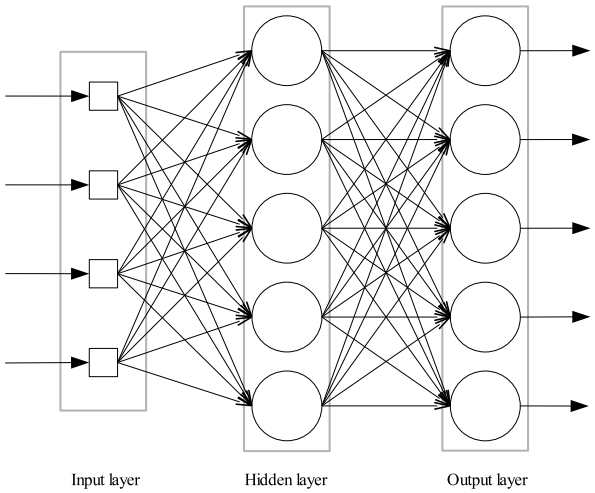



**Figure 3.1:** MLP with a single hidden layer. Squares represent input signal sensors and circles computational neurons. Each neuron sums its inputs and applies a (typically nonlinear) transform function to it. A neuron specific bias term is added to the sum before transformation.

In reinforcement learning the desired output is unknown. Instead of explicit error minimization, learning is performed through continued interaction with the environment in order to maximize a scalar index of performance given by a *critic*. The critic is typically implemented as an *objective function*, which credits desired behavior and penalizes unwanted behavior. There is often a delay in the generation of this reinforcement signal. In may be, for instance, that the NN performance cannot be reliably estimated before the end of the task. This is the case in, for example, game playing. Only after the game it is certain which side won and thus the reinforcement should be given only then.

In comparison to supervised learning, reinforcement learning is much more applicable to complex problems, as it is usually hard or even impossible to provide a correct answer to them and typically the task performance evaluation in them must be delayed.

Unsupervised learning, the third learning type, requires no information on “correct output”, as it is based solely on the correlations among input data [63]. Typical use for unsupervised learning is clustering. Clustering is a form of unsupervised pattern recognition, in which the aim is to unravel the underlying similarities in the input vectors and group “similar” vectors together into clusters, which can then be analyzed and labeled by a human expert [58].

For feedforward MLP’s in supervised learning tasks, which is probably the most common case the NN’s are applied to, the training is typically done using some variant of gradient descent, such as resilient backpropagation [53], scaled conjugate gradient [32] or various quasi-Newton or Levenberg-Marquardt algorithms, Bayesian regularization backpropagation [13], for instance.

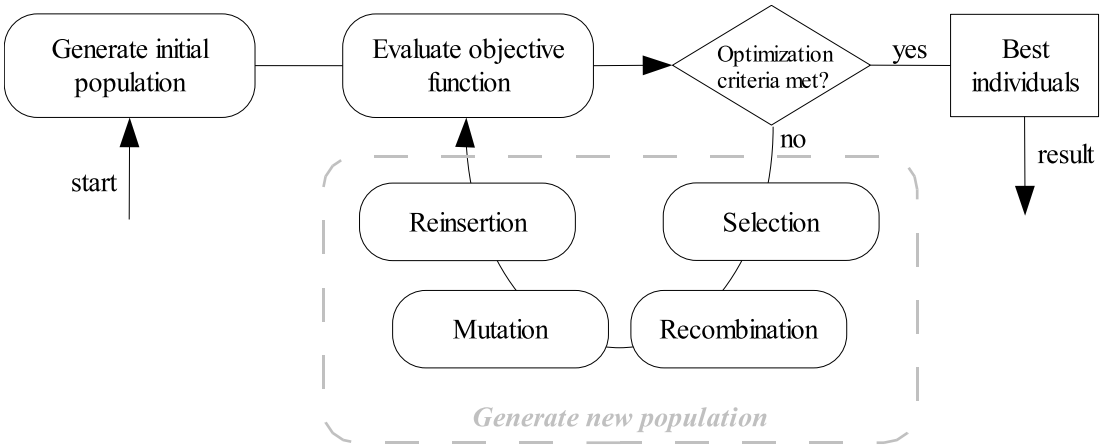



**Figure 3.2:** Structure of a single population evolutionary optimization algorithm. An iterative evolution process refines the initial population into a population of fit solutions using evaluate-select-recombine-mutate-reinsert cycle. The selection centers the search into promising areas by selecting the fittest individuals for reproduction. Recombination and mutation of the promising individuals explores the regions around them and reinsertion stores the most promising individuals into the population so they can act as a base for the next evolution cycle. When optimization criteria are met, the best individual(s) are returned as the result of the search.

## 3.2 Genetic Algorithms

The term *genetic algorithm* (GA), was first used by Holland, whose book *Adaptation in Natural and Artificial Systems* of 1975 was instrumental in creating what is now a flourishing field of research and application that goes much wider than the original GA [50]. The subject now includes *evolution strategies*, *genetic programming*, *artificial life* and *classifier systems*. All these related fields are nowadays often grouped under the heading of *evolutionary computing* or *evolutionary algorithms* (EA). [50]

While optimization had a fairly small place in Holland's work on adaptive systems, it has since become the main purpose of use for GA's. A basic scheme for optimization using a single population evolutionary algorithm is shown in Figure 3.2. The generic algorithm can be described in following steps:

1. *Generate initial population of individuals.* These are often randomly initialized, but specific domain knowledge or population from a prior run can be used to generate a more fit initial population.

2. *Evaluate objective function to determine the quality of each individual.* Objective function is the function that is being optimized, in the case of reinforcement learning it is the reinforcement signal.

3. *Test for optimization criteria fulfillment.* Determine if the optimization criteria have been

met? If they are, go to 8, otherwise go to 4. The optimization criteria can be anything from running for a specific number of generations to using a heuristic to estimate when learning has stagnated or testing whether the evolved solution is good enough in some problem specific sense.

4. *Selection.* Determine which individuals are chosen for recombination and how many offspring each selected individual produces.

5. *Recombination.* Produce new individuals by combining the information contained in the selected parents.

6. *Mutation.* Each variable in each offspring individual has a low probability to be mutated by a small perturbation.

7. *Reinsertion.* Decide which individuals to insert into the new population (from old population and from the generated offspring). Continue from step 2.

8. *Return a desired number of best individuals.* Often one, but if the results are further optimized using some other method, it may make sense to try a number of best solutions, as their potential for further optimization may differ (and is usually impossible to determine at this point).

Optimization using evolutionary algorithms differs substantially from more traditional search and optimization methods. According to Pohlheim [47], the most significant differences are:

- EA's search a population of points in parallel, not a single point.
- EA's do not require derivative information or other auxiliary knowledge; only the objective function and corresponding fitness levels influence the directions of search.
- EA's use probabilistic transition rules, not deterministic ones.
- EA's are generally more straightforward to apply.
- EA's can provide a number of potential solutions to a given problem. The final choice is left to the user. (Thus, in cases where the particular problem does not have one individual solution, for example multiobjective optimization and scheduling problems, the EA is potentially useful for identifying these alternative solutions simultaneously.)

In the following subsections, different parts of the genetic algorithm are discussed in more detail. In 3.2.1, objective functions are examined. Subsections from 3.2.2 to 3.2.5 go through different subtasks in the genetic algorithm; selection, recombination, mutation and reinsertion. Subsection 3.2.6 details multi-population GA's, an expansion of genetic algorithms to multiple subpopulations, and their migration algorithms.

### 3.2.1 Objective Function

Objective function is the heart of the genetic algorithm. Failing to select a suitable objective function may lead to stagnation of learning. The objective function should reward desirable behavior and penalize undesirable behavior. In reinforcement learning tasks, the reinforcement signal is a natural choice.

If the function to be optimized is known, it should naturally be selected as the objective function. As mentioned, there is no need for the objective function to be differentiable, or even continuous. The only requirement is that it should monotonously measure the desirability of the given solution (i.e. by sorting the solutions according to the objective function, we sort the solutions according to their desirability).

However, for example in many reinforcement learning tasks, the function to be optimized is not necessarily known explicitly. Nevertheless, it is often possible to compare two individuals and determine which of them performs better. A simple objective function can then be constructed by comparing each individual to every other individual in the population and awarding it one point for each time it performs better. This idea is intrinsically linked to tournament selection and therefore will be expanded in the selection subsection.

While in theory the selection of the objective function is quite simple, in practice things often get more difficult. Typically there may be multiple objectives that should be fulfilled at the same time, or there may be no way to measure the real objective function, but we have a set of heuristics that together reasonably well predict it. The problem, then, is how to combine these multiple objectives into a single scalar objective value. A suitable solution should offer "acceptable" performance in all objective dimensions, where "acceptable" is a problem-dependent and ultimately subjective concept [12].

A weighted sum of all objectives is certainly a possible solution, but, especially in case of heuristics, two of the objectives may contain partly overlapping information, the importance of which may then get overweighted. In addition, selecting weights for different objectives is a difficult task and requires insight into the statistical behavior of the selected objective functions.

A better solution is to use a multi-objective evolutionary algorithm. Many such algorithms have been devised, most of the more successful ones using some combination of Pareto-optimal optimization[3] and niche induction[4] [12].

[3] Fonseca and Fleming [12] define Pareto-optimal family of solutions of a multi-objective optimization problem as the set of all those elements of the search space which are such that the components of the corresponding objective vectors cannot be all simultaneously improved. They also provide a more formal definition of the concept.

[4] One potential problem with multi-objective EA's is that of genetic drift or speciation, where the algorithm tends to "drift" towards areas where there are clusters of closely matched solutions and leaves other areas not well mapped out or explored at all. The effect can be reduced by using niche induction, in which the density of solutions within hypervolumes of either the decision or the objective variable spaces is restricted. [14]

### 3.2.2 Selection

In selection the individuals producing the offspring are chosen. First, each individual in the selection pool receives a fitness value (a probability of being selected for reproduction) depending on how its objective value compares to that of the other individuals in the pool. The individuals for reproduction are then sampled from the selection pool according to these probabilities.

Selection is an important step in evolutionary algorithms, as it is responsible of directing the population towards promising solution areas, while still attempting to maintain enough diversity to be able to explore new, possibly even better solution areas. This tradeoff can be controlled by varying *selective pressure*, which is defined in [47] as "the probability of the best individual being selected compared to the average probability of selection of all individuals".

Three useful measures, which can be considered when evaluating selection scheme performance, are *loss of diversity*, defined as the proportion of individuals of a population that is not selected during the selection phase; and *selection intensity* and *variance*, defined as the expected average and variance of the fitness value of the population after applying a selection method to the normalized Gaussian distribution, respectively [47].

#### Rank Based Fitness Assignment

There are two main ways to assign fitness for individuals. One is proportionate to the objective value, the other is based on the rank of the individual in population sorted by the objective value. Rank based fitness assignment overcomes the scaling problems of the proportional fitness assignment and behaves in a more robust manner, and is therefore the method of choice [47].

*Linear ranking* is a simple rank based fitness assignment, in which the best individual is assigned fitness according to the selection pressure parameter $s$ and the worst individual is given a fitness of $2 - s$. Intermediate individuals get their fitness by interpolation, thus the linear ranking fitness of the individual $i$ is:

$$p(i) = \frac{1}{N}\Big(s - \frac{2(i-1)(s-1)}{(N-1)}\Big) \tag{3.2}$$

where $N$ is the total number of individuals in the population. Valid values for the selective pressure parameter $s$ range from 1.0 to 2.0.

Blickle and Thiele [5] provide analysis of the linear ranking performance[5]. The loss of diversity $p_d$, selection intensity $I$ and selection variance $V$ as a function of selective pressure parameter $s$ are:

$$p_d(s) \;=\; \frac{1}{4}(s-1)$$

[5]The formulas from [5] have been converted to use the more widely used selective pressure parameter $s$ instead of $\eta^-$. The relationship between the two parameters is $\eta^- = 2 - s$

$$\begin{aligned} I(s) &= \frac{1}{\sqrt{\pi}}(s-1) \\ V(s) &= 1 - I(s)^2 = 1 - \frac{(s-1)^2}{\pi} \end{aligned} \tag{3.3}$$

Plots of the functions are shown in Figure 3.3.

To provide more intensive selection pressure non-linear ranking methods can be used. A typical example of these is *exponential ranking*, in which the fitness assignment function is:

$$p(i) = \frac{c-1}{c^N - 1} c^{i-1} \tag{3.4}$$

where $c$ is a parameter of the method. Selective pressure of the method is proportional to $1 - c$.

Blickle and Thiele [5] give the following performance measures for the exponential ranking:

$$\begin{aligned} p_d(\alpha) &= \frac{1 - \ln \frac{\alpha-1}{\alpha \ln \alpha}}{\ln \alpha} - \frac{\alpha}{\alpha - 1} \\ I(\alpha) &\approx 0.588 \frac{\ln \ln \frac{\pi}{\alpha}}{3.69^{\alpha}} \\ V(\alpha) &\approx \ln\left(1.2 + \frac{2.8414}{2.225\alpha - \ln \alpha}\right) \end{aligned} \tag{3.5}$$

where $\alpha = c^N$. $I(\alpha)$ and $V(\alpha)$ are approximations derived using genetic programming optimization for symbolic regression. They have a relative error of less than 6% for $\alpha \in [10^{-20}, 0.8]$. Plots for the measures are shown in Figure 3.3.

Another example of non-linear selection is *truncation selection*, in which only a the best portion $T$ of the population is allowed to reproduce. Each of the individuals in $T$ are reproduced $1/T$ times. Typical values for $T$ range from 0.1 to 0.5. The analysis in [5] gives truncation selection the following performance measures:

$$\begin{aligned} p_d(T) &= 1 - T \\ I(T) &= \frac{1}{T} \frac{1}{\sqrt{2\pi}} e^{-\frac{f_c^2}{2}} \\ V(\alpha) &= 1 - I(T)(I(T) - f_c) \end{aligned} \tag{3.6}$$

where $f_c$ is determined by $T = \int_{f_c}^{\infty} \frac{1}{\sqrt{2\pi}} e^{-\frac{f^2}{2}} \mathrm{d}f$. Plots are shown in Figure 3.3.

### Sampling Methods

After fitness has been assigned, the population is sampled to pick the individuals for reproduction. A naive way to sample the individuals is called *roulette wheel sampling*, and can be visualized as spinning a roulette wheel, the sectors of which are set equal to the relative fitness of each individual (see Fig. 3.4(a)). The wheel is spun once for each required individual. The wheel is more likely to stop on bigger sections, so fitter strings are more likely to be chosen on each occasion. This is not satisfactory. Because each parent is chosen separately, there is no

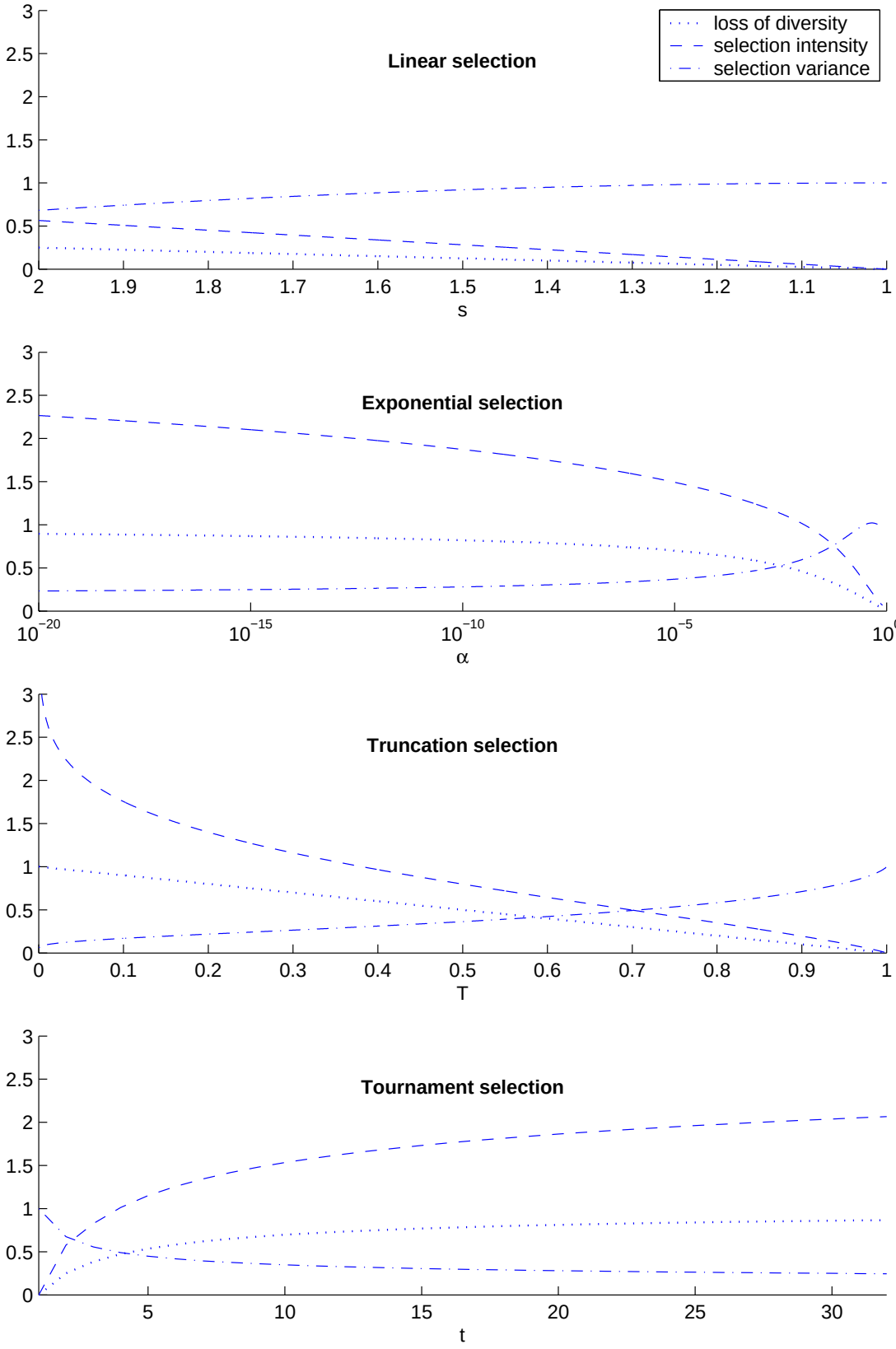
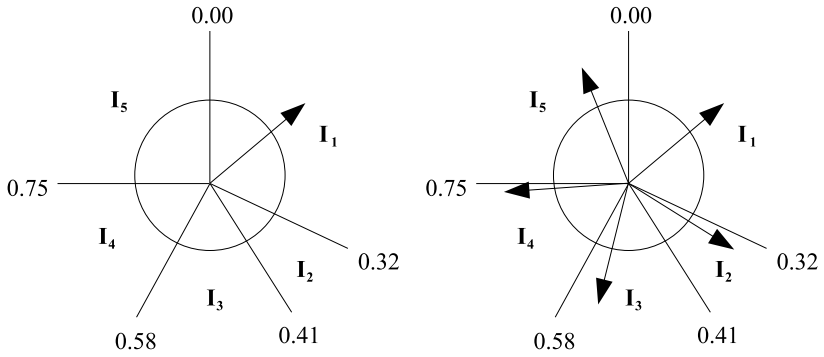



**Figure 3.3:** Loss of diversity, selective pressure and selective variance plotted for four different selection methods. A selection method with high selection intensity and low loss of diversity is usually preferable. As loss of diversity represents the proportion of the individuals that are not selected, the maximum value for it is 1.0, meaning no individuals are selected. Note the reversed parameter scale on the linear selection plots and the logarithmic parameter scale on the exponential selection plots. The selection variance of the linear selection is quadratic function of its selection pressure despite the near linear appearance on this scale.




(a) Roulette wheel sampling (b) Stochastic universal sampling

**Figure 3.4:** Selection pool of five individuals with fitnesses 0.32, 0.09, 0.17, 0.17 and 0.25 respectively. The probability of selection of each individual is proportional to the area of the sector of a roulette wheel. In roulette wheel sampling, a single pointer is spun five times to select five individuals. In stochastic universal sampling, a wheel with five equally spaced pointers is spun once to select five individuals. This ensures fair selection of individuals and thus avoids the sampling noise of the ordinary roulette wheel selection. [50]

guarantee that any particular individual, not even the best in the population, will be chosen for reproduction. This sampling error can act as a significant source of noise [19].

An elegant solution to the sampling problem, known as *stochastic universal sampling* (SUS), was introduced by Baker in [2]. It avoids the high stochastic variability of roulette wheel method by sampling all needed samples in a single pass. Instead of having only a single arm, the roulette wheel is imagined to have multiple arms, spaced at equal distances from each other (illustrated in Fig. 3.4(b)). Spinning the wheel simultaneously samples all needed individuals. From the statistical sampling theory viewpoint this corresponds to systematic random sampling [50]. Experimental work by Hancock [19] demonstrates the superiority of this approach in comparison to roulette wheel sampling. In all practical work, SUS should be preferred to roulette wheel.

### Tournament selection

*Tournament selection* is a selection method without explicit fitness assignment. In fact, it is possible to implement tournament selection based genetic algorithm without using an explicit objective function, which makes it suitable for a wide range of problems for which no explicit objective function can be formulated, but in which ordering the solutions by comparing them

with each other is feasible (such as game playing)[6].

In tournament selection, $t$ individuals are chosen at random from the selection pool and the best of them is selected for reproduction. A single tournament is held for each member of the selection pool. Goldberg and Deb have shown that the expected result for a tournament with $t = 2$ is exactly the same as for linear ranking with $s = 2$ (see [19] and references therein). The selective pressure can be increased by using larger $n$ and decreased by giving the better individual only a certain probability to win [19]. Blickle and Thiele [5] provide the following analysis results for tournaments of size $t \in [2, 30]$:

$$\begin{aligned} p_d(t) &= t^{-\frac{1}{t-1}} - t^{\frac{t}{t-1}} \\ I(t) &\approx \sqrt{2(\ln(t) - \ln(\sqrt{4.14\ln(t)}))} \\ V(t) &\approx \sqrt{\frac{2.05 + t}{3.14 t^{\frac{3}{2}}}} \end{aligned} \tag{3.7}$$

where $I(t)$ is an approximation by steepest descent method having a relative error of less than 1% for $t > 5$ and less than 2.4% for $t \in [2, 5]$, and $V(t)$ is an approximation by symbolic regression using the genetic programming optimization method having a relative error of less than 1.6%. Analytic solutions for $I(t), t \leq 5$ and $V(t), t = 2$ can be found in [5]. Plots for the measures are shown in Figure 3.3.

Tournament selection differs from the rank based selection methods much as roulette wheel and SUS sampling differ from each other. Because each tournament is held individually, it suffers from exactly the same sampling errors [19]. One way of coping with these, at the expense of a little extra computation, is described in [50]. For a $t$-tournament of population with $N$ individuals, $t$ items need to be drawn $N$ times. To do this, $t$ random permutations of numbers $1, \ldots, N$ are constructed and concatenated into one long sequence, which is then chopped up into $N$ pieces, each containing the $t$ indices of the individuals to be used in the consecutive tournaments. If $N$ is not an exact multiple of $t$, there is a small chance of some distortion where the permutations are joined, but this is not hard to deal with. Unfortunately, this requires centralized computation (for generating permutations) and thus reduces the otherwise good parallel implementation suitability of the tournament selection method.

### 3.2.3 Recombination

Recombination produces new individuals by combining the information combined in (usually two) parents. Parents are chosen randomly from the individuals that were selected for recom-

[6]Note that tournament selection is by no means the only way to deal with this. Another option is to use population relative objective function. For example, for each individual, arrange $t$ tournaments against other individuals and award 1 point per win, -1 point per loss and 0 points for a draw. This, however, basically is the tournament selection method described in another way. The benefit of this method is that it allows the use of any selection method and therefore a more fine grained control of fitness assignment than what is possible with tournament selection.

| | | | | |
|---|---|---|---|---|
| *Three crossover positions chosen at random: 1, 3 and 4* | | | | |
| *parent 1* | 1 | 10 | 0 | 10 |
| *parent 2* | 0 | 11 | 1 | 00 |
| *After exchange, the produced offspring are:* | | | | |
| *offspring 1* | 1 | 11 | 0 | 00 |
| *offspring 2* | 0 | 10 | 1 | 10 |

**Table 3.1:** An example of three-point crossover operation

bination in the selection step.

The role of recombination in the genetic algorithm based search is to combine the genomes of the fit individuals and (hopefully) combine their best parts to produce an even fitter individual. Recombination combined with selection works to converge the population around good solution points, whereas mutation (described in section 3.2.4) explores new areas by randomly perturbating the solutions. In practise the division is not that clear. Most of the (real valued) genetic recombination operators also perturb the values. However, the perturbations made by them are local, whereas mutation induced perturbations are more global in nature. The perturbations made by recombination operators are also dependent on both parents, unlike perturbations caused by mutation, which are independent of the individual being mutated.

### Binary Valued Recombination

The original genetic algorithms used *binary valued recombination*, where the two parent genomes were represented as bit strings and recombined using some binary crossover method. Therefore, they have been widely used and many different schemes for binary recombination (usually called crossover) have been devised.

A basic binary crossover between two individuals selects one or more crossover points at random and exchanges the bitstring segments which are after odd crossover points. An example with three crossover points is given in Table 3.1.

According to the number of crossover points used, the method is called either *single-point crossover*, or *multi-point crossover*. A generalization of multi-point crossover, in which the bits to be exchanged are specified by a separate crossover bitmask generated stochastically using a Bernoulli distribution, is called *uniform crossover*.

The idea behind multi-point and many other variations of the binary crossover operator is that parts of the chromosome representation that contribute most to the performance of a particular individual may not necessarily be contained in adjacent substrings (see [47] and references therein).

A wide variety of other binary crossover operators exist. For more indepth coverage of binary crossover, we refer the reader to [50].

### Real Valued Recombination

Many of the optimization problems have variables that are real valued. For them, binary crossover methods are not suitable. An example clarifies the issue:

> *Assume we have two single variable individuals $x = 4$ and $y = 3$. The binary presentations for them are $x_b = 100$ and $y_b = 011$. Therefore the binary recombination schemes may produce any three-bit bitstring, that is, values from 0 (000) to 7 (111). The issue becomes even more difficult with floating point and negative numbers.*

While different binary encodings, such as gray coding[7], can be used to alleviate the problem, a better way to avoid it is to define a real valued recombination operator.

*Discrete recombination* [41] is a very crossover like real valued recombination operator. It selects one of the corners of the hypercube defined by the parents as the offspring. More formally, if $x = (x_1, \ldots, x_n)$ and $y = (y_1, \ldots, y_n)$ are the parents to be recombined, the offspring $z = (z_1, \ldots, z_n)$ can be computed by:

$$z_i = \{x_i\} \text{ or } \{y_i\} \qquad i = 1, \ldots, n \tag{3.8}$$

where both $x_i$ and $y_i$ have an equal probability (i.e., 0.5) to be chosen. Discrete recombination for two variable case is illustrated in Figure 3.5(a).

*Extended intermediate recombination* [61, 41][8] can be defined as:

$$z_i = x_i + \alpha_i(y_i - x_i) \qquad i = 1, \ldots, n \tag{3.9}$$

where $\alpha_i$ is a scaling factor chosen uniformly at random over an interval $[-d, 1+d]$. In ordinary *intermediate recombination* $d = 0$, for extended intermediate recombination $d > 0$. A typical choice is $d = 0.25$. A separate $\alpha_i$ is sampled for each $i$.

An offspring produced by intermediate recombination is randomly chosen from the hypercube defined by its parents. In extended intermediate recombination the hypercube is scaled by $1 + d$ before picking the offspring. This is illustrated in Figure 3.5(b).

*Extended line recombination* [61, 41][8] is similar to extended intermediate recombination, except that a single $\alpha$ is used for all variables:

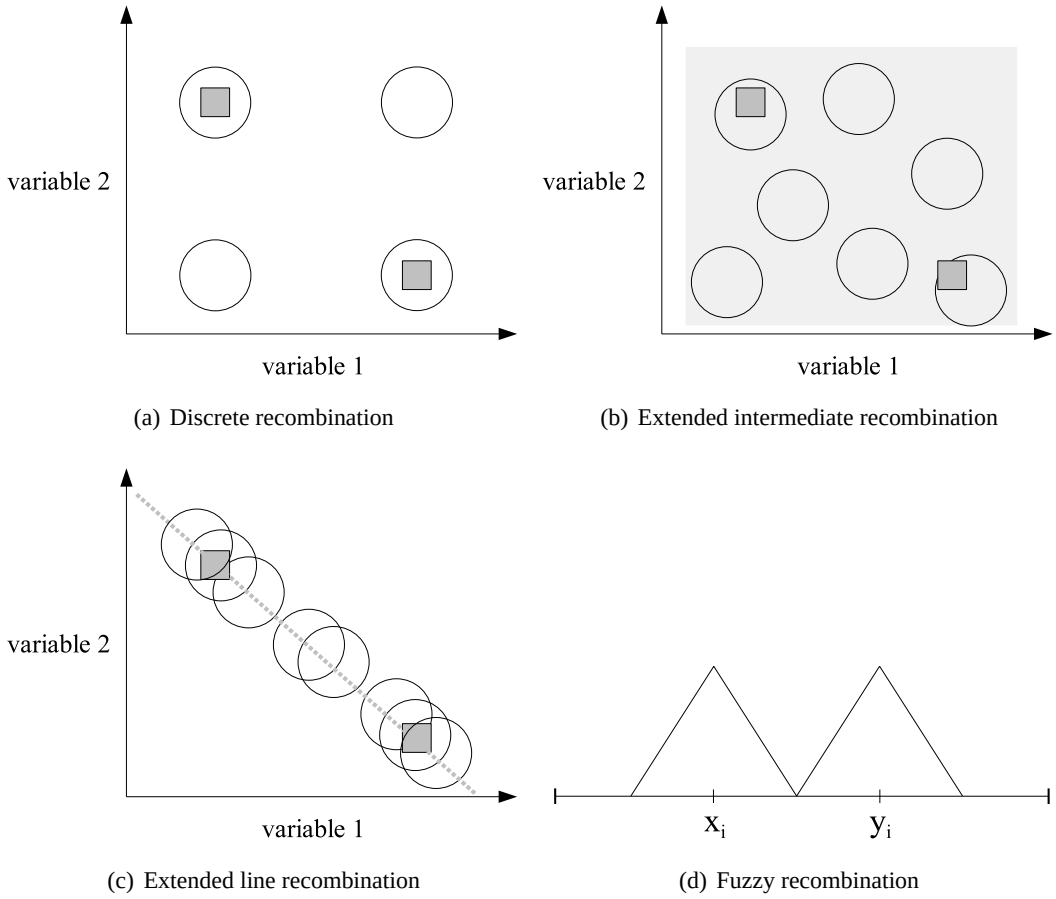
$$z_i = x_i + \alpha(y_i - x_i) \qquad i = 1, \ldots, n \tag{3.10}$$

Scaling factor $\alpha$ is again chosen uniformly at random over the interval $[-d, 1 + d]$.

Line recombination selects the offspring from a line segment between the parents. Extended line recombination scales the line segment before picking. An illustration of extended line recombination is shown in Figure 3.5(c).

[7] Gray coding maps integers to binary strings in such way that adjacent integers differ in only one bit position. A number of different Gray codes exist.

[8] generalized definition presented here is from [61], for original definition with fixed $d = 0.25$ see [41]




(a) Discrete recombination

(b) Extended intermediate recombination

(c) Extended line recombination

(d) Fuzzy recombination

**Figure 3.5:** Various real valued recombination operators. Circles represent examples of possible offspring, squares are the two parents participating in recombination. The light gray square in (b) represents the area of possible offspring. The dashed line in (c) represents the line segment of possible offspring. (d) shows the distribution of possible values for offspring variable $i$, when $d = 0.5$ and the variable $i$ values for parents are $x_i$ and $y_i$. $d$ is a parameter used to control the distribution width, see text for a more accurate definition. (a), (b) and (c) were inspired by the figures in [47], whereas (d) was influenced by [22].

Breeder Genetic Algorithm (BGA) in [40] defines an interesting biased line recombination variant, which they call the *BGA line recombination*. It is a combination of mutation and extended line recombination with a strong bias towards producing offspring outside the line segment defined by the parents. It is defined as:

$$z_i = x_i \pm range_i \cdot \delta \cdot \frac{y_i - x_i}{\| \mathbf{x} - \mathbf{y} \|} \tag{3.11}$$

where

$$\delta = \sum_{i=0}^{k-1} \alpha_i \cdot 2^{-i} \quad \alpha \in \{0, 1\}$$

and $k$ is a user defined precision constant (higher $k$ will produce more precise results, but at a cost of increased search time as more mutation will be happening close to the individuals). In 3.11 the $-$ sign is chosen with probability 0.9.

The bias towards producing offspring outside of the parent defined line segment is caused by the biased $-$ sign probability. It makes the BGA line recombination try new points in a direction defined by the parent points. In [40], the BGA line recombination is used in conjunction with discrete recombination and BGA mutation to produce a search method that has linear order of convergence for "unimodal functions and most of the popular multimodal testfunctions".

*Fuzzy recombination* [61] is another BGA related recombination operator, designed to maximize the *response to selection*[9] of the the recombination. The probability that the offspring has value $z_i$ is given by a bimodal distribution,

$$p(z_i) \in \phi(x_i), \phi(y_i), \tag{3.12}$$

with triangular probability distributions $\phi(r)$ having the modal values $x_i$ and $y_i$ with

$$\begin{aligned} x_i - d \cdot |y_i - x_i| \leq r \leq x_i + d \cdot |y_i - x_i| \\ y_i - d \cdot |y_i - x_i| \leq r \leq y_i + d \cdot |y_i - x_i| \end{aligned} \tag{3.13}$$

for $x_i \leq y_i$ and $d \geq 0.5$. In [61], $d = 0.5$ was mainly used.

An illustration of the fuzzy recombination can be found in Figure 3.5(d).

### 3.2.4 Mutation

After recombination offspring undergo mutation. Offspring variables have a low probability of being mutated by a mutation operator. Typically the probability of mutating a variable is set to be inversely proportional to the total number of variables in an individual. Based on their experimentation, Mühlenbein and Schlierkamp-Voosen [42] present the following rule of thumb: "The mutation rate $m = 1/n$ where $n$ is the size of the chromosome is almost optimal."

[9] Response to selection is defined as the difference between the population mean objective value of generation $t + 1$ and the population mean objective value of generation $t$ [41].

Bäck [3] recommends the same rate, but add that a variation of the mutation rate is useful in cases where the fitness function is a multimodal pseudoboolean function.

For variables using binary representation the mutation operation is simple. Each bit of the individual has a probability equal to the selected mutation rate to be flipped.

For efficient mutation of real valued variables, the probability of large and small mutations should not be equal. Biasing the mutation to favor small perturbations improves the search resolution near the current population points and allows the mutation to work on both local and global scales. *BGA mutation* [41, 40], the mutation operator of the Breeder Genetic Algorithm, is a popular real valued mutation operator. Unlike uniform or normal distributed mutation, BGA mutation avoids the need to adapt the mutation range in order to give a reasonable progress for small mutation rates [41]. BGA mutation can be defined as:

$$x_i = x_i \pm range_i \cdot \delta \tag{3.14}$$

where

$$\delta = \sum_{i=0}^{k-1} \alpha_i \cdot 2^{-i} \quad \alpha \in \{0, 1\}$$

and $k$ is a user defined precision constant. $range_i$ defines the mutation range and is normally set to 0.5 times the domain of definition of variable $x_i$. Each $\alpha_i$ is assigned to 1 with probability of $p_\delta = 1/k$. On average there will be just one $\alpha_i$ with value 1, so $\delta$ can be approximated by $2^{-j}$, where $j$ is selected uniformly at random from $0, \ldots, k-1$.

### 3.2.5 Reinsertion

Once the offspring have been produced by selection, recombination and mutation of individuals from the old population, it is time to reinsert them to the original population. Usually it is desirable to keep the number of individuals in the population constant, which means at least some of the old individuals must be removed from the population.

If so called *generational model*[10] of replacement is used, the reinsertion task becomes trivial. In generational model, the number of offspring produced is equal to the size of the original population and the offspring population completely replaces the original population.

Unfortunately, in generational model it is very likely that excellent individuals are replaced without producing better offspring and therefore valuable information is lost [47]. Therefore *incremental reproduction*, retaining a part of the old population, is often preferable, even though it inevitably carries the same kind of sampling errors as roulette wheel selection [19].

The simplest incremental scheme, producing less offspring than parents and replacing parents uniformly at random, is called *uniform reinsertion* [47]. While it reduces the probability that the best individuals are replaced, in many cases it would be preferable to ensure that the

[10] also known as *pure reinsertion*, and emphsteady-state reproduction

best individuals continue to next generation. *Elitist reinsertion* does just this. It replaces the worst parents with the offspring.

Elistism ensures that the subsequent population will have some fit individuals to direct the search. However, it alone does not guarantee that the overall quality of the population improves. To ensure this, it should be combined with *fitness-based reinsertion*. In basic fitness-based reinsertion, more offspring than needed is produced and only the best of them are inserted. In the combined elitist fitness-based approach a given number of worst individuals are replaced with the same number of most fit offspring. Pohlheim [47] recommends this approach, because it prevents losing of information. Note, however, that the approach directs the search very strongly to exploit the already found information, instead of exploring to find new solutions. This may lead to premature convergence, as it is highly unlikely that the first solution in a new area will be better than the best already found solutions, even though the area itself might contain extremely good solutions. Even worse, it is also possible that some part of the genome of a poor individual is extremely good, and combined with an earlier good individual would produce a superior solution.

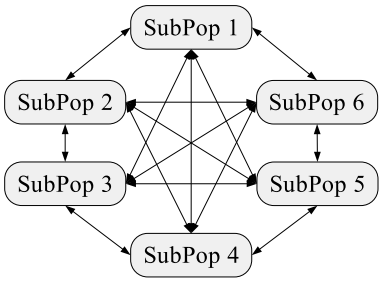
### 3.2.6 Multi-Population Genetic Algorithms

Balancing between avoiding premature convergence and having enough selective pressure to reach good solutions in reasonable time can be hard. Ideally we would like to have quite strong selective pressure to direct the search towards promising solutions, while still supporting exploration. However, too strong selective pressure can lead the population to converge around a single solution, which causes the search to stagnate as the recombination operators produce only similar solutions and mutation operators have an extremely low probability to hit areas with better fitness than that of the converged population.

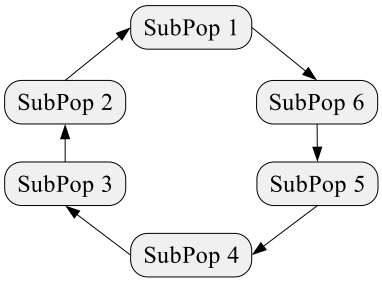
To some extent, this can be remedied by using *Multi-Population Genetic Algorithms*[11] (MPGA). In MPGA multiple subpopulations are used. These subpopulations evolve independently, exchanging genes only once per a certain number of generations (called *migration interval* [62] or *isolation time* [47]). The number of exchanged individuals (called *migration size* [62] or *migration rate* [47]), the selection method used to select the migrating individuals and the *migration topology* determine how much genetic diversity can occur in the subpopulations and the exchange of information between the subpopulations [47].

Each subpopulation maintains some degree of independence and thus explores a different region of the search space, which improves the search quality [62]. The migration constantly adds new genetic material to subpopulations and therefore reduces the probability of premature convergence. Even if a certain subpopulation converges, after migration it may restart the evolution process, especially if the individual that migrated has better fitness than the existing population and an elitist or a highly selective GA scheme is used. In addition, it can be

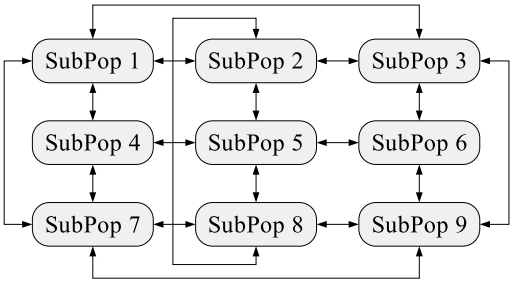
[11] Also known as *Island Model Genetic Algorithms*.




(a) Unrestricted migration topology




(b) Ring migration topology




(c) Neighborhood migration topology

**Figure 3.6:** Various multi-population genetic algorithm subpopulation topologies. By restricting subpopulations the individuals from a specific subpopulation are allowed to migrate to, it is possible to get a more fine grained control over speed with which the innovations propagate through the whole population. Figure after [47].

speculated that the independent subpopulations can concentrate on optimizing a certain part of the genome and incorporate the optimizations made to other parts by other populations during migration and the recombination phase following it.

When designing MPGA's, migration topology is an important consideration. It defines where the individuals from the subpopulations may migrate. The most general migration hierarchy is the *unrestricted migration topology*, in which the migrating individuals are free to move to any other subpopulation. The most basic is the *ring migration topology*, where the subpopulations are organized as a directed ring and migrate to the next subpopulation in the ring. In *neighborhood migration topology*, the subpopulations are organized as a toroidal lattice and can migrate to any of the neighboring subpopulations. Migration topologies are illustrated in Figure 3.6.

There are two typical ways to select the migrating individuals in MPGA's. They are picked either uniformly at random or based on fitness (the best individuals migrate). Usually the worst individuals of the receiving subpopulation are replaced by the migrating individuals.

In addition to topology and migration type, subpopulation size and migration rate must be selected. Typically symmetrically sized subpopulations are used and therefore the problem of selecting subpopulation size is analogous to the problem of selecting the population size in

single-population GA's. For migration rate, wide variety of values have been used. Belding [4] explores migration rates of 10%, 20% and 50% using Royal Road problems[12]

According to Whitley *et al.* [62], Multi-Population Genetic Algorithms have often been reported to display better search performance than serial single population models, both in terms of the solution quality, as well as the total number of evaluations required to reach it. In [4], MPGA's outperform SPGA's in some of the Royal Road problems with regard to the metrics of best fitness, average fitness, and number of times the optimum was reached.

Many efficient parallel implementations of MPGA's have been made. For example, Belding [4] reports superlinear speedup on two different parallel computer configurations.

## 3.3 Neuroevolution Using Evolutionary Algorithms

The goal in neural network training is to adapt the weights so that the network performs a desired action, that is, approximates such (unknown) input-output mapping function that produces output which leads to desired performance of the system controlled by the neural network. The approximation quality is determined by a user defined error metric, which measures how well the system reaches its goals. Therefore, neural network training can be seen as an optimization task, where the weights of the neural network are optimized to minimize the training error function. As evolutionary algorithms have turned out to be good tools for function optimization, it is reasonable to try to use them for NN training as well. It turns out that in many cases using evolutionary algorithms instead of ordinary gradient descent based training is preferable.

### 3.3.1 Why Neuroevolution Instead of Gradient-Based Methods?

Artificial evolution methods use large population to search for solutions, so in a sense they are evolving a number of solutions whereas a gradient descent method only iterates one. Naturally, this involves an extra computational cost. In addition, there is still relatively little knowledge about the *modus operandi* of the evolutionary algorithms, whereas the way the gradient descent algorithms learn is rather well known. Thus more proven knowledge on efficient usage of gradient descent methods exists, whereas much of the knowledge on applying EA's is derived from experimental studies. Therefore the natural question is, why and when should neuroevolution be used?

One of the biggest problems with gradient descent based NN training methods is their susceptibility to getting trapped in local minima, instead of finding the global minimum. Evolu-

[12] Royal Road fitness landscape functions were devised by Mitchell *et al.* [37]. They are idealized problems in which certain features most relevant to GA's are explicit, so that GA's performance can be studied in detail. Originally, they were expected to be extremely simple for the GA's to solve, because it was thought that their structure would lay out a "royal road" for the GA to follow to the optimal solution. However, the results were exactly the opposite. GA's performed rather poorly. For further details, theoretical discussion and analysis of the possible reasons, [50] provides an excellent starting point.

tionary algorithms avoid this problem, because they are not dependent on gradient information. In fact, as their operation does not explicitly depend on the error function at all, they can treat large, complex, discontinuous, non-differentiable and multimodal spaces, which are typical for the real world problems [63]. This error function independence also means high modularity for implementations. The error functions, operating environments and even actual tasks the system is performing can be changed without a need for altering the way the learning system operates (although some parameter tweaks are typically necessary).

The evolutionary algorithm based training is also independent of the topology and operation of the network being trained. As the networks are simply sets of parameters from the EA point of view, the type of the network being trained can be changed at will. EA's are capable of training networks for which defining other types of training methods is complicated, such as recurrent and higher order NN's or spiking neural networks (SNN) [33].

The evolutionary approach also makes it easier to generate NN's with special characteristics. For example, the NN's complexity can be decreased and its generalization capability increased by including a complexity (regularization) term in the fitness function. Weight sharing and weight decay can also be incorporated easily. [63]

In addition, for some problems evolutionary training has been reported to be significantly faster and more reliable than backpropagation based methods [63]. For reinforcement learning problems, such as pole balancing and robot arm control, neuroevolution has been able to produce better results more efficiently than the traditional reinforcement learning methods such as Q-Learning or Adaptive Heuristic Critic [39, 38].

### 3.3.2 Evolutionary Neural Network Training Approaches

A high level description of the neural network training using evolutionary algorithms can be given as follows:

1. Initialize a population of neural networks using some neural network weight initialization method, for example the Nguyen-Widrow [43] method.
2. Evaluate task performance for each NN in the population.
3. Construct genotype[13] vectors for the neural networks.
4. If the desired end condition is not reached, evolve next generation using the selected EA.
5. Construct neural network from genotype vector and continue from step 2.

[13] Originally biological terms, *genotype* refers to the genetic description of the individual, whereas *phenotype* is a realization of an individual from a specific genotype. An obvious analogy from object oriented programming is class (genotype) and instance (phenotype). Here, by genotype we mean the description used by the EA to evolve solutions, and by phenotype, the neural network built from a specific genotype.

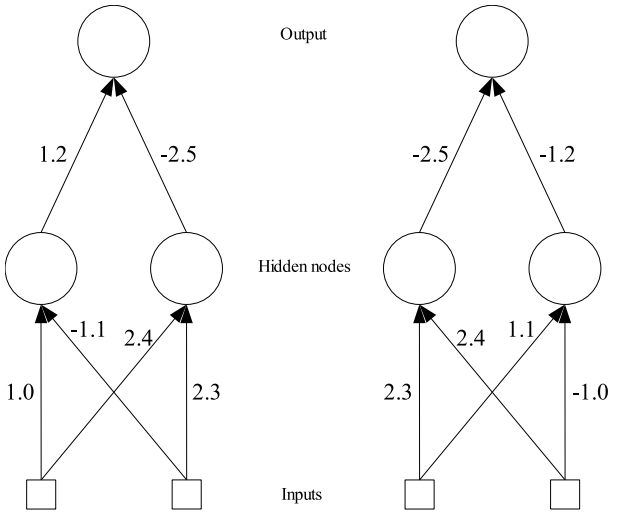



**Figure 3.7:** Two functionally equivalent feedforward neural networks. The hidden nodes have been swapped and one of the nodes has had the signs of all its weights flipped. Recombination is problematic because the functionally equivalent nodes are not in the same place and will therefore get recombined with functionally incompatible neuron, even though there exists a better recombination partner. After [9].

The naive approach would be to simply construct a genotype vector from neural network by concatenating its weights. However, this leads to a problem called the *competing conventions*[14] problem.

The competing conventions problem stems from the fact that it is possible to construct many different functionally equivalent neural networks. For example in fully connected MLP, the order of hidden layer neurons may be permuted freely and the network retains its functionality. In addition, the signs of the hidden neuron weights and the sign of its bias may be inverted, if the activation function is odd symmetric (such as $\tanh$). Thierens [59] shows that in this case, there are $2^n n!$ functionally equivalent $n$ hidden neuron neural networks. This redundancy in the network representation makes the recombination problematic. The following example, originally from [9], clarifies the issue:

> *Consider the two feedforward networks in Figure 3.7: if they use an odd symmetric activation function, they are functionally identical; two hidden nodes have been swapped, and all of the weights associated with one of them have been multiplied by -1. Despite being functionally identical, the networks will appear very different in genetic algorithm. Crossover will produce weak offspring, because the corresponding parts are not in corresponding places.*

[14] also known as *label convention problem* or *permutation problem*

Thierens [59] proposed a way to avoid functional redundancy problem by sorting the nodes before crossover. He uses the following method: First, for hidden neurons with bias < 0, flip signs of all weights (including bias). Next, sort neurons in hidden layers into increasing bias order. While this approach avoids the multiple representations redundancy, it still does not seem to avoid the whole competing conventions problem. For example, if there are two otherwise almost identical neurons in two networks, which only differ slightly by their bias term, their position in the genotype vector may differ and they can be recombined with wrong neurons. Or if the two neurons are identical, but one of the networks has more smaller bias terms in its hidden layer than the other, the positions will again differ.

The problem with the approach proposed by Thierens is that while it provides unique reprensentation for neural networks, it does not ensure that similar neurons in otherwise differing networks will get recombined. To adequately solve the problem, it would be preferable that similarly functioning neurons were recombined. In section 3.4, some methods capable of this are reviewed.

The usual way of dealing with the competing conventions problem in simple ordinary EA based neuroevolution is to simply avoid doing recombination altogether, or to invent tailored genetic operators [9]. If recombination is left out, Genetic Programming (GP) and Evolutionary Strategies (ES) are good choices, since their primary search operator is mutation and therefore the negative impact of the permutation problem is reduced [63].

### 3.3.3 Hybrid Training

While evolutionary algorithms perform global search well, gradient based methods are often better in locally optimizing the found solution. Therefore, if the error function permits the use of gradient based methods, it makes sense to try a hybrid approach. In the hybrid approach, the efficiency of the evolutionary training is improved by incorporating a local search procedure into the evolution. EA's are used to locate a good region in the search space and then a local search procedure is used to find a near-optimal solution in this region. Hybrid training has been used successfully in many application areas. [63]

## 3.4 Specialized Neuroevolution Methods

As described in the previous section, competing conventions is an important problem that effectively prevents having useful results in neural network training with recombining EA's. As discussed, the problem is partly[15] caused by the fact that in fully connected feedforward network the hidden layer neurons may be freely permuted, while still retaining the same network

[15] As mentioned in section 3.3.2, the other part of the problem is the sign flipping redundancy when using odd symmetric activation. This part of the problem can be easily avoided by avoiding the use of odd symmetric activation functions, for example by using an ordinary sigmoid or logarithmic sigmoid instead.

functionality. If this problem could be avoided, using recombination in neuroevolution would become feasible.

### 3.4.1 Symbiotic, Adaptive Neuro-Evolution

While neural networks have redundant representations, individual hidden neurons have no such problems. If we take a hidden neuron and all of its incoming and outgoing weights, the representation for the functionality is unique (as long as nonsymmetric activation function is used). Symbiotic, Adaptive Neuro-Evolution (SANE) [39, 38] does just this. Instead of evolving whole neural network, it separately evolves neurons and neural network blueprints, which specify which neurons should be connected together to form the network.

In each generation, networks are formed according to the blueprints, and evaluated on the task. Neurons compete on the basis of how well, on average, the networks in which they participate perform. A high fitness means that the neuron contributes to the successful networks and, therefore, suggests that it cooperates well with other neurons. The blueprints evolve to effectively combine the well working neurons to produce even better results. This coevolution of neurons and topologies evolves good networks more quickly, because the network subfunctions are allowed to evolve independently [16]. Since neurons are not tied to a single network, a neuron that may be useful is not discarded if it happens to be part of a network that performs poorly. Likewise, bad neurons do not benefit from being part of a high scoring network. Instead of trying to produce a monolithic solution network, SANE breaks the problem down to that of finding solutions for smaller, interacting subproblems.

In addition, evolving neurons instead of full networks maintains diversity in the population. If a single neuron type becomes prevalent, it will not as often be combined with other types of neurons in the population. Therefore, because difficult tasks usually require several different types of specialized neurons, the dominant neuron type will receive less fitness and thus, less offspring, bringing diversity back into the population.

Unfortunately, SANE still suffers from the competing conventions problem. Because blueprints describe complete networks, the recombination at blueprint level is subject to competing conventions.

There are also problems with the neuron level recombination. Since the neurons are mated together as a single population, budding specializations may get destroyed by the recombination. In addition, SANE's ability to evolve recurrent networks is limited, as a neuron's behavior in a recurrent network depends critically upon the neurons to which it is connected, and in SANE it cannot rely on being combined with similar neurons in any two trials.

### 3.4.2 Enforced Subpopulations

Enforced Subpopulations (ESP) [16] addresses the problems of SANE. Like SANE, it is a neuron-level cooperative coevolution method, i.e., evolves neurons instead of full networks and combines them to form a complete network. However, in ESP, subtasks are explicit: each neuron in the neural network has its own subpopulation, and a neuron can only be recombined with members of its own subpopulation. This way the neurons in each subpopulation evolve independently, and the subpopulations rapidly specialize into some network sub-function. This avoids the competing conventions problem and makes evolving recurrent networks with ESP feasible.

ESP can be used to evolve any single hidden layer neural network, such as feed-forward, simple recurrent (Elman), fully recurrent, and second-order networks. Topology is not evolved and therefore must be specified beforehand. Typically fully-connected topology is used. The evolved genotype consists of a vector of real numbers that represent connection weights. The evolution cycle in ESP has the following steps:

1. *Initialization.* For each hidden neuron, an initial population consisting of $n$ individuals is created. Each individual contains the input, output and possibly recurrent connection weights of a neuron, intialized with a random string of real numbers. (Alternatively, some NN initialization method could be used to initialize the population.)

2. *Evaluation.* A hidden layer of the neural network is formed by selecting one neuron from each subpopulation at random. Task performance of the network is evaluated. The achieved score is added to the cumulative fitness of each neuron that participated in the network. This process is repeated until each neuron has participated in an average of, for example, 10 trials.

3. *Check Stagnation.* If the performance of the best network has not improved in $b$ generations, *burst mutation* is performed. In burst mutation, the best network is saved and new subpopulations are created by adding Cauchy-distributed noise to each of the neurons in the best solution.

   If the performance has not improved after two burst mutations the network size is adapted. In *size adaptation*, the best network found so far is evaluated after removing each of its neurons in turn. If the fitness of the network does not fall below a threshold when missing some neuron, then it is not critical to the performance of the network and its corresponding subpopulation is removed. If no neurons can be removed, a new subpopulation of random neurons is added and the size of the network hidden layer increased to include the new subpopulation. This way, ESP will adapt the size of the network to the difficulty of the task. This makes it more robust in dealing with environmental changes and tasks where the appropriate network size is difficult to determine.

4. *Recombination.* The average fitness of each neuron is calculated by dividing its cumulative fitness by the number of task performance evaluations it participated in. Neurons in each subpopulation are then sorted by their average fitness. Each neuron in the top quartile is recombined with a higher-ranking neuron using single point crossover and Cauchy-distributed mutation. The offspring replace the lowest-ranking half of the subpopulation.

5. *End Condition.* If the best found network performs sufficiently well in the task, the cycle ends. Otherwise, the cycle continues from step 2.

Steps 1,2,4, and 5 are the backbone of the algorithm. Step 3 is used to avoid premature convergence. The burst mutation in a sense reinitializes the search around the best currently found solution, therefore searching for optimal modifications to it. The size adaptation is both a form of complexification [16] and a form of pruning. It prunes unneeded complexity or, if none is found, adds complexity to provide more solution flexibility.

The subpopulation architecture in ESP provides it advantages over SANE's single population. The subpopulations specialize for specific tasks and the neurons only recombine with other neurons evolving for the same task. In addition, coevolution is more efficient, as the networks are always guaranteed to have neurons for each subtask, whereas SANE may form networks that contain multiple members of some specializations and omit members of others. This is especially useful when evolving recurrent networks, as a neuron's specific recurrent weight will always be associated with a specific subpopulation. As the subpopulations specialize, neurons evolve to expect the kinds of neurons to which they will be connected [16].

While ESP is very efficient in finding good solutions, as the pole balancing comparisons in [16] show, it does not evolve topology. Even though different pruning methods may be used to minimize the solution network, evolving computationally more efficient networks might be possible, if also the topology was evolved.

### 3.4.3 Neuroevolution of Augmenting Topologies

NeuroEvolution of Augmenting Topologies (NEAT) [56] is a Topology and Weight Evolving Artificial Neural Network (TWEANN). In addition to evolving connection weights, it evolves the structure, or topology, of the network. Starting from a minimal topology (all inputs connected directly to all outputs), NEAT simultaneously evolves the weights and incrementally complexifies the topology to produce a near minimal solution for the problem. This minimality gives it a performance advantage compared to other neuroevolution approaches. In addition, by evolving topology, NEAT avoids the need to pre-select it. This is a considerable merit, because when dealing with difficult problems, topology selection can be a difficult and time consuming process.

[16] the incremental elaboration of solutions through adding new structure [57]

The main problem in topology evolving is how can individuals of different topologies be recombined? In addition, competing conventions becomes a much more complex problem when evolving topology is allowed, because a certain function can be realized with different topologies. NEAT solves these problems by using *historical markings*, which keep track of the historical origin of each gene. Tracking the historical origin in NEAT is very simple. Whenever new structure is evolved, a *global innovation number* is incremented and assigned to the corresponding new gene. This innovation number uniquely identifies each topological innovation. Therefore, if historical markings of the two genes match, they can be recombined.

The recombination of two individuals becomes easy. Simply line up their genes, sorted by their innovation numbers, and perform recombination between genes that both individuals have. In NEAT recombination, offspring inherits matching parent genes randomly. Disjoint (genes in the middle of the genome that do not match) and excess (unmatching genomes at the end) genes are inherited from the more fit parent, or in case of equal fitnesses, randomly. The NEAT recombination is illustrated in Figure 3.8.

In NEAT, an individual consists of two kinds of genes: node genes (corresponding to the input, output and hidden nodes in the neural network) and connection genes (corresponding to the links between the nodes in the neural network). Node genes provide a list of inputs, hidden nodes and outputs that can be connected (initially only inputs and outputs). Connection genes specify the in-node, out-node and weight of the connection, whether or not the connection gene is expressed (enabled), and the innovation number. An example of a NEAT genome and the corresponding network is given in Figure 3.9.

As the NEAT networks start minimal, topology mutation operators have an important role. They complexify the solution by creating more connections and nodes. There are three types of mutations in NEAT: *add node mutation*, *add connection mutation* and *connection weight mutation*.

In the add node mutation, an existing connection is split and a new node is placed in between its nodes. The old connection is marked disabled and two new connections are created. The connection leading into the new node receives a weight of 1, and the new connection leading out receives the same weight as the old connection. This way the adding of the node has minimal initial effect. The new nonlinearity in the connection changes the function slightly, but otherwise it acts like the old connection. Note that disabling the old connection instead of simply removing it is important. Even disabled genes are used in recombination, so there is a chance that disabled gene becomes enabled again in future generations.

In the add connection mutation, a single new connection gene connecting two previously unconnected nodes with a random weight is added. The connection weight mutation is similar to weight mutations in other neuroevolution systems.

When a topological mutation changes the structure of the network, it typically initially reduces the fitness of the solution. Therefore, recently augmented structures would have little

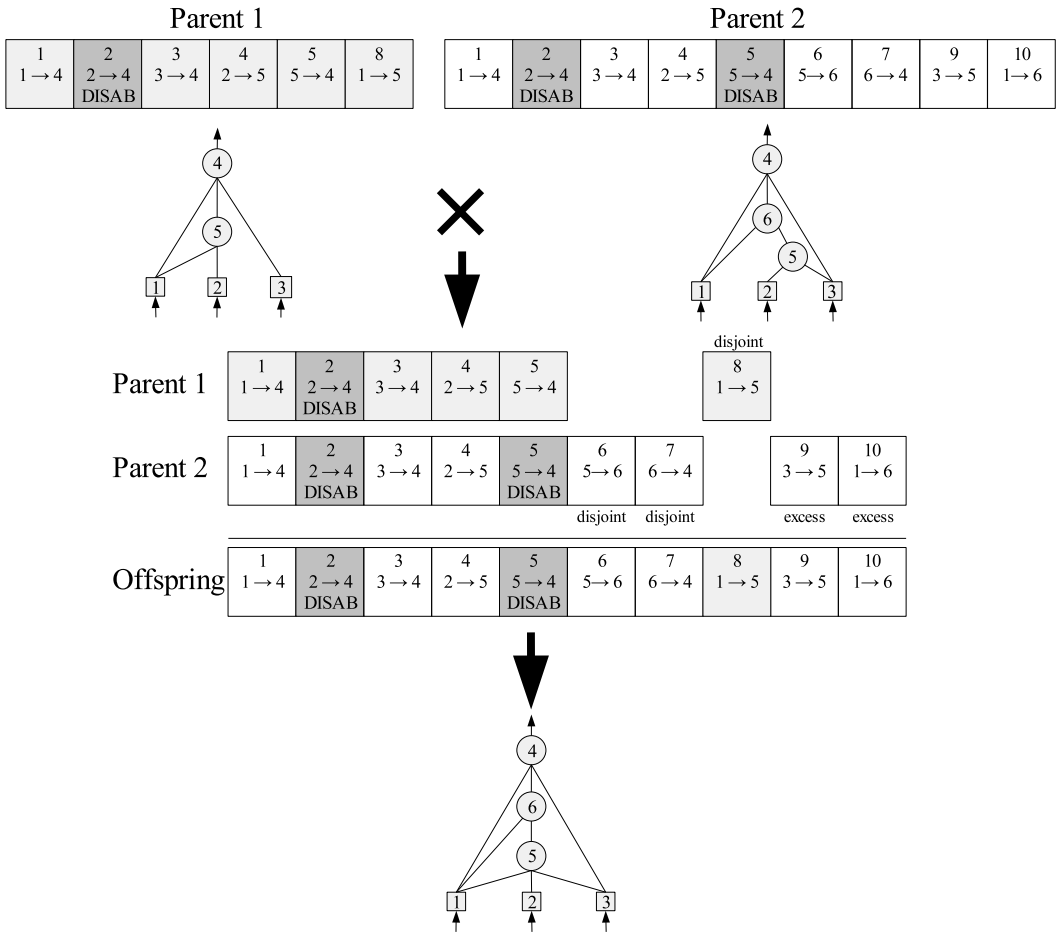



**Figure 3.8:** In NEAT recombination, genomes are matched up using innovation numbers. Although the individual topologies differ, their innovation numbers reveal which gene matches which. Without any topological analysis, a new structure that combines the overlapping parts of the two parents as well as different parts can be created. In NEAT, matching genes are inherited randomly, while disjoint and excess genes are inherited from the more fit parent. Here equal fitness is assumed, so the disjoint and excess genes are also inherited randomly. The disabled genes may become enabled again: if a gene is disabled in only one of the parents, there is a preset chance that it will be enabled in the offspring. [56]

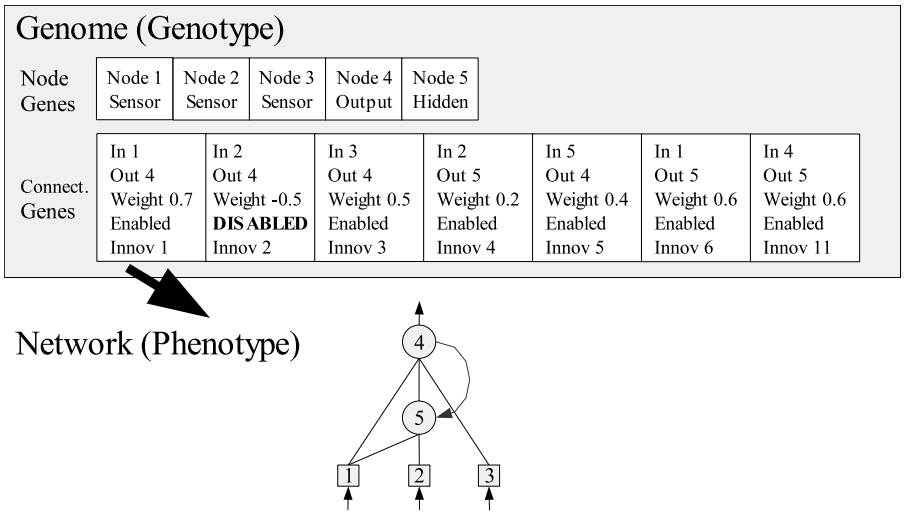



**Figure 3.9:** A genome with five node genes (3 input, 1 hidden and 1 output), and seven connection genes (one of which is recurrent), and the corresponding phenotype. The second connection gene is disabled, so the connection it specifies is not expressed in the phenotype. [56]

hope of surviving, even if their innovation was a crucial one, if NEAT would not protect innovation through *speciation*. The idea of speciation is to divide the population into species, which consist of networks with similar topologies. The networks compete mainly within their own species instead of with the population at large. This protects the innovations, as significant topology changes separate the offspring into its own species, thus giving it time to optimize its performance.

NEAT determines the similarity of two individuals by measuring the number of excess and disjoint genes between them. The compatibility distance $\delta$ of different structures in NEAT is defined as a linear combination of excess $E$ and disjoint $D$ genes, and the average weight differences of matching genes $\overline{W}$, including disabled genes:

$$\delta = \frac{c_1 E}{N} + \frac{c_2 D}{N} + c_3 \cdot \overline{W} \tag{3.15}$$

The coefficients $c_1$, $c_2$, and $c_3$ can be used to adjust the importance of the three factors, and the factor $N$, the number of genes in the larger genome, normalizes for genome size ($N$ can be set to 1 if both genomes if both genomes are small, for example consist of fewer than 20 genes).

The speciation in NEAT is implemented using a compatibility threshold $\delta_t$. An ordered list of species is maintained. In each generation, genomes are sequentially placed into species. Each existing species is represented by a random genome from the previous generation of the species. An individual is placed in the first species which it is compatible with, that is, for which its similarity is below the threshold, $\delta < \delta_t$. This way, species do not overlap. If an individual is not compatible with any species, a new species is created, with the individual as its representative.

*Explicit fitness sharing* [15] is used to control the size of the species. All individuals in the same species must share the fitness of their niche. This limits the size of the species, even if many of its individuals perform well. This way any one species cannot take over the entire population, which is crucial for speciated evolution to work. The sharing adjusted fitness $f'_i$ for individual $i$ is calculated according to its distance $\delta$ from every other organism $j$ in the population:

$$f'_i = \frac{f_i}{\sum_{j=1}^{n} \mathrm{sh}(\delta(i,j))}, \tag{3.16}$$

where sharing function $\mathrm{sh}$ is set to 0 when distance $\delta(i,j)$ is above threshold $\delta_t$; otherwise, $\mathrm{sh}(\delta(i,j))$ is set to 1. Thus, the sum in the denominator reduces to the number of organisms in the same species as individual $i$.

Every species is assigned a number of offspring in proportion to the sum of adjusted fitnesses $f'_i$ of its members. The lowest performing members of the species are eliminated from the population and the entire population gets replaced by the offspring of the remaining organisms in each species.

Using history markings, speciation induced innovation protection and minimal topology starting, NEAT is able to evolve solutions for difficult non-Markovian tasks such as double pole balancing without velocity information (DPNV) remarkably efficiently (experiment details and comparisons to other methods can be found in [56]). The performance of NEAT is comparable to that of ESP if a good starting topology for ESP is used. While NEAT avoids the need to specify topology, it requires considerably more parameters than ESP (23 against 5) [16]. Luckily, NEAT performance is robust for moderate parameter variations [56].

In addition to increasing search efficiency, the topology evolution in NEAT may improve the computational speed of the end solution network, as minimal topology means minimal computational costs. This has some relevance when evolving networks for use in devices of low computational power, such as mobile phones. The automated topology evolution also relieves network designer from the search of optimal topology, which is typically a difficult task, especially for recurrent networks.

*Modular NEAT* [51] is a recent NEAT modification which directs genetic search to evolve reusable modules and blueprints which show how to combine them. Modular NEAT could be described as SANE which uses NEAT evolved multiple input, multiple output network modules instead of neurons, and is able to use a single module in multiple, possibly overlapping, positions. Figure 3.10 illustrates the way Modular NEAT blueprints combine NEAT evolved modules to create neural networks. In [51], Modular NEAT was tested in game playing domain containing symmetries, and produced better quality results four to six times as efficiently as ordinary NEAT.

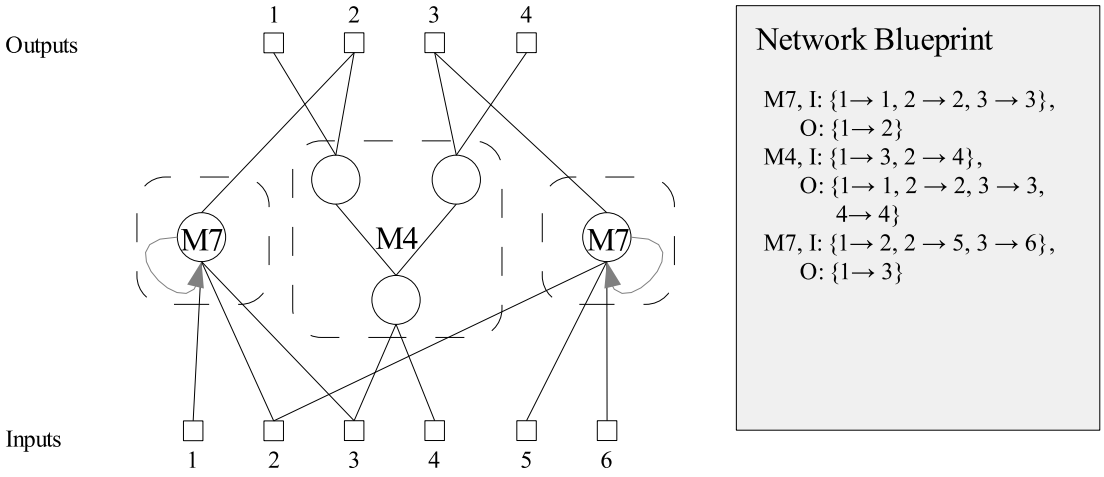



**Figure 3.10:** An example of Modular NEAT blueprint combining NEAT evolved modules to produce a neural network. Two three input, one output modules containing a hidden neuron with recurrent connection (M7); and a two input, four output module containing three hidden neurons (M4) are combined to produce a neural network. The blueprint specifies the used modules and how they are binded to form the network. (In the picture, the bindings are given as sets of *abstract neuron → concrete neuron* connections.)

# Chapter 4

# Neuroevolution of Artificial Bandwidth Expansion

The NeuroEvolution based Artificial Bandwidth Expansion (NEABE) algorithm presented here was designed to be used in digital speech transmission systems. The main design goals were to create an algorithm that would double the sampling frequency from that used in current speech transmission systems (ie. from 8 kHz to 16 kHz), add new frequency components to the highband of the signal and leave the original narrowband untouched. Additionally the online portion of the algorithm was required to be able to run in real-time using ordinary DSP hardware.

The idea of using neuroevolution to reach these goals was born when the preliminary tests in tuning the magnitude shaping parameters of a prior Artificial Bandwidth Expansion (ABE) algorithm (see chapter 2.1.3, described more thoroughly in [30]) using genetic algorithms (GA) proved to be very successful.

The prior ABE version used fixed logic to decide type of the speech signal frame (voiced, fricative or stop consonant) and expanded the frame according to its type. This, however, is not very accurate because the transitions between different phones in speech are very smooth and thus hard limits create unneeded discontinuities to the expansion process. It would be better if the classification was adaptive and the transition between the classes smooth. Even better, it would be beneficial if the problem of frame classification could be avoided altogether. Frame classification is not really needed to solve the bandwidth expansion problem, because it is not necessary to know the frame class to be able to expand it. In other words, it is not necessary to recognize speech to be able to expand the speech bandwidth.

Unfortunately, dropping the frame classification leads to significant problems in expansion method design. In the context of the old ABE method, it would mean selecting the highband magnitude shaping curves and the parameters that are significant for tuning them, without using any strict phoneme classes, which would be very difficult at best. As parameter tuning

using evolutionary methods had been successful, it made sense to try to extend the idea to evolving not only parameters, but the whole expansion method. For this, a generic function approximator would be needed. Neural networks seemed to be the most natural choice as they have been used for speech processing and even artificial bandwidth expansion before (see chapter 2.2.6). In addition, there has been a considerable amount of research on neuroevolution, the process of evolving neural network controllers for different control tasks, for example in robotics and process industry.

Thus the decision to try to use neuroevolution for artificial bandwidth expansion was made. To test the concept as efficiently as possible, it was built on the principles of the old ABE algorithm as much as possible. This, as will be shown later in this thesis, lead to some choices that probably are not ideal in the sense of expansion result quality, but which made it possible to test the concept faster. Hence, despite the good results for some types of speech data, this work should be seen more as a proof of concept than as a industrial grade solution to the bandwidth expansion problem.

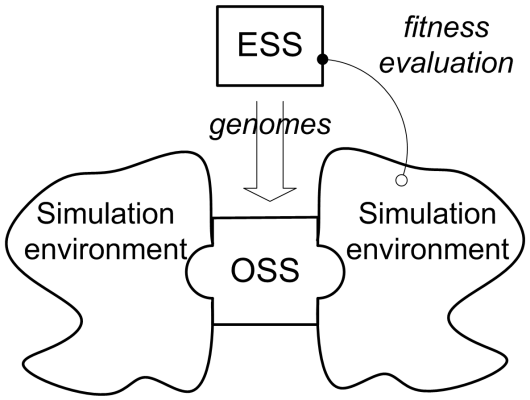
This chapter introduces the NEABE algorithm. First section gives a conceptual overview of the NEABE system, explaining the general idea behind it. Second section describes the evolution subsystem of NEABE in detail and third the online subsystem. A detailed explanation of the current NEABE implementation and its parameters is given in section four.

## 4.1 Conceptual Overview

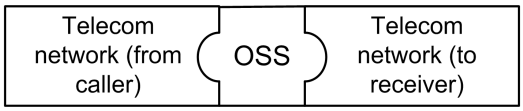
As is typical for neuroevolution systems, the NEABE system consists of two subsystems. There is the evolution subsystem (ESS) which evolves individuals for the online subsystem to use. The online subsystem (OSS), as its name implies, is the real-time component of the system. It handles the actual bandwidth expansion procedure, using the genome passed by the evolution subsystem to configure its modules to expand the bandwidth of the speech signal.

The basic idea is to use the evolution system to define what is needed for the expansion to succeed. After the evolution process has found a solution good enough for the problem in a simulation environment, the online subsystem can be implemented in the actual target environment (telecommunication network in this case) and the evolved genome used to perform the artificial bandwidth expansion.

This process is illustrated in Figure 4.1. The evolution subsystem generates a random population of genomes and, using the online subsystem, expands a predefined set of learning samples with each of them. It calculates a fitness value for each of the individuals by evaluating an objective function, which measures the quality of the expansion result using some metric appropriate for the problem. It then evolves the population, recombining and mutating the individuals. This evaluation–evolution cycle is continued until a specified end condition is reached.




(a) Evolution subsystem (ESS) searches a solution for the problem in simulation domain by testing different genomes in online subsystem (OSS) and evaluating their fitness.




(b) The evolved genome is used as part of the real world telecommunication system to perform artificial bandwidth expansion.

**Figure 4.1:** Evolution subsystem configures online subsystem to solve the problem in simulation environment. The solution is transferred to the real operating environment.

After an acceptable solution has been found, its genome is permanently merged with the online subsystem to form the final expansion algorithm. Should the processing environment change, a new genome can be evolved later using the same evolution process in a properly altered simulation environment, and installed to the actual processing system replacing the old genome.

This leads to considerable adaptability. The solution can be adapted to the language(s) spoken in the target telephone system, thus improving the quality of the expansion by avoiding the need for the algorithm to be a "jack of all trades". Also multiple solutions could be used, using a fixed control logic to select between them. For example there could be one genome for a noisy environment and another for a less noisy one, and a control logic to monitor when to switch from one to the other.

The final solution can also be optimized if needed, for example by pruning[1] the neural network.

## 4.2 The Evolution Subsystem

The structure of the Evolution Subsystem is presented in Figure 4.2. The Evolution Subsystem consists of three main modules: Learning Sample Management Module (LSMM), which man-

[1]Pruning is the name given to the process of examining a solution network, determining which units are not necessary to the solution and removing those units [54]. This is often done by determining non-essential units by a form of sensitivity analysis, in which the value of a unit is set to zero and the impact of the change is evaluated by testing with all test samples. In this case, pruning a network would require listening the impact to the test samples.

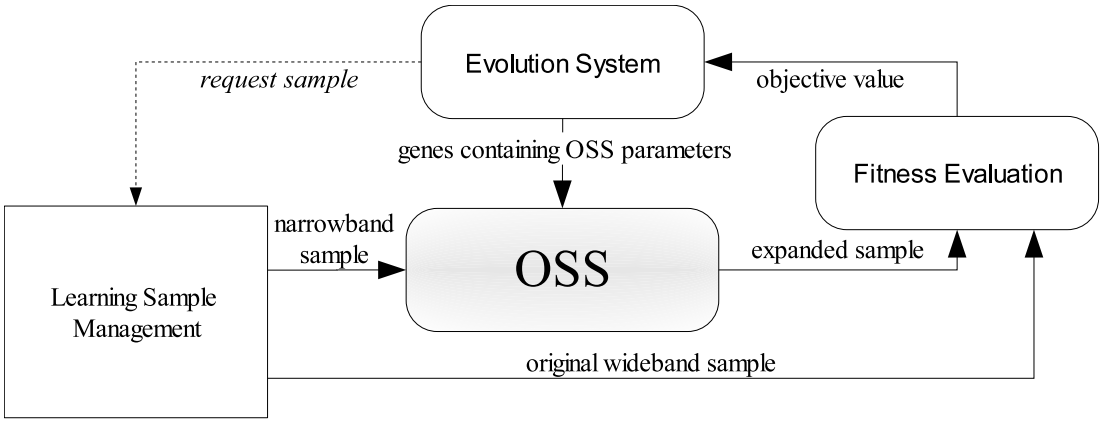



**Figure 4.2:** The structure of the Evolution Subsystem. Online Subsystem (OSS) in the picture is a major subsystem that contains the modules needed for standalone operation in the final operation environment.

ages the speech samples that are used to train the system; Fitness Evaluation Module, which evaluates the quality of the expansion made by the online subsystem, using some metric that measures the psychoacoustic quality[2] of the expanded sample as accurately as possible; and the actual Evolution Module which performs the artificial evolution by mutating and recombining the best performing individuals.

The modules have relatively few interdependencies, so it would be possible to replace one of them without changing the others, if certain simple interfaces are adhered to. This makes the system flexible and enables separate development of the modules.

The basic algorithm of the ESS learning process can be compressed into the following steps:

1. Produce the initial population of solutions in Evolution Module.
2. For each of the genomes

   For each of the samples in the Learning Sample Management Module

   a. Process the narrowband sample with the Online Subsystem, configuring it using the current genome.
   b. Evaluate the fitness in Fitness Evaluation Module, comparing the expansion result with the reference signal received from the LSMM. Produce an objective value that will be used by the Evolution Module to create a fitness ranking for the genomes.
3. Rank the genomes by their objective values and select genomes for reproduction using some rank based selection method.
4. Generate offspring by letting the selected genomes reproduce using mutation and crossover.

[2] similarity to the original wideband sample in the psychoacoustic sense

5. Replace part of the population with the produced offspring.

6. Test if one of the end conditions is met (one of the genomes produces a good enough solution or a prespecified iteration limit is encountered). If not, continue from 2 with the new population.

### 4.2.1 Learning Sample Management Module

This module handles the preprocessing of the samples in the training simulation. It simulates the processes a telephone speech signal goes through when it is transmitted from the speaker to the receiver. It is responsible for providing the narrowband signal to the online subsystem during the training simulation, and providing the corresponding wideband reference signal to the fitness evaluation module.

During preprocessing, the samples are transformed from 16 kHz wideband signals to 8 kHz narrowband signals. The system must avoid introducing any processing delay or the delay must be countered somewhere during the teaching process, because for the fitness function to be successful the wideband- and extended signals fed to it must be as synchronized as possible.

There are three processing paths in the current NEABE implementation. One simulates a call from a GSM mobile phone, another one a call from an ordinary PSTN phone. The third one is a simple passthrough processing path, which simply downsamples and lowpass filters the wideband signal to produce the narrowband signal.

As the online subsystem uses frame based processing, the narrowband signal is split into frames containing 10 ms of speech. There is some overlap between the frames to reduce the effect of FFT windowing and to enable linear averaging between the frames to avoid sudden jumps at frame edges. Hence, the actual processed frames are 98 samples (12.25 ms) long.

### 4.2.2 Evolution Module

The Evolution Module is responsible for evolving new genomes that will act as parameters for the Online Subsystem. It also acts as a process controller for the learning process, directing the other modules of the Evolution Subsystem.

There are multiple possible evolution methods that could be used. As the Online Subsystem of the current implementation uses a neural network to calculate the magnitude shaping parameters and does not have any other parameters to set by evolution, an evolution mechanism specialized for neuroevolution like ESP [16] or NEAT [56] could be used. These benefit from a faster, more efficient and flexible evolution, as they take the structure of the neural networks into account. NEAT would also relieve us from having to specify the structure of the neural network beforehand.

Nevertheless a simple unrestricted migration topology multipopulation GA was used in the current proof-of-concept implementation to save time, because a generic GA software library

was readily available. For ESP and NEAT, only implementations geared towards solving some specific problem exist.

A GA with multiple subpopulations was selected in an attempt to better preserve the population diversity and to enable multiple concurrent evolution processes with limited information exchange, so the different populations could concentrate on different aspects of the problem. Migration between the subpopulations is fitness based, $migration\_rate$ percent of the population being replaced by individuals drawn randomly from the best $migration\_rate$ percent of the individuals in other populations.

### Generating the initial population

The Evolution Module is responsible for generating the initial population. In the simplest case, a completely random set of genes could be generated, utilizing only random number generator. However, for genes acting as neural network weights, this is not optimal. Nguyen and Widrow [43] propose a method for initializing the weights of a neural network and show that the method improves the learning speed of a 2-layer neural network when using backpropagation learning.

Even though learning methods using artificial evolution are not as susceptible to the choice of initial weights as the backpropagation based methods, it may still make sense to initialize the weight-genes using the Nguyen-Widrow method. This way the initial values are more suitable for the neural networks and may produce better initial results. More importantly, as Nguyen and Widrow state in their paper, it is reasonable to expect that picking weights so that the hidden units are scattered in the input space will substantially improve learning speed of networks with multiple inputs. This is a valid assumption for evolution based training as well, as the better scattering of the weights will give the evolution process a better chance to start from an area with relatively good solutions. Therefore, the genes that specify the neural network weights are initialized using the Nguyen-Widrow method.

### Selecting Learning Samples

The evolution process is computationally quite heavy. Processing around a hundred learning samples for approximately a hundred individuals in the population would create a rather large computational load. To reduce this load the Evolution Module does not use all the samples in the LSMM for all generations. Instead a prespecified number of random samples are drawn for each generation and the samples are used for processing of all individuals in the generation. To ensure that the samples used during the generation are not from the same speaker one sample is drawn from each speaker. The training database of the current NEABE implementation has 12 samples from 8 speakers so the computational load reduces to under 10% of what it would be if all samples were used for all generations.

To ensure that the evolved result is near optimal for all training samples, they are all used for

each generation during the last 10 generations of the evolution. This way the initial evolution can be done fast, but end results will still get fine tuned toward the training sample set optimum.

### 4.2.3 Fitness Evaluation Module

The Fitness Evaluation Module evaluates how well a given sample was expanded, by comparing the expanded signal received from the Online Subsystem to the wideband signal from the Learning Sample Management Module.

The comparison metric used should measure the difference between two signals as psychoacoustically accurately as possible. In principle, as artificial bandwidth expansion is only speech enhancement, not speech synthesis, the original wideband signal should not even be needed. We could simply measure whether the expansion improves the speech quality and make the evolution select the algorithm parameters that would most improve the speech quality. Unfortunately, no such measure exists. In fact, there is not even any generally agreed upon way to measure the psychoacoustical difference between two speech signals, so a method that approximates the difference accurately enough for expansion quality evaluation purposes must be selected instead.

However, this is not very straightforward either. It is made slightly easier by our use of rank based fitness assignment (see chapter 3.2.2), because the actual values produced by the metric are not important. Therefore the metric can be highly nonlinear (in regard to quality) as long as it introduces an approximate subjective speech quality based ordering.

The spectral envelope of the expanded speech signal is the principal key for a high subjective quality of the speech produced by any bandwidth expansion system [27]. Therefore, it makes sense to use a spectral metric as the quality measure for the expanded signal.

#### Framing the Signals for Spectral Quality Metric Calculation

To compute the spectral quality metric between two signals, a frame based processing scheme is usually used. For FFT based spectral estimation this is necessary. A fundamental assumption when using FFT is that the signals under investigation are wide-sense stationary [36]. Speech is not a WSS process, because the articulators are moving and shaping the vocal tract. However, by selecting a time frame in which the vocal tract movement is neglible[3], the stationarity assumption holds satisfactorily [35].

The time duration of the analysis frame should be long enough to span at least one pitch period, but on the other hand so small that the articulators do not move considerably during the frame. If the frame length is too short, the results will fluctuate very rapidly depending on the exact positioning of the frame [49]. If the frame length is too long, the speech process during it will not be stationary enough and the results of the spectral estimation will not be accurate.

[3] on the order of 15-20 ms for most vowels

When selecting the frame length it is useful to avoid the length used by the expansion process of the Online Subsystem, because using the same length might let some frame synchronized error get through, as the frame pre-FFT windowing reduces the importance of the samples on the frame edges. The risk of this is reduced by the temporal overlap in frames, but as speech is not stationary and, especially during glides, the exact frame time interval can affect the FFT analysis results considerably, using a different framing scheme for the fitness evaluation may still improve the system robustness.

**Selecting the Spectral Quality Metric**

There are many different spectral distance metrics which could be used as a quality measure. For example, Epps and Holmes [11] use the following spectral distortion metric to measure spectral distortion between two envelope shapes in the extension band:

$$D_{HC} = \sqrt{\frac{1}{K}\sum_{k=1}^{K}\int_{0.25\omega_s}^{0.5\omega_s}\left[20\log_{10}\left(G_C\frac{A_k(\omega)}{\hat{A}_k(\omega)}\right)\right]^2 d\omega}, \tag{4.1}$$

where

$$G_C = \frac{1}{0.25\omega_s}\int_{0.25\omega_s}^{0.5\omega_s} 20\log_{10}\left(\frac{\hat{A}_k(\omega)}{A_k(\omega)}\right)d\omega \tag{4.2}$$

and $A_k(\omega)$ is the original and $\bar{A}_k(\omega)$ the predicted envelope of the $k$'th (temporally aligned) frame of wideband speech and $\omega_s$ is the wideband sampling frequency (16 kHz). The compensating gain factor $G_c$ has the effect of removing the mean difference between the two envelopes, and thus $D_{HC}$ measures only the spectral distortion between the envelope shapes.

Another distance measure is log spectral distortion (LSD), which can be defined for the artificial bandwidth expansion as:

$$d_{LSD}^2 = \frac{1}{2\pi}\int_{-\pi}^{\pi}\left(20\log_{10}\frac{\sigma_{rel}}{|A_{mb}(e^{j\omega})|} - 20\log_{10}\frac{\hat{\sigma}_{rel}}{|\hat{A}_{mb}(e^{j\omega})|}\right)^2 d\omega, \tag{4.3}$$

where $A_{mb}(e^{j\omega})$ and $\sigma_{rel}$ denote the modeled frequency spectrum and relative gain of the missing frequency band of the original wideband signal, and $\hat{A}_{mb}(e^{j\omega})$ and $\hat{\sigma}_{rel}$ denote the corresponding parameters of the artificially expanded band. [27]

The LSD can also be expressed in the cepstral domain. For a sequence of speech frames the root mean square average of the LSD is given by:

$$\bar{d}_{LSD} = \frac{\sqrt{2}\,10}{\ln 10}\sqrt{E\Big\{\frac{1}{2}(c_0-\hat{c}_0)^2 + \sum_{i=1}^{\infty}(c_i-\hat{c}_i)^2\Big\}} \tag{4.4}$$

where $E\{\cdot\}$ denotes the expectation operation and cepstral coefficients $c_0, c_1, \ldots$ are calculated from the AR coefficients and the relative gain.[27]

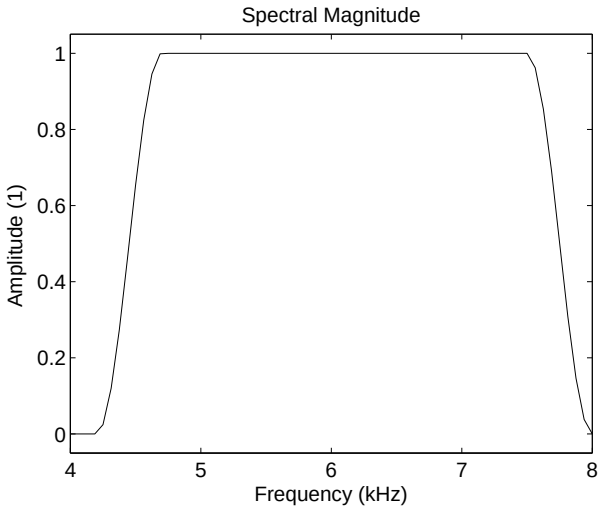



**Figure 4.3:** Raised cosine bandpass filter with 500 Hz transition bands shown in linear amplitude plot.

Two distance metrics have been tested with NEABE. A simple log spectral MSE and a distance metric based on the cepstral MSE of the first $N_c$ coefficients, where $N_c$ is selected such that the coefficients containing information about the spectral harmonics are ignored and the information about the spectral envelope is included.

The MSE metric is also influenced by the harmonic structure. However, because the low-band to highband transform is always the same during the evolution run and thus only the spectral envelope can be altered by the evolution system, the harmonics should not have any considerable effect on the direction of the evolution process.

When expanding telephone band speech, there is an additional problem that the narrowband signal has been bandpass filtered. When the base expansion band is generated directly from the narrowband this causes the expansion band to have some gaps (see section 4.3.3 to understand why). These gaps should not be included into the spectral distance calculation, as their inclusion would cause a genome to benefit from considerably amplifying the bands the gaps are at, which would only add noise to the system. Due to this, the MSE distance is only measured from the spectral bands which are not generated from the attenuated bands of the narrowband signal. Which bands these are depends on the used Lowband to Highband Transfer Function (see section 4.3.3). Similarly, the cepstral calculations filter out the gap bands by using a raised cosine bandpass filter (illustrated in Figure 4.3) in the spectral domain before continuing with the cepstrum calculation.

To make the distance metric robust against pathological signal frames which cause zeroes to appear in the magnitude spectrum[4], the operators use $\log_{10}(1 + x)$ instead of $\log_{10}(x)$ as

[4]The evolution process can sometimes, especially in the beginning of the evolution, generate genomes that cause extreme attenuations on some spectral bands. These attenuations may be so extreme that due to the limited accuracy of the digital computer, zeros occur in the magnitude spectrum of the frame.

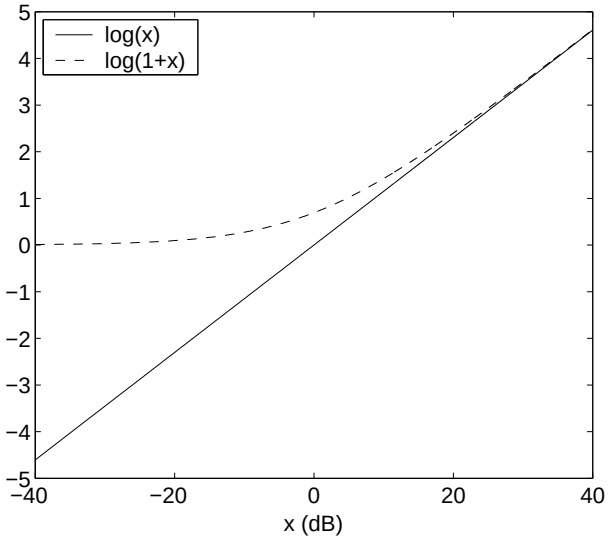



**Figure 4.4:** $\log(1+x)$ de-emphasizes differences in the lower end of the scale, but otherwise behaves similar to $\log(x)$

the logarithm function. This also de-emphasizes the errors near the spectral zeroes (see Figure 4.4), which improves the psychoacoustical expansion results, as the errors will probably get masked by nearby higher magnitude spectral information anyway.

Thus the used frame distance measures are:

$$SMSE_f = \int_{V_1}^{V_2} \Big( \log_{10}(1+|X(\omega)|) - \log_{10}(1+|\hat{X}(\omega)|) \Big)^2 \tag{4.5}$$

$$CMSE_f = \sum_{n=0}^{N_c-1} \Big( \int_{-\pi}^{\pi} \ln(1+|H_{cf}(\omega)X(\omega)|)e^{j\omega n} - \int_{-\pi}^{\pi} \ln(1+|H_{cf}(\omega)\hat{X}(\omega)|)e^{j\omega n} \Big)^2 \tag{4.6}$$

where $SMSE_f$ is the frame log spectral mean square error, $V_1$ is the beginning of the valid expansion band, $V_2$ is the end of the valid expansion band, $X(e^{j\omega})$ is the spectrum of the reference signal frame and $\hat{X}(e^{j\omega})$ is the spectrum of the expanded signal; and $CMSE_f$ is the frame cepstral mean square error, $N_c$ is the number of cepstral coefficients to include in the evaluation, $H_{cf}$ is the spectral domain raised cosine bandpass filter and $X(e^{j\omega})$ and $\hat{X}(e^{j\omega})$ are as before.

### Combining Frame Objective Values to Produce Genome Objective Values

The frame objective values need to be combined to produce a single objective value for each genome. This is done in two parts. First, quality values for frames of each training signal are combined to produce an objective value for the signal and then the signal objective values are

combined by simply calculating an average.

The way the frame objective values are combined is highly important, because it affects the training greatly. In an ideal case we would have training data that would contain equal number of frames for each phoneme, but on the other hand we would like the data to contain frames in a natural order so that the system could learn to exploit the information in the frame order, if recurrent feedback is used.

These principles are somewhat mutually exclusive, so even though the speech samples have been carefully selected to contain all types of phonemes in Finnish, the number of frames per phoneme varies considerably. As an example, /s/ is typically a phoneme of extensive duration, so there are quite many frames containing it, whereas unvoiced plosives, such as /k/, /p/, /t/, are usually quite short and thus few frames contain them.

Of course it can be speculated that the short phonemes do not need to be modeled as carefully as the longer ones, as they are not heard a very long time and thus moderate subjective errors are (more) acceptable in them. While this is true to some extent, a notable but short time error in one frame will reduce the subjective quality considerably and cause a notable artefact in the produced speech.

An ideal objective value combiner would thus emphasize the large errors somehow. One possible scheme of doing this is to apply an extra cost factor for errors that are in some sense much larger than the average error level. This kind of outlier punishing, when implemented correctly, could possibly reduce the amount of artefacts in the produced speech.

Currently, however, the frames are combined to produce a sample objective value by simply averaging them together. The final genome objective value is simply:

$$Obj = \sum_{s \in S} \frac{\sum_{f \in F(s)} \frac{MSE_f(f)}{|F(s)|}}{|S|} \tag{4.7}$$

where $S$ is the set of all samples, $F(s)$ is the set of frames in the sample $s$, $MSE_f(f)$ is either the $SMSE$ or the $CMSE$ of frame $f$ in sample $s$, $|F(s)|$ is the number of frames in the sample $s$ and $|S|$ is the number of samples in the learning sample set.

## 4.3 The Online Subsystem

The Online Subsystem is the heart of the NEABE system. It is responsible for the actual bandwidth expansion. It is the portion of the system that will be integrated to the target system, so ideally it should be computationally efficient, straightforward to implement in any programming language and as robust as possible. The modules in the OSS should be easy to customize and change to achieve the full potential of the NEABE system, the ability to adapt to different operating environments by simple retraining.

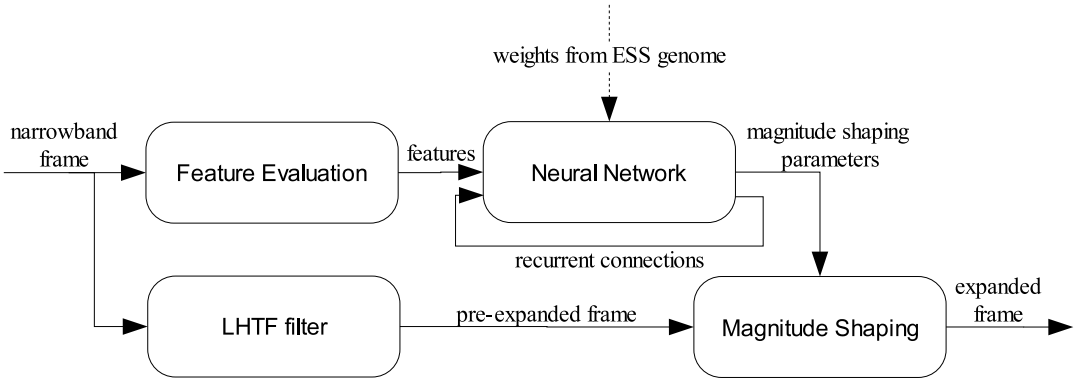



**Figure 4.5:** The structure of the Online Subsystem

The architecture of the online subsystem is presented in Figure 4.5. The online subsystem consists of four main modules: Feature Evaluation Module, which evaluates the features of the frames to be given as inputs to the neural network; Neural Network Module, which is configured by the ESS genome and is responsible for producing magnitude shaping parameters from the feature inputs; Lowband to Highband Transform Filter (LHTF), which adds the basic highband data to the frame by transforming the lowband; and the Magnitude Shaping Module, which modifies the highband produced in LHTF to make it resemble the correct wideband spectrum.

The samples to be expanded are processed frame by frame, the actual framing done by some other system (during evolution, the ESS LSMM, in the end system, by the surrounding telephone system). The OSS operation still depends on the framing method used, as the FFT window used by the system should have the same length as the frame.

A high level description of the OSS expansion process is given in the following steps:

1. Feature Evaluation Module evaluates the expansion features for the frame.

2. The evaluated features are passed to Neural Network Module as parameters. It uses them and (optionally) its own recurrent feedback parameters to evaluate a neural network, which has its weights set according to the genome set by the ESS process. Some outputs of the network are stored in the NN module to be passed as inputs for the next frame.

3. LHTF filter expands the original narrowband frame to produce a basic expanded frame, which has a highband with approximately correct harmonic spacing[5], but incorrect spectral envelope.

4. The expanded frame is attenuated and amplified in the Magnitude Shaping Module, using

[5]The highband harmonics have consistent distance to each other, but their distance to the lowband harmonics may be incorrect.

the magnitude shaping parameters produced by the NN module to control the magnitude modulation.

5. The final expanded frame is output to the surrounding system. If there are more frames to process, the process is continued from step 1.

### 4.3.1 Feature Evaluation Module

Feature Evaluation Module is quite possibly the most important module of the Online Subsystem. The expansion quality is directly related to the quality of the features selected here. If the features do not contain information that is important for deciding how a frame should be expanded, the system will fail. In effect the features are the sensors of the expansion process, without which it cannot "see".

If the sensors are twisting or leaving out the relevant information, we cannot expect the system to be able to act correctly. If the sensors are overflowing the "brain" (the decision making process located in the neural network in this case) with irrelevant information, it will not be able to make good decisions. So the selected features should be accurate, important and give all relevant information, while discarding the information unrelevant for the expansion process.

Additionally there should be as few features as possible to minimize the dimensionality of the solution space the neuroevolution process must search. Each added search space dimension slows the learning process down and adds to the risk of not reaching a solution at all. In addition, the number of training samples needed to prevent overtraining grows with the input space dimension [58], although with the current magnitude shaping approach (explained in section 4.3.4) this is not a big problem, as the magnitude shaping algorithm effectively limits the overtraining risk.

For fully connected neural networks, the number of free parameters grows quite aggressively when inputs are added[6], so the input dimensionality is of even more concern when they are used.

As a consequence of the aforementioned reasons a tradeoff emerges. On one hand, to ensure that the process has all the needed information, we should provide it with all the possible features that could be useful for the process, and on the other, we should minimize the number of features to keep the size of the search space feasible.

#### Feature Subset Selection

There are many methods available for solving the mentioned tradeoff. A good introduction to the feature subset selection is given in [18]. However, for the purposes of the initial NEABE

[6]For fully connected feedforward MLPs, each new input adds $h$ new synaptic connections, where $h$ is the number of hidden neurons in the system, so each input adds $h$ new parameters to be tuned by the evolution process.

implementation, the features were selected by a group of domain experts, mostly basing the selection on what has worked well with the prior ABE algorithm. Some consideration was given to the special qualities of neuroevolution, especially by selecting the simpler features instead of the more refined ones and instead trusting the neuroevolution to find the best way to combine them efficiently.

### Selected Features

In this section the features selected for the NEABE system are briefly explained. In the current NEABE implementation the Feature Evaluation Module is resposible for FFT transforming the signal frames, so both time- and spectral domain features can be used without extra cost and thus the selected features are usually implemented in the domain the implementation is simplest to make.

To produce good quality magnitude shaping curve[7] for the expansion, the expansion algorithm must have either implicit or explicit understanding of different phoneme classes. The older version of the ABE algorithm used explicit classification. It had separate processing paths for sibilants and stop consonants, but classified the other sounds as voiced. As it used processing methods very similar to NEABE and yielded quite good results, it seemed feasible to give NEABE some of the measures it used for classification as features, as the neural network in NEABE must implicitly perform classification like operations to be able to correctly expand the signal.

One of the measures the prior algorithm used to differentiate between voiced and unvoiced sound frames was the zero crossing rate. The zero crossing rate is defined as the number of times the signal crosses the zero-level within the frame, often normalized by the number of samples in the frame.

Unfortunately, zero crossing rate does not measure the magnitude of the direction change in any way. Even though it is a good measure for explicit classification, a more fine grained and robust measure would be beneficial for the neural network in NEABE, which has a control task (setting the Magnitude Shaping Module spline control parameters) in addition to its implicit classification task. Luckily, such a measure exists and has been used for bandwidth extension before, in [27].

**Gradient Index** Introduced by Paulus in [46] for the voiced/unvoiced classification of speech segments, the gradient index is based on the sum of magnitudes of the gradient of the speech

[7] Magnitude shaping curve is responsible for amplifying and attenuating the highband to produce natural sounding expansion. It is described in section 4.3.4.

signal at each change of direction and can be defined as:

$$x_{gi} = \frac{1}{10} \cdot \frac{\sum_{\kappa=1}^{N_\kappa - 1} \Psi(\kappa)|s_{nb}(\kappa) - s_{nb}(\kappa - 1)|}{\sqrt{\sum_{\kappa=0}^{N_\kappa - 1} (s_{nb}(\kappa))^2}}, \tag{4.8}$$

where $\Psi(\kappa) = 1/2|\psi(\kappa) - \psi(\kappa)|$, in which $\psi(\kappa)$ denotes the sign of the gradient $s_{nb}(\kappa) - s_{nb}(\kappa - 1)$. The gradient index has low values during voiced sounds and high values during unvoiced sounds. [27]

Another feature for voiced/unvoiced classification, which was used for sibilant detection in the old algorithm, was taken in to provide additional accuracy to the expansion, because it had worked rather well in the old algorithm.

**Differential Energy Ratio** Differential energy ratio is defined as the ratio of the energy of the second derivative of the signal and the energy of the signal. The second derivative is approximated with a FIR filter with impulse response $h(n) = \delta(n) - 2\delta(n-1) + \delta(n-2)$.

$$x_{der} = \frac{\sum_{\kappa=2}^{N_\kappa - 1} (s_{nb}(\kappa) - 2s_{nb}(\kappa - 1) + s_{nb}(\kappa - 2))^2}{\sum_{\kappa=0}^{N_\kappa - 1} (s_{nb}(\kappa))^2} \tag{4.9}$$

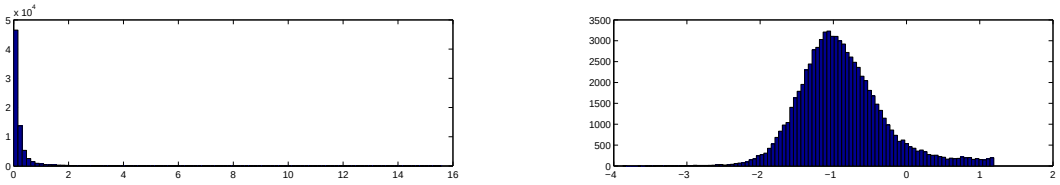
To give the neural network a chance to separate stop-consonant frames from other unvoiced sounds (such as sibilants), a feature capable of detecting temporal changes in relative signal energies was needed. The simplest of these is the ratio of current and last frame energies. However, testing the feature with a speech database revealed that logarithm of the feature was a more suitable measure, giving a nicer spread of the values, as can be seen from the feature histograms in Figure 4.6.

**Ratio of Energies**

$$x_{roe} = \log \frac{E_n}{E_{n-1}} \tag{4.10}$$

where $\log$ can be any logarithm, $\log_{10}$ was used in the current implementation.

Because the neural network in NEABE has a control task, it needs to know something about the power at the different spectral bands of the original narrowband frame to be able to deduce



(a) $\frac{E_n}{E_{n-1}}$ (b) $\log_{10}(\frac{E_n}{E_{n-1}})$

**Figure 4.6:** Feature histograms for ratio of energies before and after logarithm.

the needed amplification levels and thus the control values it should output to the Magnitude Shaping Module. Towards this end, the average magnitude of some (four in the current implementation) spectral subbands are calculated.

**Average Subband Magnitude** The average subband magnitudes are transformed into logarithmic (decibel) domain, both to make it easier for the neuroevolution to extract the relevant information and to make the features more human readable.

$$x_{asm} = 20 * \log_{10} \frac{\sum_{k=k_0}^{k_1} |S_{nb}(k)|}{k_1 - k_0} \tag{4.11}$$

where $k_0$ is the FFT index corresponding to the starting and $k_1$ the FFT index corresponding to the ending frequency of the processed spectral band and $S_{nb}(k)$ is the FFT coefficient with index $k$.

### 4.3.2 Neural Network Module

Neural Network Module (NNM) is responsible for transforming the input features into the parameters used by the Magnitude Shaping Module to produce the magnitude shaping curve for the expansion band. Its weights are the main control mechanism the Evolution Subsystem has over the Online Subsystem. In the current implementation, they are the only control mechanism as well, but basically any parameters of any modules in the Online Subsystem could be controlled by the evolution, if so desired.

Conceptually, the task of the Neural Network Module is simple. It is used as a function approximator to estimate the mapping from features to Magnitude Shaping Module parameters. All learning is done by the Evolution Subsystem, so no learning algorithm is needed in the NNM. A simple MLP network suffices for the task.

However, a question remains whether the MLP should be feedforward or recurrent. Simplicity and the smaller number of weights needed speak for feedforwardness, whereas the potential to learn to utilize long-term feature information speaks for the recurrent version. A compromise was made by choosing a feedforward neural network with optional feedback outputs. The

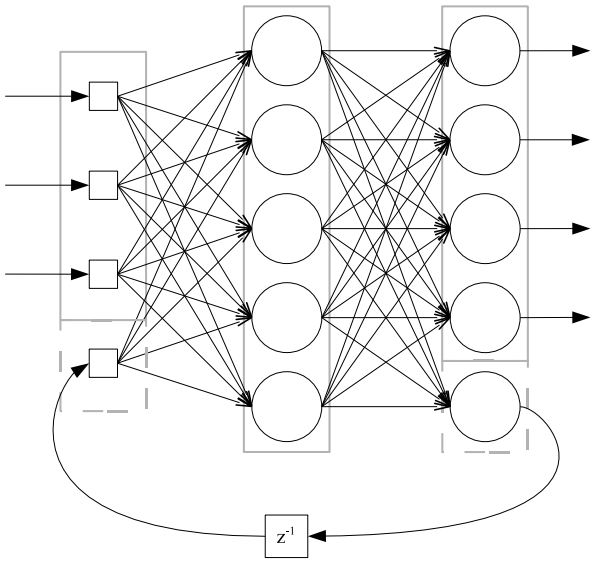



**Figure 4.7:** MLP with one feedback channel.

idea, illustrated in Figure 4.7, is that some extra outputs, in addition to the Magnitude Shaping Module parameter outputs, are added[8]. These outputs are fed back as inputs for the next frame. When the feedback outputs are used, it is up to the evolution process to decide what to do with them. The idea is that it can teach the neural network to use those outputs as feedback channels to send whatever information, which will improve the expansion fitness, as inputs to the next frame.

Of course, this is not a complete recurrent MLP. There is only a limited number of signaling channels available and thus the different neurons are forced to compete for the channels, instead of each hidden neuron having their own feedback channels directly to the other hidden neurons. However, this reduces the number of weights needed, which simplifies the learning task at hand, something which is very important when using a basic genetic algorithm for learning instead of more specialized neuroevolution methods. In addition, in theory the MLP with feedback could evolve into a structure which is equivalent of the complete recurrent network, if that was the optimal structure for the task and enough communication channels were given. Also, implemented this way, the amount of recurrent feedback can be controlled easily. In a sense, recurrence is moved from neuron level to network level.

[8]This partially recurrent neural network scheme is known as Jordan network in literature. However, in typical Jordan network, ordinary network output is fed back, whereas here additional communication outputs exist and the artificial evolution process is free to decide what to communicate.

**Input Normalization**

The inputs for the neural network are normalized during the teaching process by scaling them using the estimated means and standard deviances for the features. For N available data of the *k*th feature we have

$$\begin{aligned} \bar{x}_k &= \frac{1}{N}\sum_{i=1}^{N} x_{ik}, \qquad k = 1, 2, \ldots, l \\ \sigma_k^2 &= \frac{1}{N-1}\sum_{i=1}^{N}(x_{ik} - \bar{x}_k)^2 \\ \hat{x}_{ik} &= \frac{x_{ik} - \bar{x}_k}{\sigma_k} \end{aligned} \tag{4.12}$$

All the resulting normalized features will have zero mean and unit variance. The Online Subsystem deployed into the final operating environment will use the estimates found during training.

### 4.3.3 Lowband to Highband Transform Filter

The Lowband to Highband Transform Filter transforms the narrowband input frame into a wideband frame, creating a basic highband which will be shaped by the Magnitude Shaping Module to form the final expansion band.

Currently, the wideband frame is generated using spectral folding by controlled aliasing to produce the highband. The narrowband signal is upsampled by two, by inserting zeroes between the samples in the narrowband frame. This is the equivalent of mirroring the lowband into the highband in the frequency domain. The approach is illustrated in Figure 4.8.

Two other ways of transforming the lowband into a highband were considered. One was a simple translation in the spectral domain, creating an exact copy of the lowband as a highband. The process is shown in Figure 4.8. Using this method would have created a spectral domain discontinuity between the end of the lowband and the beginning of the highband, which probably would have caused artefacts in the expanded speech. An attempt could have been made to solve the discontinuity by letting the evolution process search a value also for the first Magnitude Shaping Module control point, but the result would not have been as good as with the aliasing method which, due to mirroring, quarantees continuity over the seam.

On the positive side, the spectral gap caused by the bandpass characteristic of telephone speech would have been smaller, because the narrowband signal reaches its unattenuated value around 300 Hz and thus the gap on the beginning of the highband would have been only 300 Hz. The gap on the high end of the narrowband starts from around 3400 Hz, thus the spectral folding produces a highband gap of about 600 Hz.

The other considered LHTF method was using nonlinear distortion [27] $s_{wb}(n) = g(s_{nb})$ of the upsampled and lowpass filtered signal to generate the highband information and combining

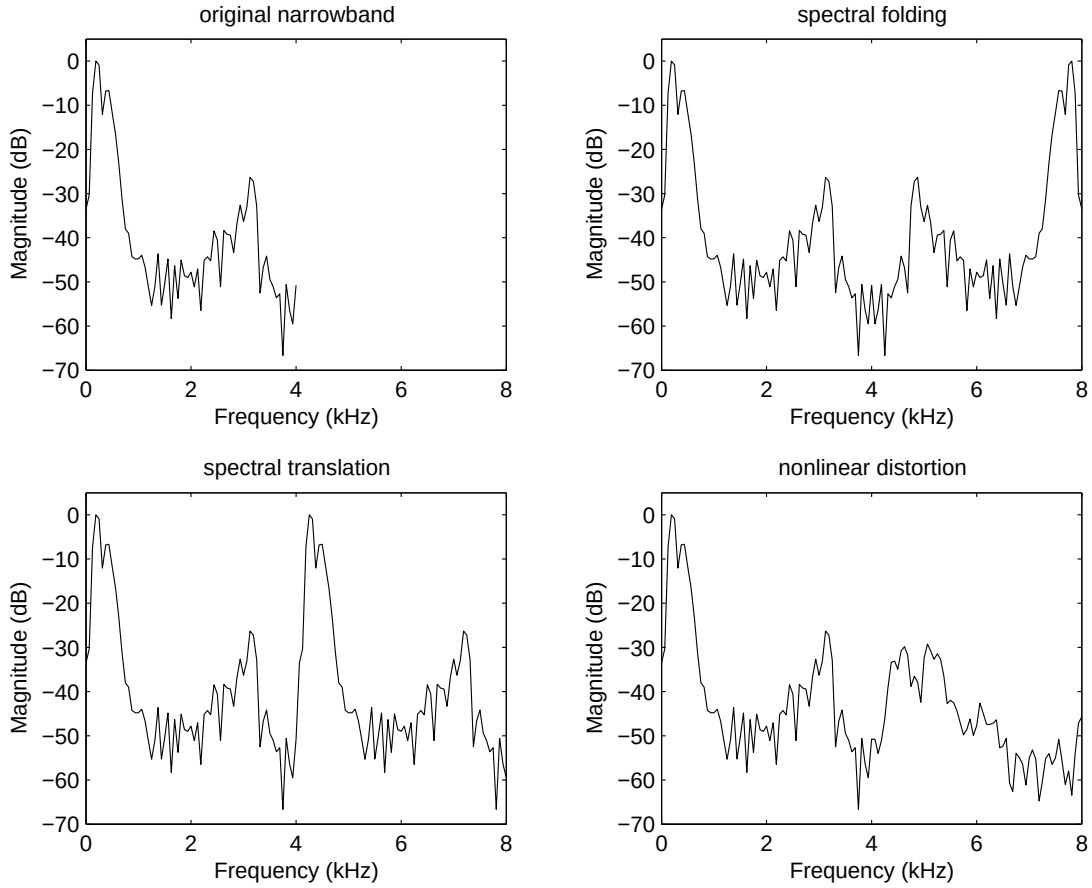



**Figure 4.8:** Example results of different LHTF methods. Upper left panel shows the original narrowband magnitude spectrum. The results of spectral folding are shown in the upper right and the results of spectral translation in the lower left panel. The quadratic distortion results, shown in lower right panel, have been power corrected by attenuating the highband so that the spectrum is continuous over the low- to highband transition.

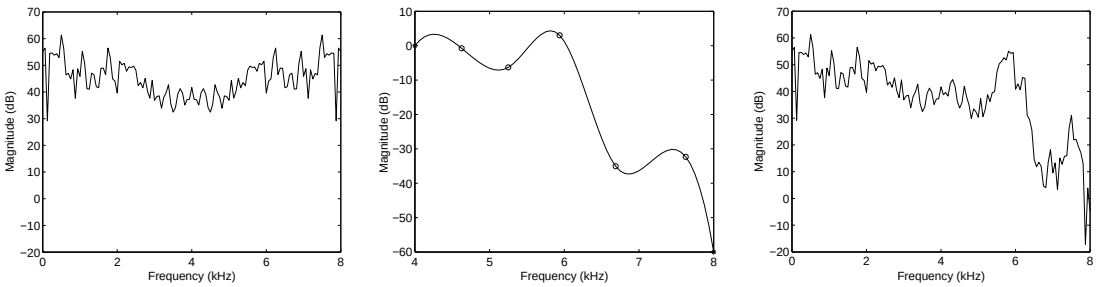
it with the original narrowband lowband in the spectral domain. The considered nonlinearity was a simple quadratic, using the function $g(s(n)) = (s(n))^2$, which, according to [27], produces only harmonic distortions and therefore ensures that the tonal components of the generated wideband signal match the harmonic structure of the bandlimited signal during voiced sounds. LHTF by quadratic distortion is illustrated in Figure 4.8.

Using nonlinear distortion would have removed the gap altogether and would have ensured the harmonic structure match, but at the cost of using artificially generated excitation information in the highband. In addition, the effects of the nonlinearity can be difficult to predict and generally require sophisticated post-processing to avoid [27]. Although this post-processing could possibly have been left for the neuroevolution controlled magnitude shaping to handle, it would have complicated the task of actual bandwidth expansion by mixing it with the nonlinearity compensation. The other alternative, separate nonlinearity compensation module along the lines of the one described in [27], would have further increased the already quite formidable computational requirements.

In addition, the prior ABE algorithm used folding, thus using it simplified the implementation, made comparisons with the old algorithm easier and reduced the need of module testing for the LHTF module.

### 4.3.4 Magnitude Shaping Module

Magnitude Shaping Module is responsible for attenuating and amplifying the highband produced by the LHTF to produce the final, natural sounding expansion band for the speech frame. It uses the information extracted from the features by the neural network to create a modulation curve which is used to adjust the spectral envelope of the LHTF generated highband to better resemble the original wideband signal. The concept is illustrated in Figure 4.9.




(a) Expanded frame after LHTF.

(b) Magnitude shaping curve for the frame. Control points marked with circles, fixed control points with stars.

(c) Expanded frame after magnitude shaping.

**Figure 4.9:** Magnitude shaping curve is used to modulate the generated wideband to more closely resemble the original wideband.

The shaping curve used in the Magnitude Shaping Module can be selected independent of the other modules, as long as the number of NNM outputs are adjusted. The neuroevolution process will strive to optimize the module input parameters for the selected curve. However, care must be taken not to introduce a curve that is too flexible, otherwise overfitting may occur. Even though the overtraining could be prevented by using a validation set to early stop the learning at the optimal point [58], it would not prevent the model from distorting the harmonic structure of the signal. Thus a magnitude shaping curve coarse enough is needed to guarantee that the harmonic structure of the speech signal remains continuous while the spectral envelope is adapted. In addition, the curve should have as few free parameters as possible to make the learning more efficient, while on the other hand it needs enough parameters to be able to adapt well to the high frequency range of the original wideband signal during training.

Preferably the magnitude shaping curve should also be smooth, as abrupt, discontinuous changes in the spectral envelope are quite rare and could cause the curve to have a long impulse response. This, in turn, would lead to quality degradation, caused by impulse response clipping induced by the short frame length. Smooth curve is also intuitively more pleasing, as the role of the Magnitude Shaping Module can be seen as the filter of the source-filter model and the role of the LHTF as the source of the model. Therefore, the magnitude shaping curve should be continuous. In practice, for this to work, the LHTF must not introduce any spectral envelope discontinuities, which, depending on the selected LHTF, may or may not be true. With spectral folding the assumption holds.

In addition to other requirements for the magnitude shaping curve, the computational efficiency needs to be considered as well, as real-time curve generation is needed.

Cubic splines were the curve-type chosen for the initial implementation. They are smooth (C2 continuity), local[9], interpolative curves with feasible computational requirements [21]. Interpolativity makes it easy to achieve a continuity in the spectral domain between the low- and high bands, by simply setting the first control point of the curve to move the beginning of the highband spectrum to coincide with the lowband spectrum endpoint. In the case of spectral folding this is even easier, as a fixed 0 dB amplification suffices. The continuity is desirable to avoid sudden changes over the lowband/highband seam.

The magnitude shaping control for splines is done using fixed frequency control points. In comparison to letting the evolution process set both the frequency and the magnitude of the control points, this reduces the number of free parameters in the model. However, it adds the requirement of selecting the control point frequency locations. The selection is an important one, as it affects the flexibility and level of control the curve has on different subbands of the signal. The number of the control points should be such that adequate frequency resolution for efficient control of the spectral envelope is reached, but the module does not alter the harmonic structure of the signal or adapt to possible noise in the teaching samples.

[9] a change in some part of the curve changes only a finite number of control points surrounding it

In addition to the number of control points, their relative locations are important. Instead of setting the control points to a fixed distance from each other, it makes sense to set them to fixed *frequency warped*[24] locations to give the control system a frequency resolution similar to that of human auditory system.

Also, when using telephone speech samples, special attention must be paid to the spectral gaps caused by the bandpass filtering of the original signal. The Fitness Evaluation Module cannot accurately evaluate fitness on the spectral gaps so it is usually best to avoid setting evolution controlled control points into the gap areas, as their values will be set according to the influence they have on spectral bands that are included in the fitness evaluation. This can lead to excessive amplifications or attenuations in the control points to get best possible fit in the evaluated subband.

In the current implementation, the first control point is fixed to 0 dB in the beginning of the expanded band and the last control point to -60 dB in the end of the expanded band. Both are in spectral gaps and, during preliminary tests, it was found that using fixed amplification values for them improved the expansion results of the telephone band speech considerably. In the case of the first control point, it is also reasonable, because it ensures continuity over the seam. In the case of the last control point, using fixed value effectively leads to lowpass filtering the signal, which reduces the expansion quality in the case of fricatives. However, the speech samples used in our implementation were already lowpass filtered with a cutoff of around 7 kHz, so the expansion quality would not have been too good in the last kilohertz band anyway.

## 4.4 Current Implementation of the NEABE System

This section overviews the details of the our current NEABE implementation. Figure 4.10 shows the block diagram of the implementation. Currently, there are two different working parameter sets, which produce a bit different results.

One parameter set uses a simple feedforward network of ten hidden neurons and recombines all individuals. The parameter set was tested without recombination as well, but no remarkable difference in expansion quality was found.

The other parameter set uses two feedback channels and a neural network of five hidden neurons. Recombination is not used, as the competing conventions problem would cause it to hinder the development of recurrent connections. The network performs better than an equivalent network without the feedback communication channels, which indicates that the channels are being used for transmitting useful information.

The performance of the two parameter sets in terms of objective function minimization is similar, but subjectively the results are somewhat different. The direct set tends to overexpand sibilants (leading to metallic, sharp /s/, for example), whereas the feedback communication set is somewhat more conservative, sometimes even underexpanding (transforming /s/ towards /ʃ/).

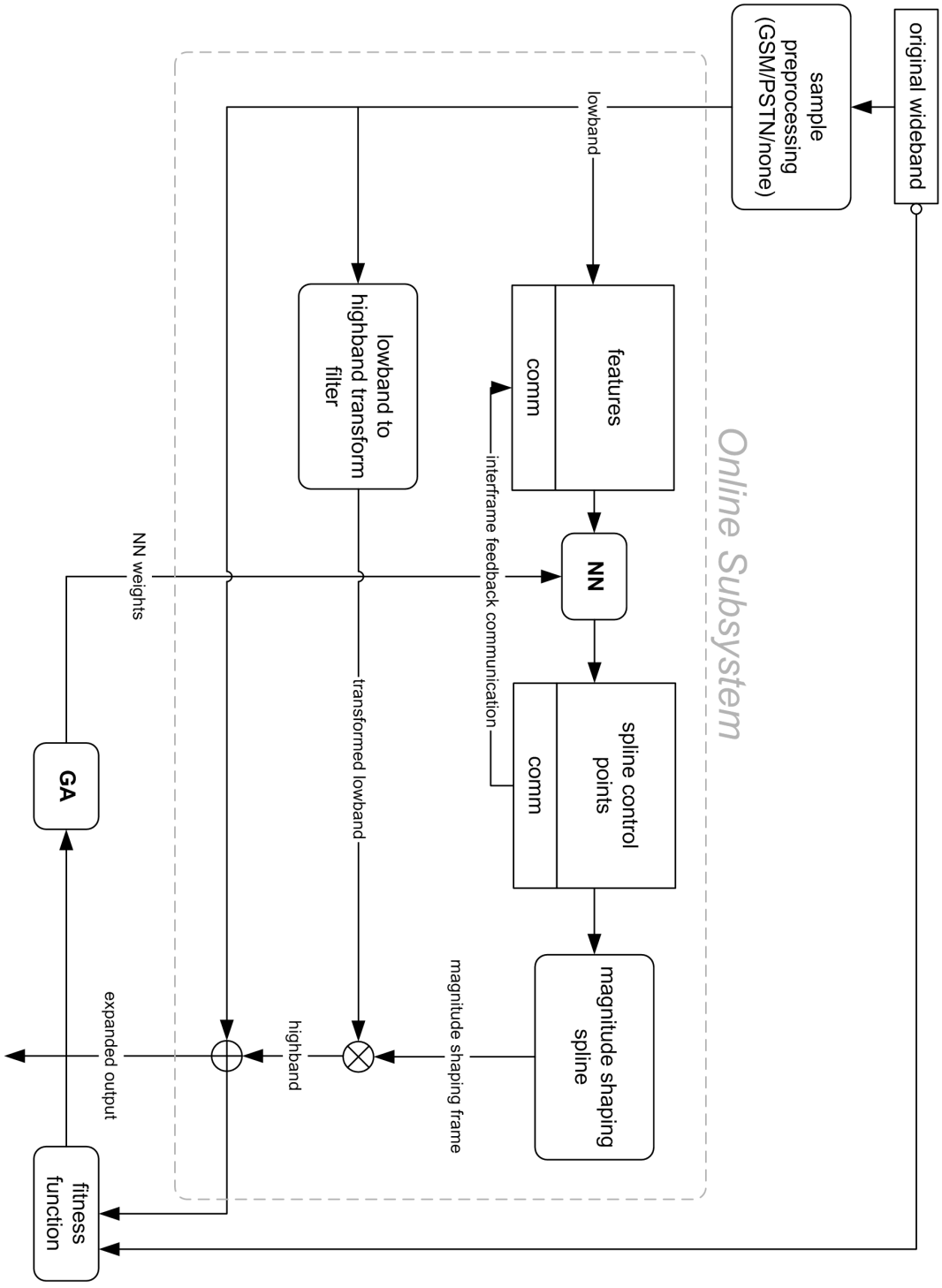



**Figure 4.10:** NEABE implementation block diagram

# Chapter 5

# Results

To evaluate the performance of the NEABE algorithm and improve our understanding of the way it works, extensive analysis was performed. This analysis was divided into three categories, each of which has its own purpose: the learning process analysis, the objective quality analysis and the subjective quality analysis.

While the subjective quality of the expanded speech is clearly the most important measure of performance for an artificial bandwidth expander, it does not, unfortunately, provide useful information for developing the algorithm further. To improve the algorithm, knowledge of its internal operation is needed. In NEABE, the operation of the algorithm can be divided roughly into two parts: *learning* and *expanding*. Here, learning means the task of deriving workable parameters for the expansion system using artificial evolution. Expanding refers to the actual expansion process, which uses the neural network controlled magnitude shaping curve to adjust the higher spectral components created by the lowband to highband transform function.

It is critical that the learning part performs adequately, because without proper learning the system fails to maximize its expansion potential. If the algorithm population converges into a point and learning stagnates early, the results will be poor. If the learning process fails to utilize its current good individuals to create even better individuals and that way move towards the global minimum of the objective function, the results will be poor. In section 5.1, an analysis of a typical NEABE learning process is provided.

Finding good individuals does not guarantee good expansion quality. Minimizing the objective function over a dataset only guarantees good performance on average. As the dataset contains speech signals with varying amounts of different phonemes, the expansion quality may be inadequate for some specific phonemes or phoneme combinations, while still performing well with other combinations. Therefore, a close look on what the evolved expander actually does is in order. Section 5.2 provides this kind of objective quality analysis, exploring the way the expansion process behaves through analyzing the performance of the expander on an example sound.

As mentioned, the subjective expansion quality is the most important performance measure for the artificial bandwidth expander. Section 5.3 describes the results and analysis of a two-part listening test which was used to evaluate the subjective expansion quality.

All the results presented in this chapter were derived from the same expander, which was trained with a 196 phrase learning dataset (4 male and 4 female speakers, 24 phrases each) using the feedback parameter set (5 hidden neurons, 2 feedback channels, no recombination during learning). The expander performance was thereafter tested using a test set of 16 samples (7 male speakers with 8 samples total, 5 female speakers with 8 samples total), from which a specific female speaker sample was mostly used for objective testing and all samples were used for the subjective listening test. Narrowband sources for the expansion in both learning and testing sets were created by simply lowpass filtering (with 4 kHz cutoff) and downsampling the original wideband samples.

## 5.1 Learning Process Analysis

To train the expander, a population of 80 individuals was run for 250 generations. The objective value (MSE of first 35 cepstral components averaged over all frames of all teaching samples, as described in section 4.2.3) for the best individual of each generation is shown in Figure 5.1.

Learning process makes major discoveries around generation 20 and 40–50, after which the learning converges into a slower descent towards the minimum. It should be noted that the results continue to improve throughout the evolution. This indicates that it might be possible to improve the expander even more by continuing the evolution further, although signs of stagnation can be seen from generation 230 onwards. The flat part in the end of the evolution is due to all samples being used for evaluating the individual performance during the last 10 generations, which removes the noise caused by the random sampling of fitness evaluation samples (see section 4.2.2 for details). The random sampling also causes the local increasing of best objective values. While best individuals are passed on by the elitist evolution strategy, they may perform worse with other randomly sampled sets of phrases.

The constant improvement of the objective value throughout evolution shows that the evolution search is successfully learning to better utilize the expansion process it is controlling. It also indicates that the population diversity is staying high enough and new discoveries are constantly made. Used selective pressure also seems to be satisfactory, leading to quick capitalization of promising individuals found, as shown by the rapid descent during generations 40–50.

### 5.1.1 Visualizing the Population Development

To get a better idea of the way the population is evolving during the search, a visualization of all individuals of all generations was needed. Unfortunately, direct visualization of the 96

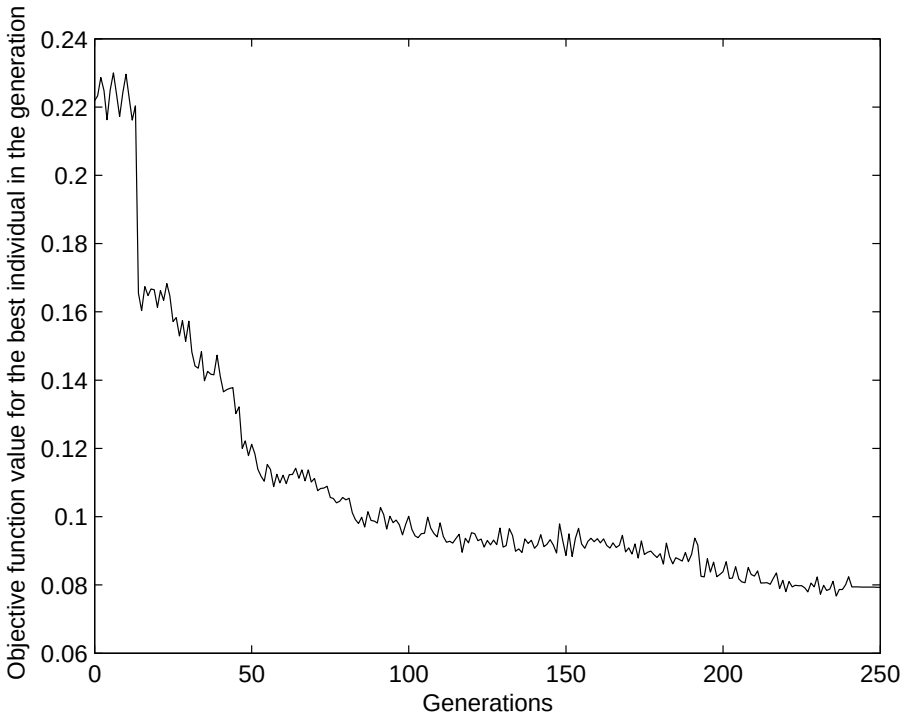



**Figure 5.1:** The objective function value for the best individual of each generation, from generation 0 to generation 250. Learning process makes major discoveries around generation 20 and 40–50, after which the learning converges into a slower descent towards the minimum. The objective function value of the final optimal point is 0.0768.

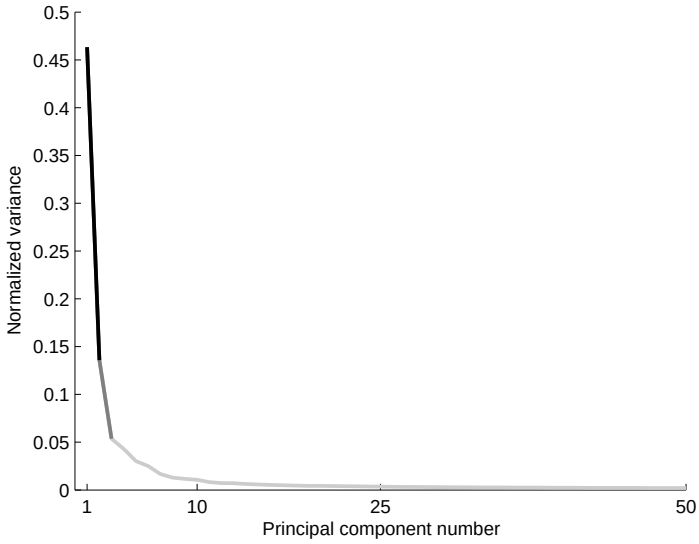



**Figure 5.2:** Relative variances of the 50 largest principal components. The variances have been normalized by the total sum of variances. To give an idea of the amount of information displayed compared to the total amount of information, the dark gray line shows largest three principal components and the black line largest two principal components. Three components give 0.054 (5.4% of total variance) of additional information compared to using only two components.

weight neural network space is impossible. To provide a visualization that contains as much of the data variance as possible, Principal Component Analysis (PCA)[1] was used.

To get an idea how much of the variance the PCA is able to display, the variances of the fifty largest principal components were plotted. These are shown in Figure 5.2. As can be seen from the picture, around ten first components seem to contain significant information. There is a significant increase in information displayed between two (59.9% of total variance) and three (65.3%) principal components, so three dimensional visualization was selected.

The three dimensional PCA visualization of the evolution of the GA population through time is shown in Figure 5.3.

Population moves through the space, exploring promising regions, changing direction abruptly when better solutions emerge. The selective pressure causes the whole population to move in the solution space to improve the values in promising areas. As recombination is not used, the only force directing the search towards the promising areas arises from selection, which causes the population to bias towards promising individuals. The promising individuals receive more offspring and thus the population steers into their direction.

Still, even in the end the populations remain spread out, exploring yet unfound solutions. The diversity remains high partly because of mutations, and partly because of migration which

[1]PCA is a classical statistical method, which has been widely used in data analysis and compression. An introduction to this linear transform can be found in various books, for example in [17].

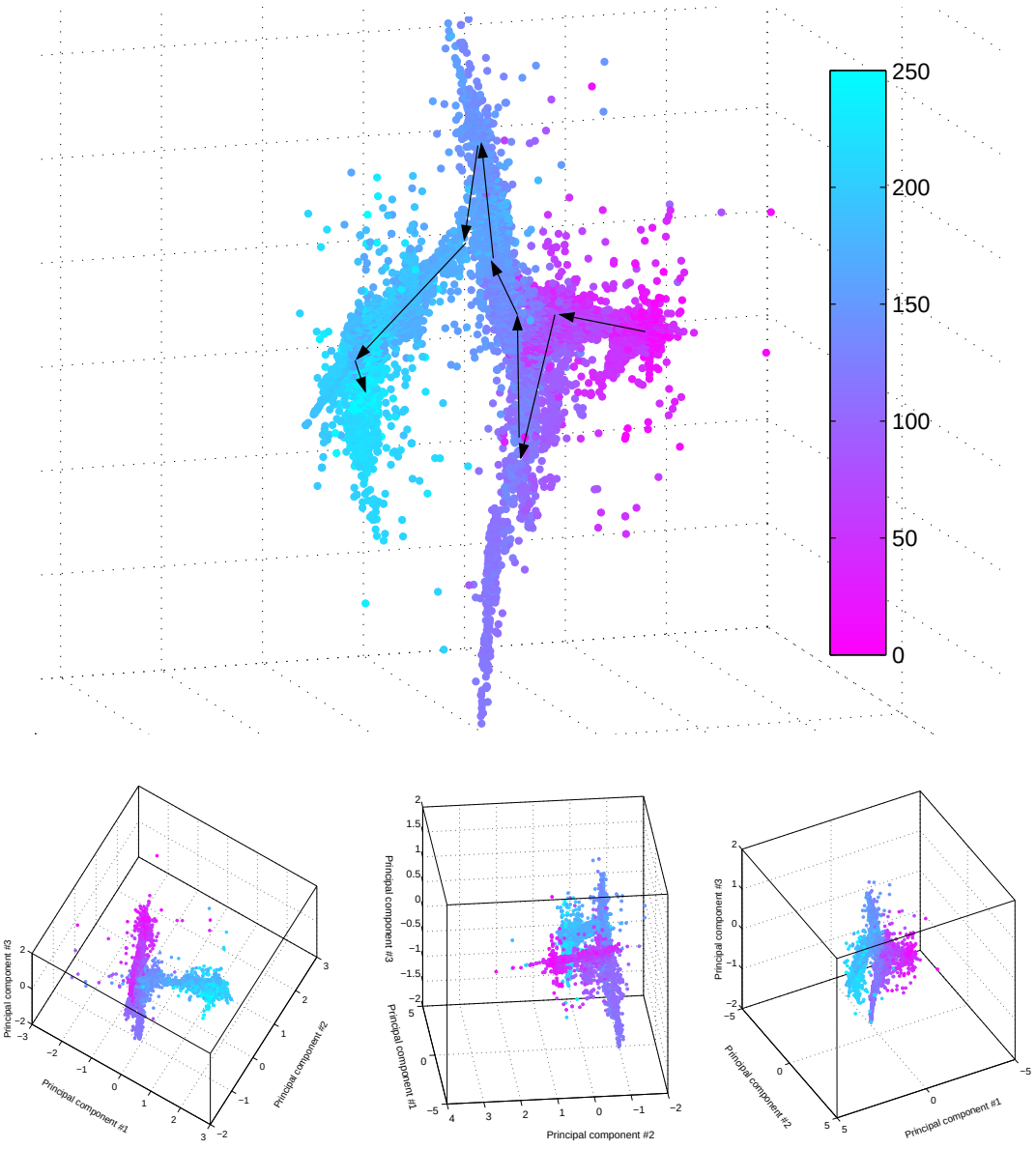



**Figure 5.3:** Visualization of the evolution of the GA population through time. Different colors represent different populations, from 0th generation to 250th generation. The visualization has been calculated using principal component analysis (PCA). Three principal components with largest variances are shown. The components have been normalized by the standard deviation of the first principal component. The arrows show the approximate path of the population movement through time.

keeps the subpopulations heterogeneous enough by supplying them worthy contestants once in every twenty generations. Because the migration is elitist, the individuals moving in have high fitness and therefore prevent the former best individual of the population from taking over the whole subpopulation.

### 5.1.2 Visualizing the Development of the Best Individual

Visualizing the best individuals is informative for two reasons. First, it can be used to ascertain that the population moves towards good individuals instead of drifting randomly around. Second, robust solutions should have good solutions around them as well, so finding a cluster of former best solutions around the final solution might indicate solution robustness. In addition, the best individual plot can be used to determine the effect of randomly selecting a subset of learning samples for fitness evaluation instead of using the whole learning sample set. Any difference should be visible as displacement of the best individual during last ten generations, when all learning material is used to direct the evolution.

A visualization of the development of the best individual through time is shown in Figure 5.4. Two large clusters are clearly visible, one near the best individual of the 125th generation and one above the best individual of 250th generation. In addition, some smaller clusters exist in both of the “spikes” in the lower picture. The best individuals of ten last generations are somewhat lower on both the second and the third principal component than the best individuals of the 20–40 generations before that. The difference is especially clear on the third principal component.

The population movement seems to follow the movement of the best individual, as expected. For each population “spike”, there is a corresponding cluster of best individuals that attracted it.

The first major cluster of individuals starts around generation 50 or so, just when the search has converged into a steady descent, as can be seen in Figure 5.1. The second cluster starts around generation 200, but there is no clear shift in objective value between these two clusters.

The displacement of the best individual of last ten generations is believed to be due to sampling effects. Randomly drawing part of the learning samples gives only noisy estimate of the objective function for the whole set. However, the last best individuals are still relatively close to the second large cluster, which indicates that despite the noise, the error in learning achieved with random sampling is not large and therefore using this method to cut down the computational cost seems valid.

### 5.1.3 Visualizing the Effects of Migration

As mentioned before, the diversity of search is believed to remain high partly because of migration, which keeps the subpopulations heterogeneous by exchanging best individuals, and on

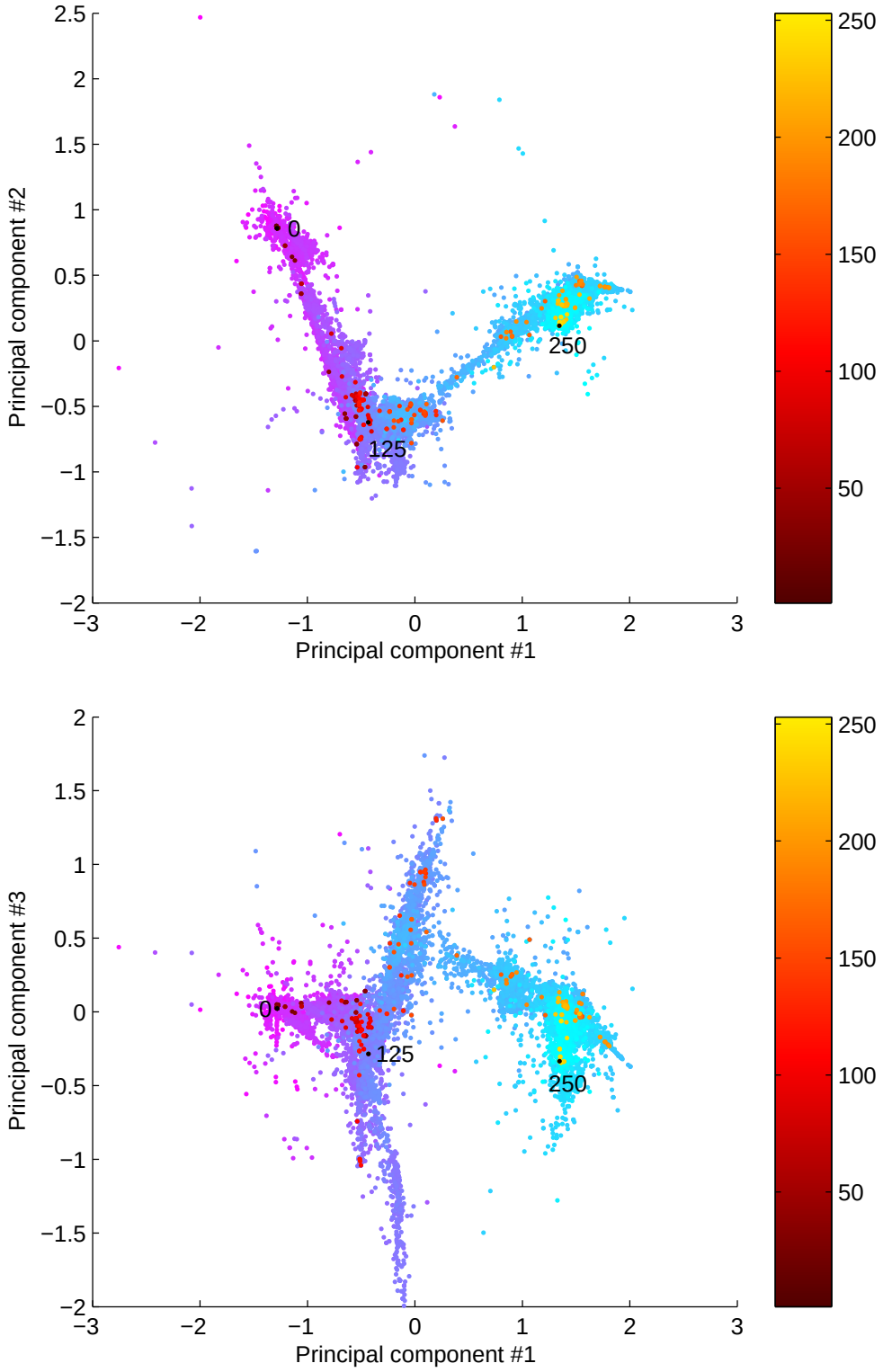



**Figure 5.4:** Visualization of the evolution of the best individual through time. The different populations are colored as in Figure 5.3. The colors shown in color bar represent the best individual of the generation, from 0th generation to 250th generation. The first picture shows the locations of the best individuals on a plane comprised of two largest principal components. The second picture shows the locations on a plane comprised of the largest and the third largest principal component. Component normalization and other details are as in Figure 5.3.

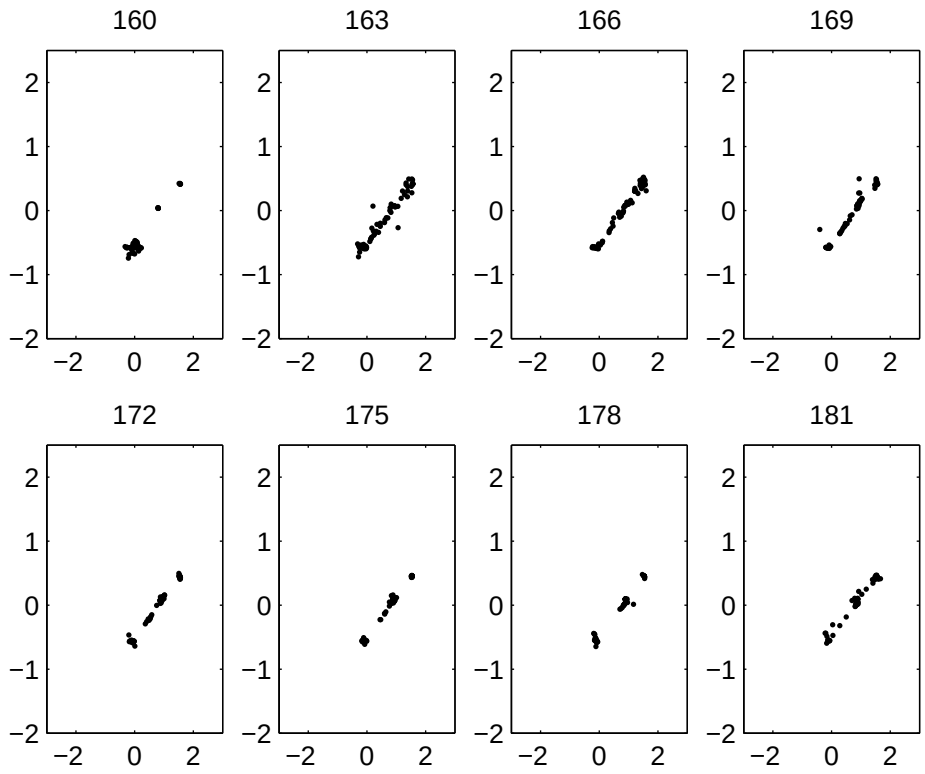



**Figure 5.5:** A visualization of convergence-divergence cycle which is believed to be caused by migration. The picture x-axis is the first principal component and the y-axis is the second principal component. The number above each frame denotes the number of the generation in the picture. As migration happens once every twenty generations, generation 160 is just before migration and generation 181 just after another migration. Converged subpopulations diverge after migration, just to converge again to new positions and wait until new migration cycle diverges them again.

the other hand by the subpopulations, which contentrate on different parts of the search space.

Support for this assumption was found by animating the population movements. The subpopulations were found to converge until migration, and spread out again right after it just to converge again later. This convergence-divergence cycle is visualized in Figure 5.5.

In the 160th generation, the subpopulations have converged. Right after this generation, the migration is once again performed. Migration brings individuals from different subpopulations into a subpopulation, causing these to compete against each other. This competition causes temporary divergence, until a certain individual and its offspring win out and the subpopulations converge again. Next migration cycle again causes the population to spread, as can be seen in the picture of the 181th generation. High migration rate (20%) causes migration to be an important factor in NEABE's search.

## 5.2 Expansion Behavior Analysis

Expansion behavior analysis in this section aims to give a high level understanding of the expansion quality and explore some specific details of interest in the expansion process, such as how specific phonemes are expanded and what is the controller's frame-to-frame behavior like. The aim is more to give an idea of how the system works than to give a detailed analysis of the system quality measured by some objective quality measures. The aim of an artificial bandwidth expansion system for speech is to improve the subjective quality, not objective, and therefore performing objective quality measurements is not very useful, except for giving an easy benchmark number to compare the performance of algorithms with.

### 5.2.1 High Level Spectral View on Expansion Results

A good way to get a high level understanding of the expansion results and their quality is to compare spectrograms of the expanded speech with spectrograms of the original wideband speech. This is done in Figure 5.6, where the spectrogram of the narrowband source speech is also shown.

As narrowband signals used here were created by lowpass filtering the wideband signals, there are negligible differences in 0–4 kHz range between the narrowband and the wideband signal. In real telephone speech, this would not be the case. It should also be noted that expansion currently does not evaluate frequencies above 7.5 kHz, because speech in the wideband speech database has been lowpass filtered with a cutoff around 7.5 kHz and therefore there exists no valid target to evaluate the expansion fitness against. In addition, when using telephone speech the spectral gap caused by the lowband to highband transform would prevent expansion above 7.7 kHz anyway.

Comparing wideband and artificially expanded signals in the band from 4 kHz to 7.5 kHz, certain issues can be noted. Mirroring effects caused by the lowband to highband transform function (LHTF) are clearly visible around the 4 kHz range. Higher up in frequency, they can be seen, but are strongly modulated down in frequencies where the original wideband does not have similar effects. An interesting artefact of this kind is the highest spectral peak trajectory, which is visible in 6–7 kHz spectral band. Original wideband does not have a similar effect. However, while this is an artefact in the objective sense, it is not audible in practice and therefore does not cause severe quality degradation.

In general, the expansion seems to roughly match the spectral power levels of the original wideband. Listening the speech sample, no clear artefacts can be distinguished, although the /s/ is stronger in the original wideband speech. This is is also visible in the spectrogram, where just after 0.6s there is a clear highband power difference.

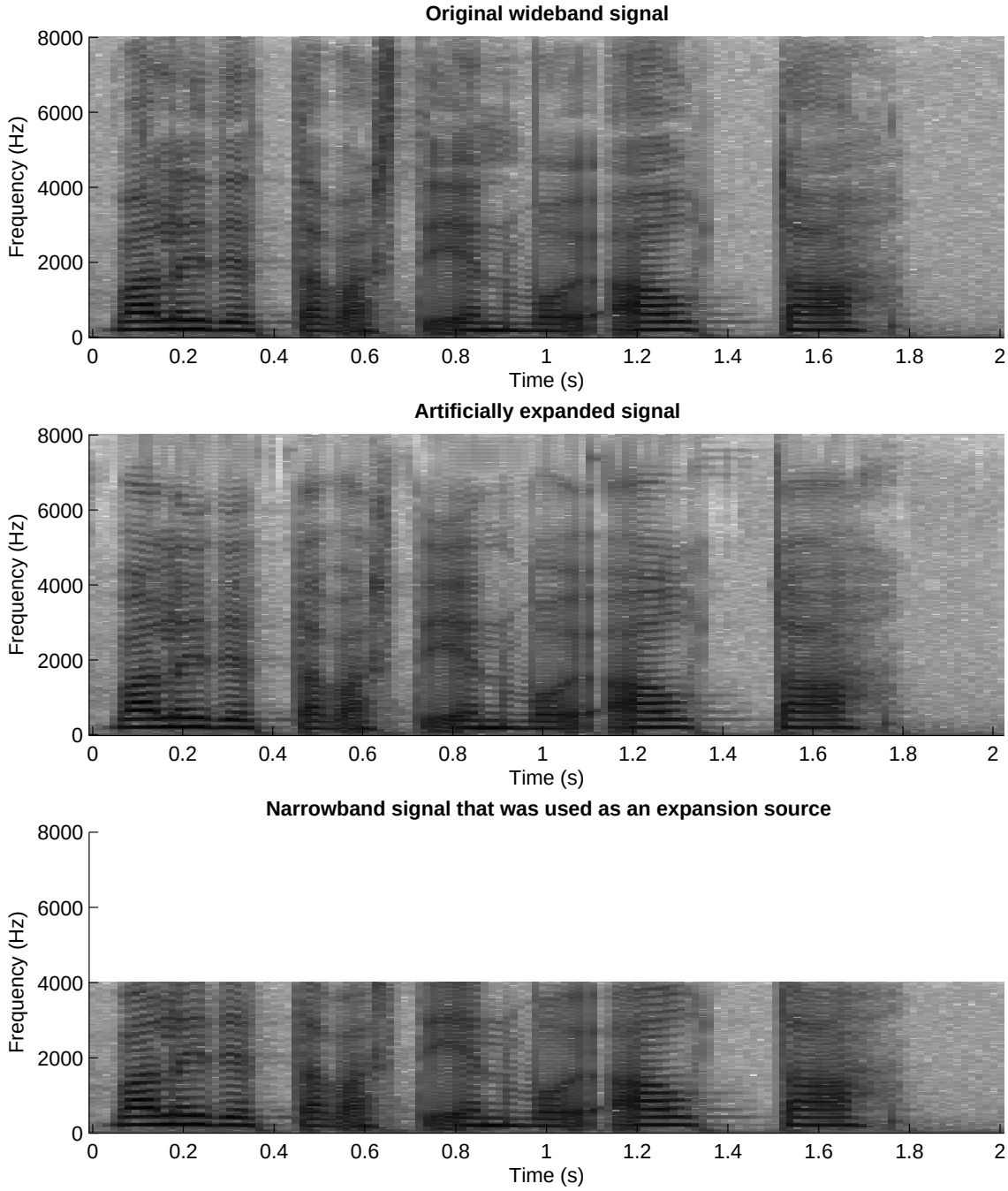



**Figure 5.6:** Spectrograms for wideband, artificially expanded and narrowband samples of a Finnish sentence “Valitettavasti en voi auttaa.” spoken by a female speaker.

### 5.2.2 Low Level Spectral View on Expansion Results

To evaluate the expansion performance on different phonemes, some clear single phoneme frames were searched. Figure 5.7 displays magnitude spectra of signal frames[2] containing /a/, /t/ and /s/. Stop consonant frame /t/ is from silent part[3] of this plosive, because sufficient attenuation for the silent part is important and has caused problems in some of the prior artificial bandwidth expanders (see for example [30]).

As can be seen, the system models vowel sounds quite well, although the 6–7 kHz stripe that was found in spectrogram is clearly visible here as well. Nevertheless the produced expansion is quite accurate up to 7 kHz, the working range of the expander.

The stop consonant expansion is also good, producing sufficient attenuation, always erring on the conservative side.

The sibilant expansion leaves something to be desired. As was noted earlier, the expansion of /s/ is too conservative, leading to a bit muffled /s/. However, even with this weak expansion the sibilant does not sound disturbing, probably partly because of the following sharp plosive.

The examples presented here should be seen as just that, examples. For example, the NEABE /s/ expansions produced with the feedback parameter set range from underexpansion, like the one here, to overexpansion, which produces a metallic sounding /s/. Vowel and stop consonant expansion processes are generally more robust, typically not producing any audible artefacts, but they are also easier expansion tasks. The negative spectral tilt in them leads to lower power in the highband, which reduces the quality requirements for the expansion band.

### 5.2.3 Controller Behavior

To learn how the expander works and what it does to achieve the results shown above, the control points of the magnitude shaping curve produced by the neural network were plotted for best individuals of different generations. The control point traces are shown in Figure 5.8.

In addition to neural network controlled control points shown in the figure, the expansion had a fixed control point at 4 kHz with 0dB amplification to ensure that the expansion is continuous over the 4 kHz mirror point, and a fixed control point of $-60$ dB at 8 kHz to fix the behavior outside the 7 kHz expansion range.

Controllers from all generations seem to have a fixed default attenuation the controller holds, unless it is reacting to something special in the input. In all other control points except point 4, the default level seems to decrease as the evolution continues. For the last generation controller, the default levels are around $-8$, $-18$, $-27$, $-31$ and $-80$ dB.

The control for the first control point is roughly similar between the 56th and 250th gen-

[2] As mentioned in chapter 4, there are two kinds of frames in NEABE, one used for expansion and other for fitness evaluation. The frames displayed here are fitness evaluation frames.

[3] The plosive sounds, like /k/, /p/ and /t/ for instance, consist of a silent part, where the air pressure is impounded in the mouth, and aburst part, where the pressure is abruptly released.

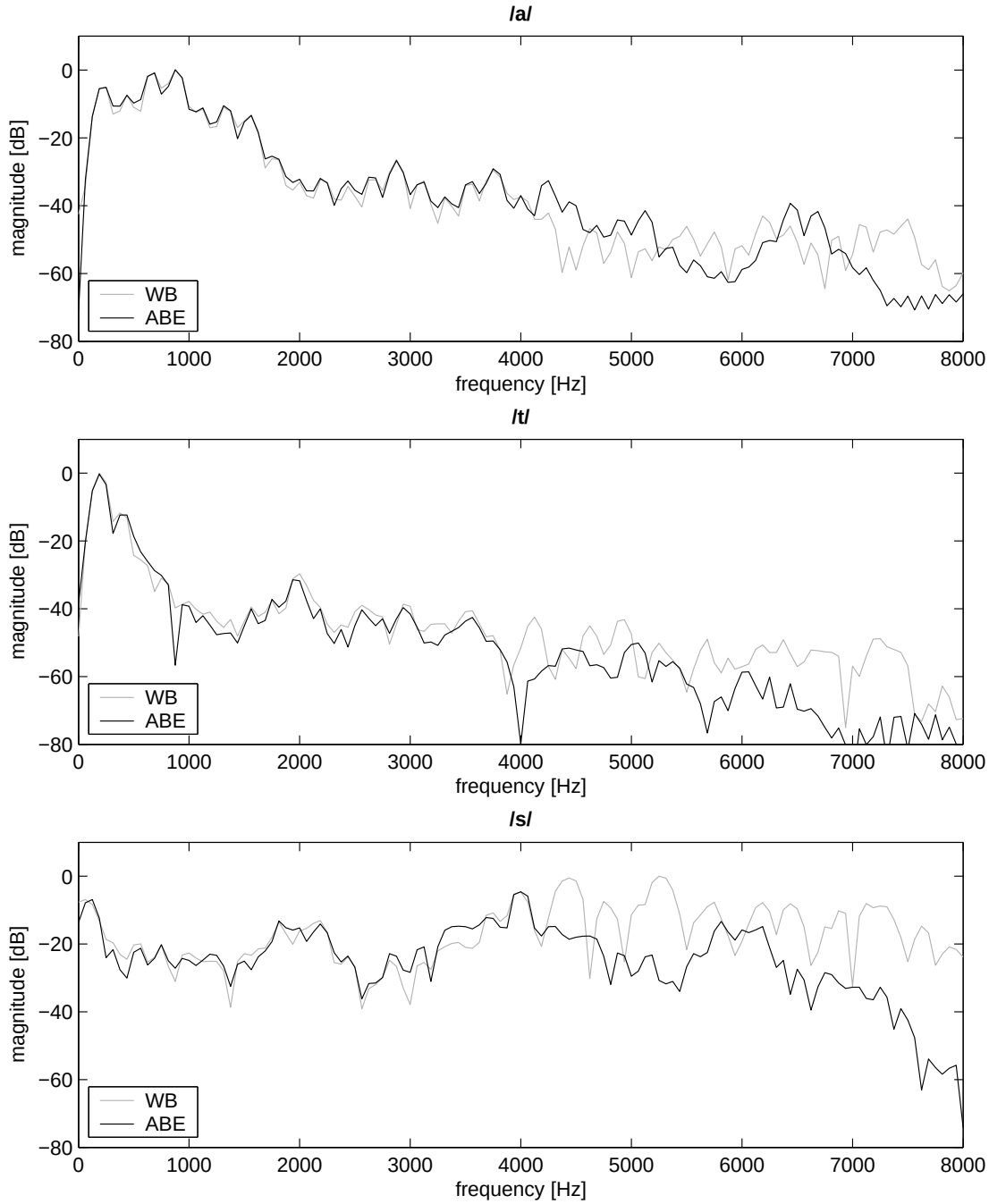



**Figure 5.7:** Magnitude spectra for /a/, /t/ and /s/. 0 dB point has been set to the highest magnitude of the original wideband signal in each plot. The /t/ frame is from the silent part of the plosive. WB represents the original wideband spectrum and ABE the artificially expanded spectrum. The frames are taken from a Finnish sentence "V**a**li**t**tettava**s**ti en voi auttaa." spoken by a female speaker.

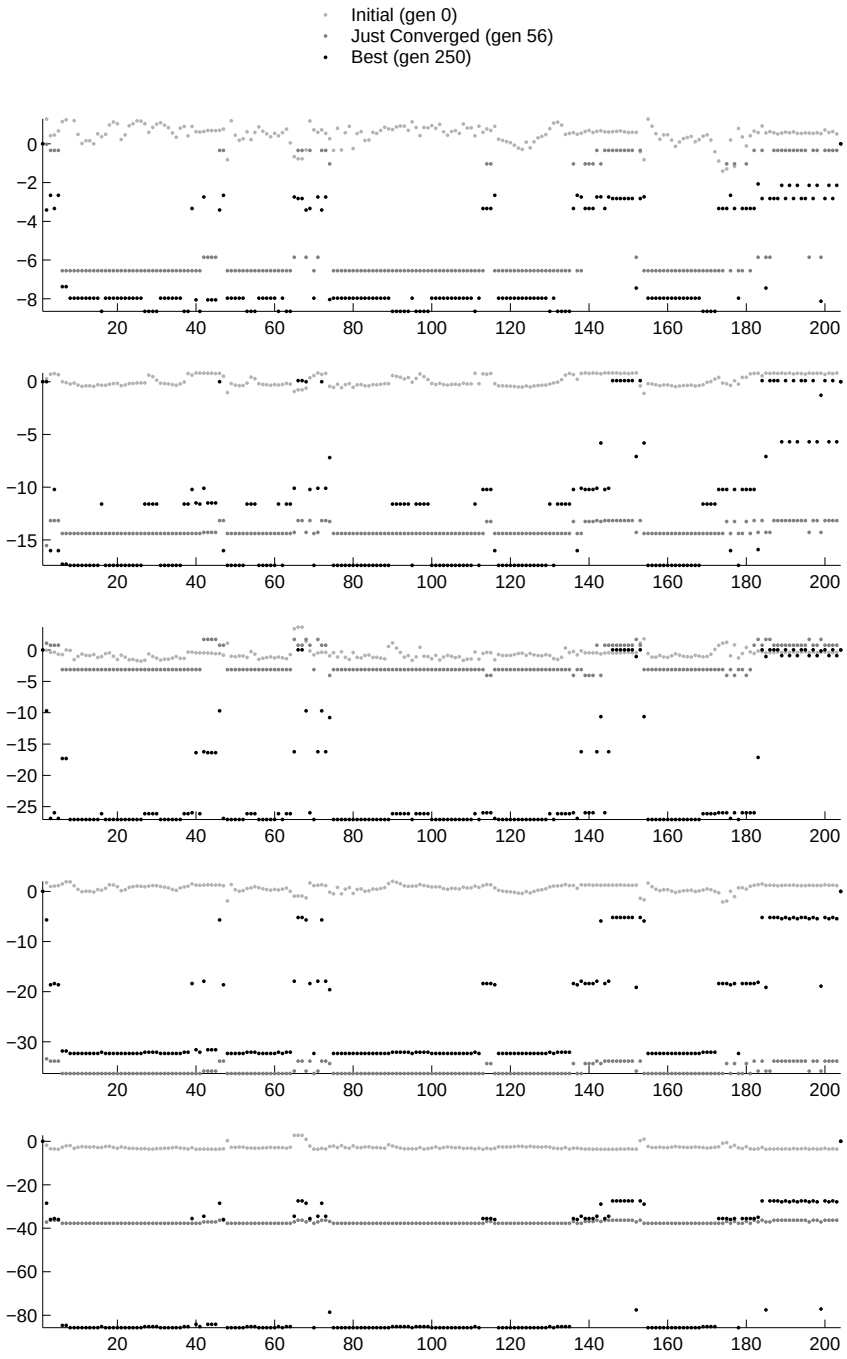



**Figure 5.8:** Traces of magnitude shaping spline control points in Finnish phrase "Valitettavasti en voi auttaa." spoken by a female speaker. Amplification at five control points (from top down), controlling shaping spline at 4.6, 5.2, 5.9, 6.7 and 7.6 kHz respectively, are shown. Amplification in dB is shown in y-axis, while x-axis has the number of the expansion frame. Best individuals from three different parts of evolution are shown. The first individual is from the initial random population, the second just after the fastest learning phase has passed and the last is the final best individual of the evolution.

eration controller. The 250th generation controller has generally a bit more attenuation and sometimes, unlike the 56th generation controller, increases attenuation from the default level. In addition, the 250th generation controller is more fine tuned to the signal, changing attenuation level more frequently.

Second control point has developed much more during the evolution. The 56th generation controller generally holds a constant level around $-13$ to $-15$ dB range, whereas the 250th generation controller has multiple attenuation levels which it uses to control the modulation in different types of frames. One of the levels, presumably used for fricative type frames, keeps the attenuation at 0 dB. The dynamic range of control point has thus inreased a lot from the 56th generation controller. Like for control point 1, the control has also become more fine tuned.

Third control point is similar to second, except that the default attenuation level has increased a lot from 56th generation version, whereas for control point two the attenuation increase was smaller. In general the third control point has started to attenuate more, only bringing the attenuation down for fricative frames, where it sometimes actually amplifies the lowband to highband transformed signal.

Fourth control point mimics the behavior of the third one. The attenuation levels are different, but the behavior is similar. An interesting detail here is that the attenuation has decreased from the 56th generation version. Probably the 56th generation version has been only roughly tuned and the control of the fourth control point has followed the control of the last control point. It is also good to note that control points are by no means independent. In cubic spline, four neighboring control points are used to set the point when interpolating. Two neighboring points control the path and two more distant points the curvature. Therefore, the 56th generation control has produced aproximately constant level of attenuation for areas between the fourth and the fifth control point, whereas the 250th generation control produces clearly descending modulation curve.

On the 56th generation controller, the fifth control point has had constant value of about $-40$ dB, whereas the 250th generation controller has a lot more amplification, but reacts to some points, presumably fricatives, by decreasing attenuation. The default attenuation of over 80 dB is more than that of the fixed sixth control point, which is at $-60$ dB. The system has for some reason benefited from extra attenuation, most probably because it has caused the spline between fourth and fifth control point to descend more rapidly. It is also good to note that the last control point is outside the range evaluated by the fitness evaluation module, which ends at 7.5 kHz. Therefore it only has to worry about fitting a part of the fourth control point side segment as well as possible to the original wideband signal. Due to this, the control point will overattenuate to get a better fit for the segment that is evaluated in the fitness evaluation, even if this greatly reduces the fit in other, unevaluated, spectral regions. However, the overattenuation is not problematic, as it is not audible due to other, louder spectral components masking it.

It should be noted that the fifth control point needs rather great attenuation even on fricative frames, because the LHTF mirrors low frequencies of the low band to the high frequencies of the high band and therefore there is quite a lot of spectral power on this frequency range to begin with. Therefore, after the $25$ dB attenuation this region may have about equal power with the fourth control point region after $0$ dB attenuation.

Overall, the control points have surprisingly quantized levels. There are relatively few smooth frame-to-frame transformations, the control points seem to react clearly or not at all. One reason for this might be a poor feature set. If the features do not represent encountered situations with enough resolution, a fixed region based controller may be the most robust alternative and therefore yield the best results.

Another possible reason is insufficient power to learn more complex relationships well enough to produce better results than those achieved with simple classifier style controllers. Using evolutionary learning method that can protect innovations and recombine individuals without suffering from competing conventions problem might enable the system to learn more fine tuned control with smoother control point transitions. Adding more control points would also make learning smoother control point transitions easier. However, there is a tradeoff as it would also increase the risk of overtraining (adapting to variations of learning data instead of modeling the phenomenon producing the data) and make learning more difficult due to the increased search space dimensionality.

Naturally, some transitions in the signal should be abrupt. For example the transitions from silent part of the plosives to the noisy part is fast. On frame level the speech is rather stationary, so most of the transitions happen between the phonems. However, coarticulatory effects[4] should ensure that the spectral properties of the speech are changing all the time and therefore the magnitude shaping spline should change as well.

## 5.3 Subjective Speech Quality Analysis

Maximizing the subjective expansion quality is the final goal in the artificial bandwidth expansion. To test the subjective quality, a small scale listening test was arranged. While it was not possible in the present project to arrange a listening test with enough listeners for the test to be statistically significant, it gives some idea about the expansion quality and, perhaps more importantly, the comments given by the test subjects help to direct the future research. By exploring the issues which the subjects found especially annoying, further understanding on what is important in the expansion task can be achieved.

[4]Coarticulation refers to articulators moving to predict the upcoming phonation while still performing the previous phonation. This causes phonemes to have different characteristics depending on their context, the phonemes that are surrounding them.

### 5.3.1 Listening Test

The test was arranged in the Laboratory of Acoustics and Audio Signal Processing at Helsinki University of Technology, in a listening room which fits the ITU-R BS.116 standard. The listening test software used was GuineaPig 2 [25]. The samples were listened using Sennheiser HD580 headphones. In both parts of the two part test, 13 naive listeners (10 men and 3 women) listened to mono sample pairs with both ears. They were allowed to listen to the samples as many times as they liked, in the order they preferred.

The test consisted of two parts and a few items long practice period before both parts, during which the user could adjust the volume and practice listening and answering. A short break was held between the two parts. After the test the users were asked if they have any comments about the test samples or the test itself.

All test samples were in Finnish and also all subjects were native Finnish speakers. The samples used for testing were not used to train the expander, but the practicing samples were from the expander training speech database.

Before the test, the user was given a written instruction text. A copy of the instruction text and an English summary can be found in appendix A.

#### Pair Comparison Test

In the first part of the test, the subjects were asked to decide which one they would rather listen to (in Finnish “Kumpaa kuuntelisit mieluummin?”). They also had the option of choosing that the samples sounded about the same (in Finnish “melkein sama”), indicating no preference for either of the samples. The user interface of the test is shown in Figure 5.9.

The test pairs consisted of a narrowband sample and an artificially expanded wideband sample. The narrowband sample was produced by lowpass filtering and downsampling the original 16 kHz sample rate wideband sample into an 8 kHz sample rate narrowband sample. The artificially expanded wideband sample was then produced by expanding the narrowband sample. The narrowband sample was then resampled to 16 kHz sample rate, because of the GuineaPig requirements (all test samples in GuineaPig must have the same sample frequency).

The sentences used in the test are shown in Table 5.1. The wideband samples were good quality recordings of clean speech without any noise or low-bitrate speech coding, sampled at 16 kHz sample rate and lowpass filtered with a cutoff around 7.5 kHz.

The test pairs were presented in random order. All pairs were presented twice, exchanging the pair item order for the second time. In addition, two null pairs were used. One of them had the narrowband version of sentence *p6* twice and the other the artificially expanded version of sentence *p7* twice. Thus, in total there were 18 test items in the first test, and 2 test items in the practice period before that.

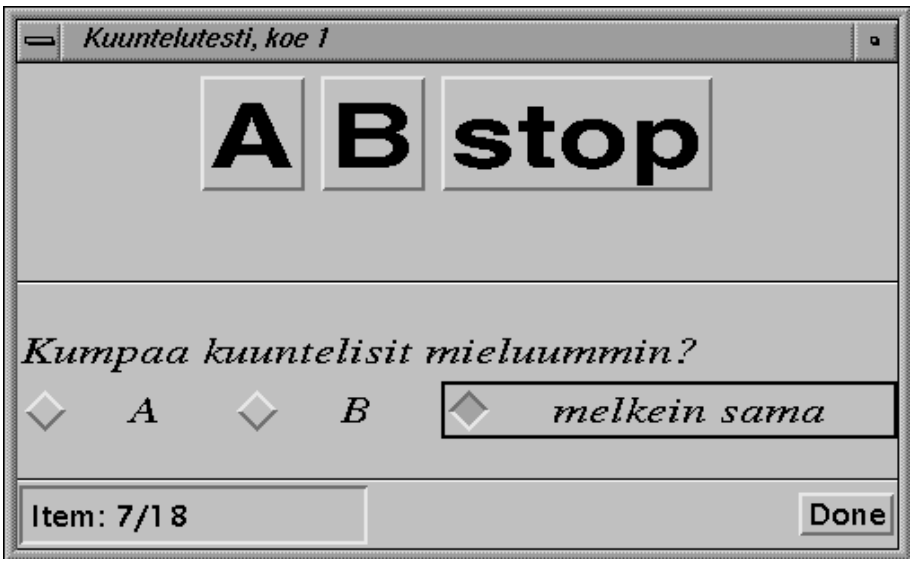



**Figure 5.9:** User interface of the pair comparison test. Pressing *A* or *B* button plays the corresponding test sample. The user may play the next sample while the previous one is still playing, in which case the previous sample stops and the next sample is played. Pressing *stop* stops the sound currently playing. After listening to both of the samples at least once, the answer area is activated. The user selects either *A*, *B* or *melkein sama* and presses *done* to continue to the next pair. The question is shown between the listening and answer areas. The number of pairs completed and the total number of pairs are shown in the lower left corner.

| Number | Speaker | Sentence |
|---|---|---|
| p1 | female 1 | Kenkien tulisi olla väljät. |
| p2 | female 2 | Varas sieppasi timanttisormuksen. |
| p3 | female 3 | Minä olen opiskelija. |
| p4 | female 4 | En tiedä mikä tämä on. |
| p5 | male 1 | Asiantuntijat olivat yksimielisiä. |
| p6 | male 2 | Aurinko nousee idästä. |
| p7 | male 3 | Kissa näkee hyvin pimeässä. |
| p8 | male 4 | Koe onnistui yli odotusten. |

**Table 5.1:** Sentences used in the pair comparison test.

| Grade | Finnish Description | English Description |
|---|---|---|
| 3 | Paljon parempi | Much Better |
| 2 | Parempi | Better |
| 1 | Vähän parempi | Slightly Better |
| 0 | Melkein sama | About the Same |
| -1 | Vähän huonompi | Slightly Worse |
| -2 | Huonompi | Worse |
| -3 | Paljon huonompi | Much Worse |

**Table 5.2:** The CCR test scale. The Finnish descriptions were used in the test, the corresponding English descriptions from the ITU-T recommendation P.800 are given as a reference.


**Figure 5.10:** User interface of the CCR test. Pressing *A* or *B* button plays the corresponding test sample. The user may play the next sample while the previous one is still playing, in which case the previous sample stops and the next sample is played. Pressing *stop* stops the sound currently playing. After listening to both of the samples at least once, the answer area is activated. The user selects the quality of sample B compared to the quality of sample A on the slider. The slider can be either clicked or dragged. When released, the slider moves to the nearest integer position. The verbal description of the selected choice appears below the slider. After selecting the grade, user presses *done* to continue to the next pair. The number of pairs completed and the total number of pairs are shown in the lower left corner.

### Comparison Category Rating Test

In the second part of the test, the listeners were asked to grade the quality of the sample B compared to the quality of the sample A in the scale given in Table 5.2. The Finnish descriptions for the grades were used. The test user interface is shown in Figure 5.10.

| Number | Speaker | Sentence |
|---|---|---|
| c1 | female 1 | Kaikki odottavat talouden elpymistä. |
| c2 | female 3 | Käärme nieli saaliinsa. |
| c3 | female 4 | Valitettavasti en voi auttaa. |
| c4 | female 5 | Juna lähtee hetken kuluttua. |
| c5 | male 1 | Jätä viesti puhelinvastaajaan. |
| c6 | male 5 | Poliisi tutkii asiaa. |
| c7 | male 6 | Ennakkosuosikki voitti kilpailun. |
| c8 | male 7 | Olutta läikkyi lattialle. |

**Table 5.3:** Sentences used in the pair comparison test.

The test resembled the ITU-T Comparison Category Rating (CCR) test specified in [44], with a few exceptions. Most notably, there were only three null pairs, one for each processing type (i.e. narrowband, artificially expanded and wideband) and the MNRU reference conditions (multiplicative noise references which are used to calibrate the judgement scale) were not included in the test.

The test pairs consisted of a wideband sample and either a narrowband sample or an artificially expanded sample. The narrowband and expanded samples were produced as in the pair comparison test. The wideband samples had similar characteristics as the source material in the pair comparison test. The sentences used in the test are shown in Table 5.3. For each sentence, both narrowband vs. wideband and expanded versus wideband pairs were created.

The test pairs were presented in random order. All pairs were presented twice, exchanging the pair item order for the second time. In addition, three null pairs were used. First had the narrowband version of sentence *c5* twice, second the wideband version of sentence *c4* twice and third the artificially expanded version of *c3* twice. In total, there were 35 test items in the second test, and 4 test items in the practice period before that.

### 5.3.2 Test Results

The numerical test results are presented below. The comments the subjects made after the test can be found in appendix B.

#### Evaluating Test Reliability

Due to the small number of subjects in the test, no statistical analysis of the results was done, as the number of subjects is too low for the test to be confident anyway. To get an idea of the reliability of the listeners, the null pair results were analyzed.

In the pair test, five subjects failed at least one null test. After analyzing the results, it was noted that all these had failed the null pair containing the artificially expanded sample.

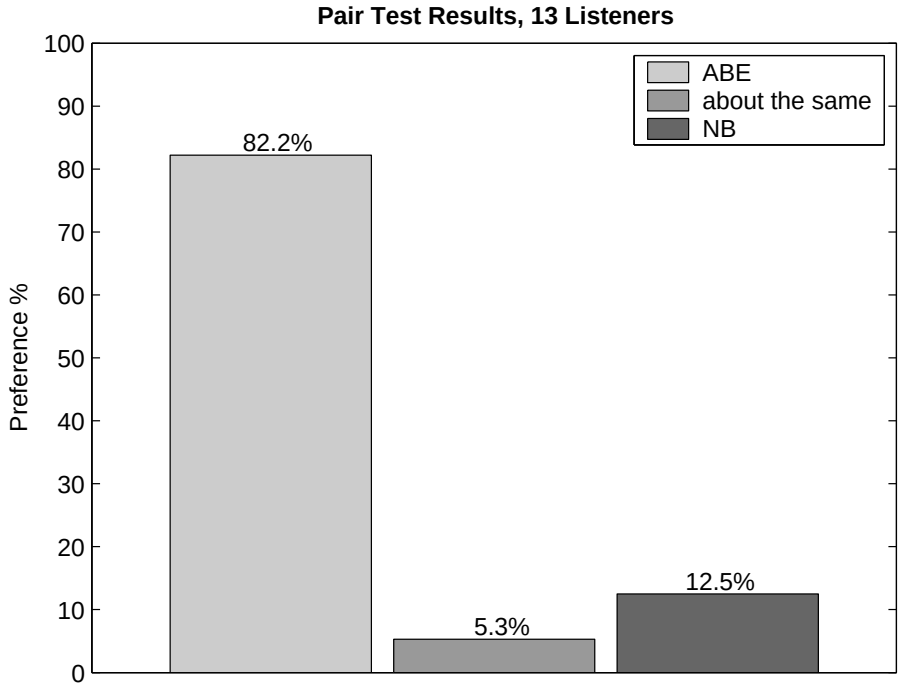



**Figure 5.11:** Preference scores for the pair comparison test. Artificially expanded (ABE) samples are clearly preferred over narrowband (NB) samples.

Listening the sample revealed that it had a disturbing artefact, which might make the sample sound different if it was played again in quick succession.

In the CCR test, three listeners failed two of the null tests, grading them to $+/-1$ instead of zero. However, their answers to test items resembled the answers of others, so they were left in.

### Pair Comparison Test

The pair test preference scores for artificially expanded and narrowband samples are shown in Figure 5.11, along with the portion of samples ranked *about the same*. Artificially expanded samples are clearly preferred.

During results processing it was noted that there were quite many cases, in which the users had preferred one type when the pair was encountered first time and another type when the pair appeared second time. When these were considered to be two *about the same* answers instead, the results changed considerably. The results calculated in this way are shown in Figure 5.12.

To see how the preference was distributed among test samples, results for each sample were calculated. These are shown in Figure 5.13. As can be seen from the results, listeners show particularly strong preference for artificially expanded sound in sample *p5*. The strongest preference for narrowband occurs in sample *p7*.

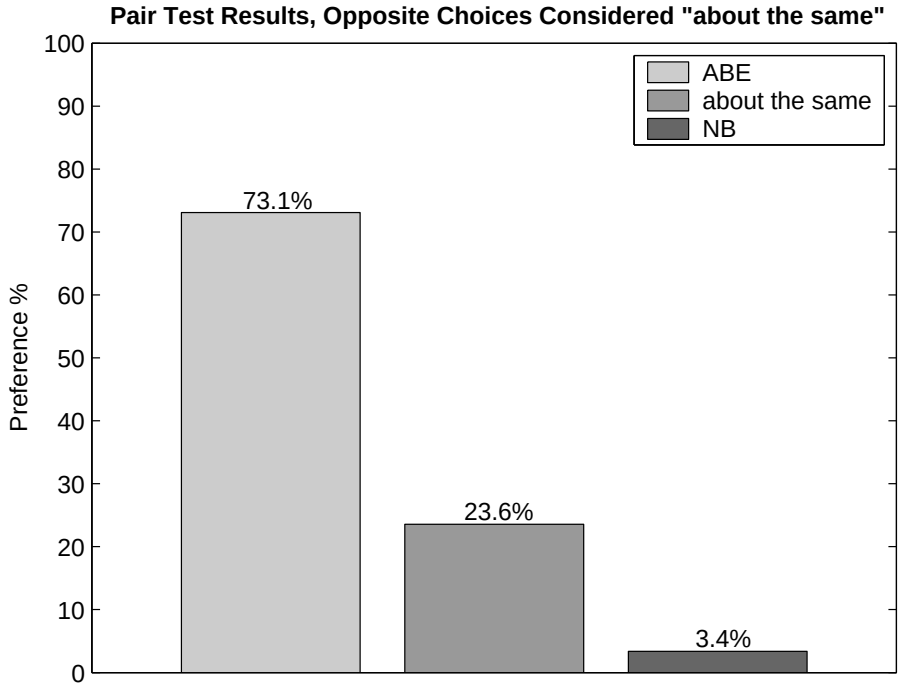



**Figure 5.12:** Pair test preference scores, when opposite preferences for the same pair are interpreted as two *about the same* answers instead. Both the ABE and the NB score decrease 9.1 percentage units.

Three of the subjects preferred ABE in all samples. In addition, one of the subjects preferred ABE over 90% of the time and rated the samples *about the same* rest of the time. Two test subjects preferred narrowband samples clearly more than others, both having narrowband preference around 40 %. However, as neither of them rated any samples (except the null pairs) *about the same*, they still preferred ABE almost 60% of the time. The other listeners preferred narrowband once out of eight sounds.

When the opposite preferences in the same pair were interpreted as *about the same*, the per subject results changed dramatically. Only three subjects had any preference for narrowband at all, and one of them selected *about equal* 60% of the time. All test subjects still preferred ABE more than narrowband.

### CCR Test

Comparison mean opinion scores (CMOS) for the CCR test are shown in Figure 5.14. The difference between the narrowband and ABE CMOS scores is 0.839 points, that is, almost one level.

To see how the preference was distributed among test samples, results for each sample were calculated. These are shown in Figure 5.15. As can be seen from the results, artificially expanded sample *c8* has particularly high CMOS. Also narrowband CMOS is high for the sample.

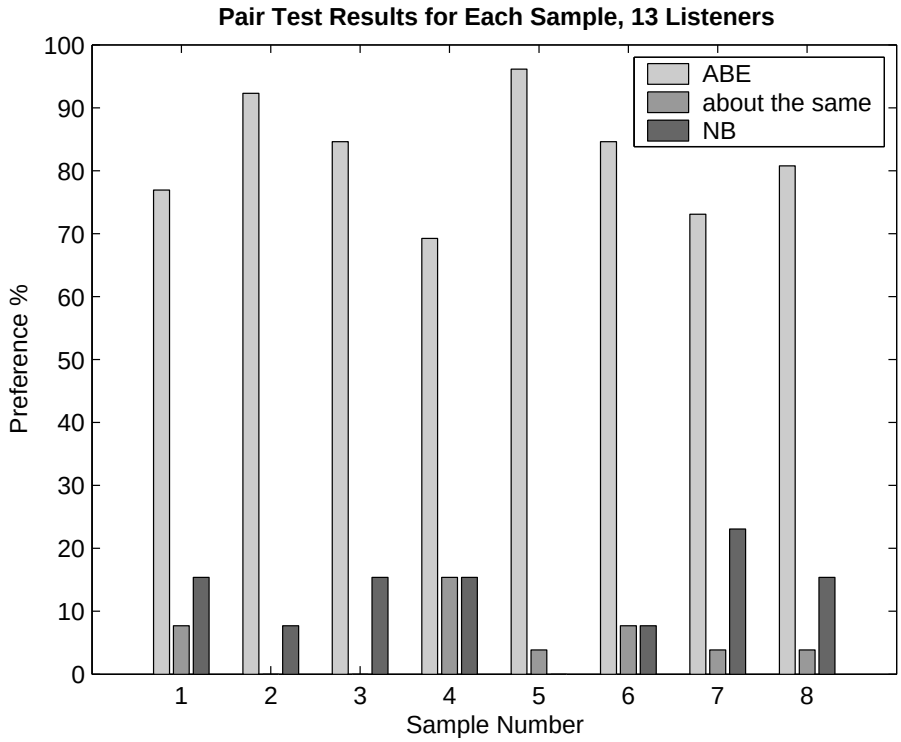



**Figure 5.13:** Pair test results for each sample. Sample five shows particularly strong preference for ABE. Narrowband gathers most support in sample seven.

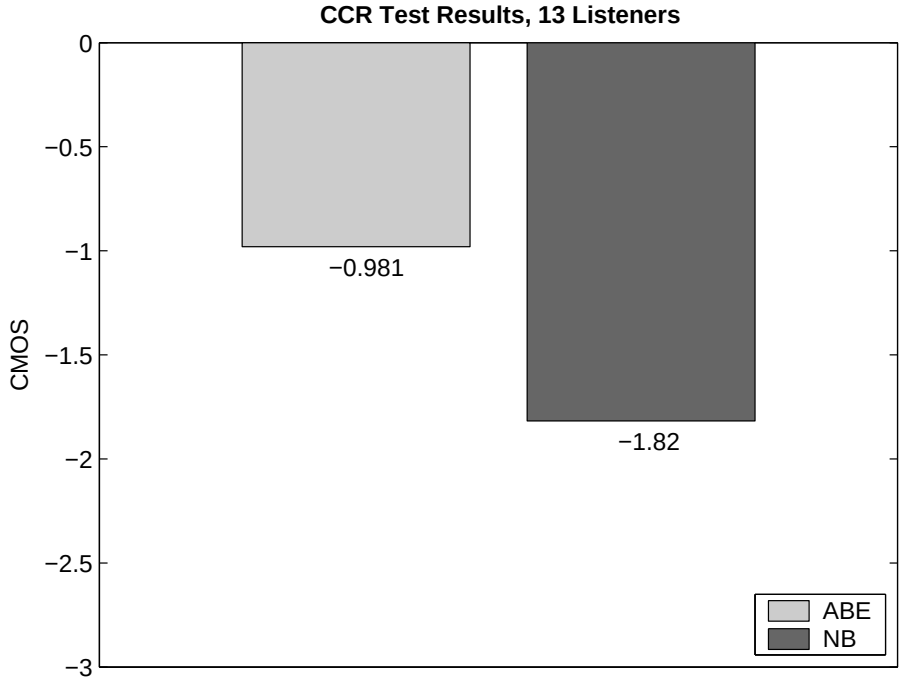



**Figure 5.14:** CCR test comparison mean opinion scores for narrowband and artificially expanded samples. Artificially expanded samples have significantly higher CMOS.

The expansion performs worst in sample two. The difference between expanded and narrowband samples is smallest in sample *c5*, but ABE still has higher CMOS. The difference is largest in sample *c2*.

Expanded sample CMOS scores do not seem to directly follow narrowband CMOS scores, although some correlation does exist. Poor narrowband CMOS seems to be dampened by ABE, leading to only slight CMOS reduction in expanded samples.

Two listeners assigned ABE a CMOS over $-0.5$. Three listeners gave it a CMOS of $-1.5$ or less. All subjects considered narrowband CMOS to be under $-1.35$. No one rated ABE worse than narrowband, the smallest difference was 0.25 CMOS points, while the largest was 1.5 CMOS points.

One of the subjects who preferred narrowband about 40% of the time in the pair test still graded the ABE samples 0.875 CMOS points higher than the narrowband samples.

In 13 cases a listening test subject could not distinguish between artificially expanded sample and original wideband sample in either of the pairs (that is, the subject rated both WB-ABE comparison and ABE-WB comparison as 0). Eight listeners had at least one of these cases. Five of the listeners could not distinguish difference in sentence *c8*. A single subject gave artificially expanded *c8* sentence a CMOS worse than $-1$. The subject rated it as $-2$. However, when the pair came in reverse order, the subject rated it $+1$.

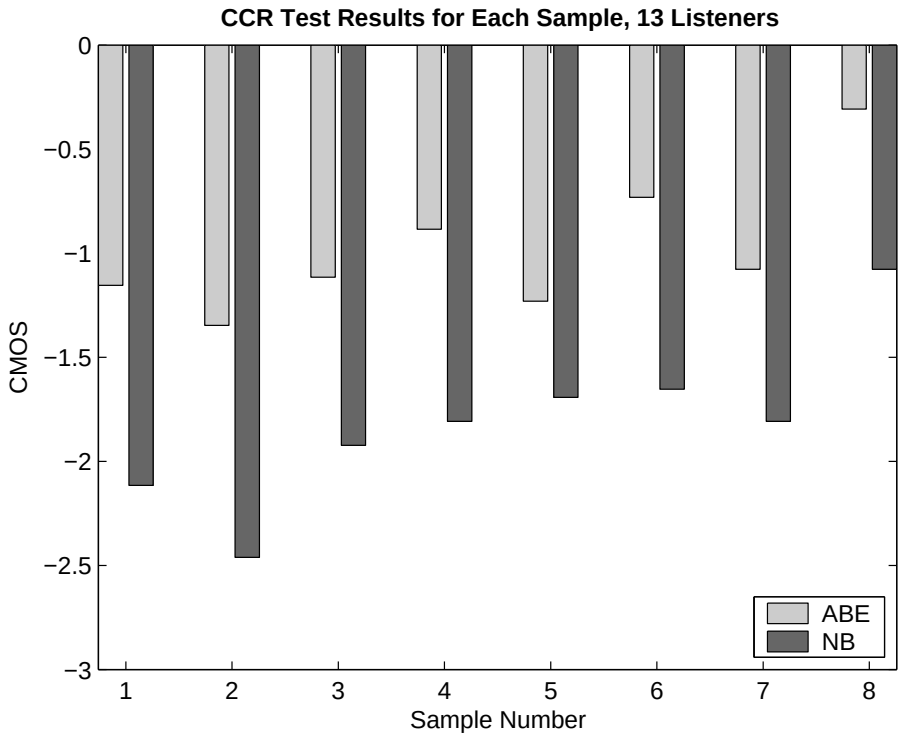



**Figure 5.15:** CCR test comparison mean opinion scores for each sample. Expansion has highest CMOS in sample 8, as does the narrowband. Worst ABE scores are in sample two, in which the narrowband also performs worst. The highest difference between expanded and narrowband samples can be found in sample 2, whereas the difference is lowest in sample 5.

### 5.3.3 Result Analysis

The tests indicate clear preference for the artificial bandwidth expansion system. Even with the low number of subjects the support in the pair tests is so clear that it seems improbable that narrowband speech would be preferred in a larger test. All subjects preferred expanded speech more often than narrowband. The clearly higher average CMOS in CCR coupled with the fact that all subjects rated ABE higher than the narrowband lends further support to the preference for the expansion. As expanded speech is preferred in all samples of both tests, there are no cases where the majority of the listeners would feel that the artificial bandwidth expansion degrades the sound quality. This is clearly visible in pair test results. As shown in Figure 5.12, after opposite choices for the same sentence are considered equivalent to *about the same* the preference for narrowband is only 3.4%.

For some samples, the expansion quality seems to be almost indistinguishable. Sentence *c8* seems to be a prime example. It often scored a 0 and always at least a -1 (with a single exception mentioned in 5.3.2, in which the grading is not very consistent).

#### Problematic /s/ Expansion

What makes *c8* so easy for the expander? Looking at the sentence it can be seen that it has no /s/. There is only one other sample with no /s/, *c4*. Its ABE CMOS score is the third best of all samples. It seems that expansions without /s/ produce better results than expansions with /s/. This leads us to believe that the expansion of /s/ is problematic. From experience with other artificial bandwidth expansion algorithms [28, 30], the sibilants are known to be difficult to expand correctly, so it is not surprising to find their expansion problematic here. Further support for the difficulties in /s/ can be found from the comments made by the listening test subjects. Most of them noted something about the /s/. The /s/ was either too metallic (indicating too much power in the high frequencies) or spoken with a lisp (indicating too little power in the high frequencies).

Interestingly, *c8* has much better CMOS than *c4*, even though neither of them has any /s/. However, *c4* is spoken by a female speaker with (according to some listeners) an annoying way of speaking. One test subject mentioned that the expansion fits better for male than for female speakers. According to him, if the voice already has a high pitch, it may become annoyingly metallic after expansion. While there are some slight differences in test scores for male and female voices, there is little support for this claim in the test results.

Sentence *p5* seems to be an example of a successful /s/ expansion. It has no notable artefacts and its ABE preference is highest of all pair comparison test sentences. Studying its expansion process might reaveal what makes a sibilant expansion successful.

Sentence *p7* is an archetype of a problematic expansion sentence. It has multiple /s/'s which have been somewhat underexpanded. In addition, one of them produces a long hissing artefact,

which further degrades the sentence quality. When the sentence was expanded with the "direct" parameter set (see section 4.4 for parameter set details), which produces a sharper expansion, the sibilants were not underexpanded, but the artefacts got worse. The hissing can also be heard in the narrowband sample, where it annoys less as there are no content in the high frequency band. Therefore, it seems likely that *p7* is a somewhat pathological expansion case, where the narrowband artefacts prevent proper expansion.

Poor /s/ expansion is the key to relatively low CMOS difference between the expanded sample and the narrowband sample in *c5*. Again there is an audible artefact in the narrowband, which is then emphasized by the expansion process.

#### Exploring the Narrowband Preference in Pair Comparison

As mentioned in 5.3.2, two subjects preferred narrowband samples considerably more than others. For one of them the preference decreased considerably when opposite choices for the same sentence are considered equivalent to selecting *about the same*. For the other, preference after that was still over 20%. However, he graded the ABE samples in CCR test 0.875 CMOS points higher than the narrowband samples, which is over the average difference of 0.839 CMOS points.

There is evidence of a similar phenomenon in other artificial bandwidth expansion quality tests, for example in [30]. People are used to listening to narrowband samples, so especially when no wideband reference is available, they tend to prefer the sound type they have heard before.

However, here the explanation is lacking. If being used to listening to narrowband samples were the cause, the results should be similar for most of the test subjects.

So perhaps the subject simply preferred narrowband samples for their softness. When asked to rate the quality of the samples in CCR, he may have put more emphasis on the clarity of the samples and less on how pleasing the sample sounds.

#### Improving Sound Quality on Poor Narrowband Samples

In section 5.3.2, an interesting phenomenon was found in the CCR test results. As seen in Figure 5.15, even when the narrowband quality is very poor, the expansion quality roughly retains its level. The expanded samples remain around $-1$ CMOS whether the narrowband source is at $-2.5$ CMOS or at $-1.5$ CMOS. While there is far too little data to make any definite conclusions, it could be speculated that the high frequency content added by the expansion improves the perceived quality even when the added content amplifies the narrowband artefacts. This might not be the case with a sharper, more aggressive expander, but with this somewhat conservative expansion strategy the artefacts are not emphasized too much, while enough high frequency power is still added to improve the sound clarity.

# Chapter 6

# Conclusions

In this thesis, an algorithm that uses neuroevolution to produce solutions for the artificial bandwidth expansion problem was presented. The algorithm consists of two separate parts, the online subsystem, which does the actual expansion, and the evolution part, which configures the parameters of the online subsystem to produce high quality results with a given data set. The expansion works by first generating initial high frequency data from the narrowband signal using simple, well known bandwidth expansion methods, and then shaping the highband using a cubic spline based magnitude shaping function controlled by a neural network.

The algorithm behavior was analyzed and the quality of the expansion produced by it evaluated using a listening test. Turned out that the algorithm was capable of yielding high quality artificial expansions for clean speech. The expanded speech was clearly preferred by the listeners in a short listening test, and was rated almost one CMOS point higher in comparison category rating (CCR) test, than the corresponding narrowband source sound, when the original wideband signal was used as a reference. The expansion was also quite robust. The CCR score remained high even when the narrowband score decreased.

More important than the current bandwidth expander and its results is the developed expansion evolution architecture. Because of the modular design, it is easy to modify and adjustments to the behavior of the expander can be made easily. This offers flexibility rarely seen in other, typically hand tuned, expanders. Because of the minimized internal dependencies between different parts of the algorithm, exploring new features sets, lowband to highband transform functions, magnitude shaping functions and neural network architectures is simple. Adapting the expander to new languages, adding postprocessing and tuning the expansion to be more conservative or aggressive can be done by making simple changes to the system and/or the data used in the learning process.

## 6.1 Future Work

For the current implementation, many of the modules were selected to resemble the earlier method to ensure low number of variables in the development phase and thus ease the system implementation. By making selections that were known to work, the idea of automatically evolving solutions could be tested with less risk of system failing because of poorly chosen expansion method. For this reason, there is still a lot of research to be made to realize the full potential of this new method.

Using psychoacoustical metrics in the fitness evaluation would probably improve the expansion results. The system would be directed to model expansion features that are important for the human hearing, instead of aiming for a good expansion in the spectral mean square sense.

The feature set used by the neural network of the expander could be improved. By systematically testing a large set of features available in the literature and searching for a set that yields the best subjective results, the quality of the expansion could be improved. Or instead of trying to come up with feature sets, direct expansion from spectral (or cepstral) components to spectral components could be attempted.

Adjusting the way the system learns would probably be helpful. Changing the neuroevolution algorithm to NEAT or ESP [16] would avoid the competing conventions problem, enabling the efficient use of recombination. More efficient learning algorithm would ensure that the evolved solutions are as efficient as possible, using the underlying expansion mechanism to its fullest extent. Even more importantly, NEAT based evolution would allow tackling search spaces with higher dimensionality, as it starts minimally and therefore evolves efficiently on higher dimension problem spaces as well.

Incremental training, explained by Gomez in [16], could be used to learn more complex expansion models. By evolving solutions to simpler problems before trying to tackle the actual problem, the evolution can be guided to find solutions to problems that would be too hard for it when starting with random population. In this case, incremental training would be an ideal tool when teaching the expansion to handle real noisy telephone signals.

The expansion consistency could possibly be improved by introducing artificial noise to the evolution process. As described by Gomez[16], making the controller behavior noisy often forces the evolved solutions to be robust.

In addition to the ideas mentioned before, ideal structure of the expander could be searched for. Avoiding the spectral gap induced by the lowband to highband transform function, improving the magnitude shaping function and searching for an effective neural network architecture would all improve the potential expansion quality the expander could achieve.

# Bibliography


[1] Carlos Avendano, Hynek Hermansky, and Eric A. Wan. Beyond nyquist: Towards the recovery of broad-bandwidth speech from narrow-bandwidth speech. In *Proceedings of the European Conference on Speech Communication and Technology*, Madrid, Spain, September 1995.

[2] James E. Baker. Reducing bias and inefficiency in the selection algorithm. In *Proceedings of the Second International Conference on Genetic Algorithms and their Application*, pages 14–21, Hillsdale, NJ, 1987. Lawrence Erlbaum Associates.

[3] Thomas Bäck. Optimal mutation rates in genetic search. In *Proceedings of the Fift International Conference on Genetic Algorithms*, pages 2–8, San Mateo, CA, 1993. Morgan Kaufmann Publishers.

[4] Theodore C. Belding. The distributed genetic algorithm revisited. In *Proceedings of the 6th International Conference on Genetic Algorithms*, San Francisco, CA, 1995. AISB, Morgan Kaufmann.

[5] Tobias Blickle and Lothar Thiele. A comparison of selection schemes used in genetic algorithms. Technical report, Swiss Federal Institute of Technology, December 1995. 2nd edition.

[6] Cheung-Fat Chan and Wai-Kwong Hui. Quality enhancement of narrowband celp-coded speech via wideband harmonic re-synthesis. In *Proceedings of the International Conference on Acoustics, Speech and Signal Processing*, volume 2, pages 1187–1190, Munich, Germany, April 1997. IEEE.

[7] Kumar Chellapilla and David B. Fogel. Evolution, neural networks, games, and intelligence. *Proceedings of the IEEE*, 87(9):1471–1496, September 1999.

[8] George V. Cybenko. Approximation by superpositions of a sigmoidal function. *Mathematics of Control, Signals, and Systems*, 2(4):303–314, 1989.

[9] Paul J. Darwen. Computationally intensive and noisy tasks: Co-evolutionary learning and temporal difference learning on backgammon. In *Congress on Evolutionary Computation*, pages 872–879, San Diego, CA, July 2000.

[10] Niklas Enbom and W. Bastiaan Kleijn. Bandwidth expansion of speech based on vector quantization of the mel frequency cepstral coefficients. In *Proceedings of the IEEE Workshop on Speech Coding*, pages 171–173, Porvoo, Finland, June 1999. IEEE.

[11] Julien Epps and W. Harvey Holmes. A new technique for wideband enhancement of coded narrowband speech. In *Proceedings of the IEEE Workshop on Speech Coding*, pages 174–176, Porvoo, Finland, June 1999. IEEE.

[12] Carlos M. Fonseca and Peter J. Fleming. An overview of evolutionary algorithms in multiobjective optimization. *Evolutionary Computation*, 3(1):1–16, 1995.

[13] F. Dan Foresee and Martin T. Hagan. Gauss-newton approximation to bayesian learning. In *Proceedings of the IEEE International Conference on Neural Networks*, volume 3, pages 1930–1935, Houston, TX, June 1997. IEEE.

[14] Valerie J. Gillet, Wael Khatib, Peter Willet, Peter J. Fleming, and Darren V. S. Green. Combinatorial library design using a multiobjective genetic algorithm. *Journal of Chemical Information and Computer Sciences*, 42(2):375–385, 2002.

[15] David E. Goldberg and Jon Richardson. Genetic algorithms with sharing for multimodal function optimization. In *Proceedings of the Second International Conference on Genetic Algorithms and their Application*, pages 41–49, Hillsdale, NJ, 1987. Lawrence Erlbaum Associates.

[16] Faustino John Gomez. *Robust Non-linear Control through Neuroevolution*. PhD thesis, The University of Texas at Austin, August 2003.

[17] Rafael C. Gonzales and Richard E. Woods. *Digital Image Processing*. Addison-Wesley, 1992.

[18] Isabelle Guyon and André Elisseeff. An introduction to variable and feature selection. *Journal of Machine Learning Research*, 3:1157–1182, March 2003.

[19] Peter J. B. Hancock. An empirical comparison of selection methods in evolutionary algorithms. In *Proceedings of the AISB workshop on Evolutionary Computation*, Selected Papers, Lecture Notes in Computer Science, pages 80–94, Leeds, U.K., March 1994. AISB.

[20] Simon Haykin. *Neural Networks – A Comprehensive Foundation*. Prentice Hall, San Diego, 2. edition, 1999.

[21] Donald Hearn and M. Pauline Baker. *Computer Graphics, C Version*. Prentice Hall, 2. edition, 1997.

[22] Francisco Herrera and Manuel Lozano. Adaptive genetic operators based on coevolution with fuzzy behaviors. *IEEE Transactions on Evolutionary Computation*, 5(2):149–165, April 2001.

[23] Mitsuhiro Hosoki, Takayuki Nagai, and Akira Kurematsu. Speech signal band width extension and noise removal using subband hmm. In *Proceedings of the International Conference on Acoustics, Speech and Signal Processing*, volume 1, pages 245–248, Orlando, FL, May 2002. IEEE.

[24] Aki Härmä, Matti Karjalainen, Lauri Savioja, Vesa Välimäki, Unto K. Laine, and Jyri Huopaniemi. Frequency-warped signal processing for audio applications. *Journal of the Audio Engineering Society*, 48(11):1011–1031, November 2000.

[25] Jussi Hynninen and Nick Zacharov. Guineapig - a generic subjective test system for multichannel audio. In *106th Audio Engineering Society Convention*, Munich, Germany, May 1999.

[26] Bernd Iser and Gerhard Schmidt. Neural networks versus codebooks in an application for bandwidth extension of speech signals. In *Proceedings of the European Conference on Speech Communication and Technology*, Geneva, Switzerland, September 2003.

[27] Peter Jax. *Enhancement of Bandlimited Speech Signals: Algorithms and Theoretical Bounds*. PhD thesis, Rheinisch-Westfälische Technische Hochschule Aachen, 2002.

[28] Peter Jax and Peter Vary. Wideband extension of telephone speech using a hidden markov model. In *Proceedings of the IEEE Workshop on Speech Coding*, pages 133–135, Delavan, WI, September 2000. IEEE.

[29] Peter Jax and Peter Vary. On artificial bandwidth extension of telephone speech. *Signal Processing*, 83(8):1707–1719, August 2003.

[30] Laura Kallio. Artificial bandwidth expansion of narrowband speech in mobile communications systems. Master's thesis, Helsinki University of Technology, 2002.

[31] Ray D. Kent and Charles Read. *The Acoustic Analysis of Speech*. Singular Publishing Group, San Diego, 1992.

[32] Martin F. Møller. A scaled conjugate gradient algorithm for fast supervised learning. *Neural Networks*, 6:525–533, 1993.

[33] Wolfgang Maass. Networks of spiking neurons: The third generation of neural network models. *Neural Networks*, 10(9):1659–1671, 1997.

[34] John Makhoul and Michael Berouti. High-frequency regeneration in speech coding systems. In *Proceedings of the International Conference on Acoustics, Speech and Signal Processing*, pages 428–431, Washington, DC, May 1979. IEEE.

[35] John D. Markel and Augustine H. Gray, Jr. *Linear Prediction of Speech*. Communication and Cybernetics Series. Springer-Verlag, Berlin, 1976.

[36] Paul Masri and Andrew Bateman. Identification of nonstationary audio signals using the fft, with application to analysis-based synthesis of sound. In *Proceedings of the IEE Colloquium on "Audio Engineering"*. IEE, May 1995.

[37] Melanie Mitchell, John H. Holland, and Stephanie Forrest. When will a genetic algorithm outperform hill climbing. In Jack D. Cowan, Gerald Tesauro, and Joshua Alspector, editors, *Advances in Neural Information Processing Systems*, volume 6, pages 51–58, San Francisco, CA, 1994. Morgan Kaufmann.

[38] David E. Moriarty. *Symbiotic Evolution of Neural Networks in Sequential Decision Tasks*. PhD thesis, The University of Texas at Austin, May 1997.

[39] David E. Moriarty and Risto Miikkulainen. Efficient reinforcement learning through symbiotic evolution. *Machine Learning*, 22:11–32, 1996.

[40] Heinz Mühlenbein. The breeder genetic algorithm-a provable optimal search algorithm and its application. In *Proceedings of the Colloquium on Applications of Genetic Algorithms*, IEE 94/067, pages 5/1 – 5/3, London, U.K., March 1994. IEE.

[41] Heinz Mühlenbein and Dirk Schlierkamp-Voosen. Predictive models for the breeder genetic algorithm: Continuous parameter optimization. *Evolutionary Computation*, 1(1):25–49, 1993.

[42] Heinz Mühlenbein and Dirk Schlierkamp-Voosen. The science of breeding and its application to the breeder genetic algorithm bga. *Evolutionary Computation*, 4(1):335–360, 1994.

[43] Derrick Nguyen and Bernard Widrow. Improving the learning speed of 2-layer neural networks by choosing initial value of the adaptive weights. In *Proceedings of the IEEE International Joint Conference on Neural Networks*, volume 3, pages 21–26, Ann Arbor, MI, July 1990. IEEE.

[44] ITU-T Recommendation P.800. Methods for subjective determination of quality, August 1996.

[45] Kun-Youl Park and Hyung Soon Kim. Narrowband to wideband conversion of speech using gmm based transformation. In *Proceedings of the International Conference on Acoustics, Speech and Signal Processing*, volume 3, pages 1843–1846, Istanbul, Turkey, June 2000. IEEE.

[46] Jürgen W. Paulus. Variable rate wideband speech coding using perceptually motivated thresholds. In *Proceedings of the IEEE Workshop on Speech Coding for Telecommunications*, pages 35–36, Annapolis, MD, September 1995. IEEE.

[47] Hartmut Pohlheim. Geatbx: Genetic and evolutionary algorithm toolbox for use with matlab. http://www.geatbx.com/docu/index.html, 1998. Referred May 25 2004.

[48] Yasheng Qian and Peter Kabal. Dual-mode wideband speech recovery from narrowband speech. In *Proceedings of the European Conference on Speech Communication and Technology*, Geneva, Switzerland, September 2003.

[49] Lawrence R. Rabiner and Ronald W. Schafer. *Digital Processing of Speech Signals*. Signal Processing Series. Prentice-Hall, New Jersey, 1978.

[50] Colin R. Reeves and Jonathan E. Rowe. *Genetic Algorithms: Principles and Perspectives: A Guide to GA Theory*. Operations Research/Computer Science Interfaces Series. Kluwer Academic, Boston, 2003.

[51] Joseph Reisinger, Kenneth O. Stanley, and Risto Miikkulainen. Evolving reusable neural modules. In *Proceedings of the Genetic and Evolutionary Computation Conference*, New York, NY, July 2004. Springer-Verlag.

[52] Craig Reynolds. Evolutionary computation and its application to art and design. http://www.red3d.com/cwr/evolve.html, 2002. Referred May 25 2004.

[53] Martin Riedmiller and Heinrich Braun. A direct adaptive method for faster backpropagation learning: The rprop algorithm. In *Proceedings of the IEEE International Conference on Neural Networks*, volume 1, pages 586–591, San Francisco, CA, March 1993. IEEE.

[54] Jocelyn Sietsma and Robert J. F. Dow. Neural net pruning – why and how. In *Proceedings of the IEEE International Conference on Neural Networks*, volume 1, pages 325–333. IEEE, July 1988.

[55] Ing Yann Soon, Soo Ngee Koh, C. K. Yeo, and W. H. Ngo. Transformation of narrowband speech into wideband speech with aid of zero crossings rate. *Electronics Letters*, 38(24):1607–1608, November 2002.

[56] Kenneth O. Stanley and Risto Miikkulainen. Evolving neural networks through augmenting topologies. *Evolutionary Computation*, 10(2):99–127, 2002.

[57] Kenneth O. Stanley and Risto Miikkulainen. Competitive coevolution through evolutionary complexification. *Journal of Artificial Intelligence Research*, 21:63–100, 2004.

[58] Sergios Theodoridis and Konstantinos Koutroumbas. *Pattern Recognition*. Academic Press, 1999.

[59] Dirk Thierens. Non-redundant genetic coding of neural networks. In *Proceedings of the 1996 IEEE International Conference on Evolutionary Computation*, Nagoya, Japan, May 1996. IEEE.

[60] Aurelio Uncini, Francesco Gobbi, and Fracesco Piazza. Frequency recovery of narrow-band speech using adaptive spline neural networks. In *Proceedings of the International Conference on Acoustics, Speech and Signal Processing*, volume 2, pages 997–1000, Phoenix, AR, March 1999. IEEE.

[61] Hans-Michael Voigt, Heinz Mühlenbein, and Dragan Cvetković. Fuzzy recombination for the breeder genetic algorithm. In Larry Eshelman, editor, *Proceedings of the 6th International Conference on Genetic Algorithms*, pages 104–111, San Francisco, CA, 1995. Morgan Kaufmann.

[62] Darrel Whitley, Soraya Rana, and Robert B. Heckendorn. Island model genetic algorithms and linearly separable problems. In *Proceedings of the AISB Workshop on Evolutionary Computation*. AISB, 1997.

[63] Xin Yao. Evolving artificial neural networks. *Proceedings of the IEEE*, 87(9):1423–1447, September 1999.

[64] Hiroshi Yasukawa. Quality enhancement of band limited speech by filtering and multirate techniques. In *Proceedings of the International Conference on Spoken Language Processing*, pages 1607–1610, Yokohama, Japan, March 1994. IEEE.

[65] Hiroshi Yasukawa. Signal restoration of broad band speech using nonlinear processing. In *Proceedings of the European Signal Processing Conference*, pages 987–990, Trieste, Italy, September 1996. IEEE.

# Appendix A

# The Listening Test Instructions

The following listening test instruction text was given to the user before starting the test. The subject could refer the instructions at any point of the test. The instructions are in Finnish. They briefly explain the parts of the test, what the subject is supposed to do in each part, the use of the user interface and the scale used in the CCR test.

## Ohjeita kuuntelukokeeseen

Tässä kokeessa on kaksi osaa, joissa molemmissa kuunnellaan lausepareja. Osien välillä pidetään lyhyt virkistystauko.

Ensimmäisessä osassa tehtävänäsi on valita ääninäytteistä **se, jota kuuntelisit mieluummin (*A* tai B)**. Mikäli näytteet kuulostavat mielestäsi niin samanlaisilta, ettet osaa niistä miellyttävämpää, valitse ***melkein sama***.

Toisessa osassa vertaat kahta ääninäytettä ja kerrot, **onko näytteen *B* laatu näytteeseen *A* verrattuna mielestäsi**:

3 *paljon parempi*
2 *parempi*
1 *vähän parempi*
0 *melkein sama*
-1 *vähän huonompi*
-2 *huonompi*
-3 *paljon huonompi*

Ennen kumpaakin osaa on muutaman lauseparin mittainen harjoitusjakso, jonka aikana pääset harjoittelemaan vastaamista. Harjoittelujakson aikana sinulla on mahdollisuus säätää äänenvoimakkuus haluamallesi tasolle näytöllä olevaa äänensäädintä käyttäen, sekä tarvittaessa esittää testiohjelman käyttöön liittyviä kysymyksiä.

Testissä voit kuunnella lauseparin ääniä valitsemassasi järjestyksessä haluamasi määrän kertoja. Kun olet kuunnellut molemmat näytteet vähintään kerran, aktivoituu vastausosa ja voit valita vastauksesi. Kun olet antanut mielipiteesi, voit siirtyä seuraavaan näytepariin painamalla *done* nappia.

Pyydämme, ettet keskustele testiin liittyvistä asioista muiden kokeeseen osallistuvien kanssa ennen kuin hekin ovat tehneet testin.

Kiitos avustasi!

# Appendix B

# Comments on the Listening Test

Comments the subjects made after the listening test. Comments by different subjects have been separated by a horizontal line. Additions and clarifications made by the author are in parenthesis.

Comments were written down by the author during a short interview after the test. Not all comments were written down, typically emphasis was put on the comments that had not been said before. Two of the listeners had no comments to make.

## Original Comments in Finnish

- Suurimmassa osassa ääniä korkeampi oli selkeämpi.

---

- Kun äänet tulivat toisen kerran, kuuntelutilanne oli tavallaan eri, koska äänet olivat edeltävien äänten takia erilaisessa kontekstissa ja tuli ehkä valittua eri tavalla.
- Jos äänet kuitenkin erilaiset, vaikea valita *melkein sama*.

---

- Junasta puhuva naisääni jotenkin “ärsytti” minua eniten. Ei siis mitenkään niin, että olisin jotenkin inhonnut kyseistä näytettä/puhujaa - se vain osui eniten korvaan. Muut puhujat puhuivat mielestäni ihan normaalisti/neutraalisti, mutta tässä näytteessä puhuja painotti sitä viimeistä sanaa jotenkin hassusti/epätavallisesti. Tuntui siltä että puhuja puhui suomea “väärin”, kun painotti viimeisen sanan viimeistä tavua kun painotuksen olisi pitänyt olla sanan alussa. (Muillakin oli samantyylisiä kommentteja, mutta koska puhetyyli ei varsinaisesti liity kaistanlaajennukseen, on ne jätetty pois.)
- Yleensä /s/:t ja ehkä /r/:tkin tuntuivat aiheuttavan eniten eroa näytteiden välillä - siis että yleensä toinen oli selkeästi parempi kuin toinen kun tuli sellaisia näytteitä joissa näitä kirjaimia oli. Esim. se oluesta jotain puhuva näyte ei eronnut A:n ja B:n välillä koskaan niin selkeästi kun siinä ei ollut paljon /s/:ää tai /r/:ää.

- Aika usein toinen näyte oli jotenkin yleisesti epäselvempi - siis ei niin että se olisi ollut hiljaisempi, vaimeampi tai vaikeampi kuulla, vaan niin että ihan kuin puhuja olisi puhunut nenäliinan tms. takaa, eli äänteet olivat kaikki yhtälailla epätarkkoja ja epäteräviä.

---

- Osa näytteistä matalia ääniä, jolloin vaikea kuulla eroja matalilla taajuuksilla.

---

- Ärsytti hirveästi, jos “korjaus” pahensi /s/:n ässävikaa.
- Diskanttien korostaminen paransi selkeyttä.
- Tynnyrimäisyys puurouttaa (äänen) nopeasti, vaikea saada selvää.
- Jos diskanttia liikaa, särkee /s/:t, mutta edelleen saa selvää. (vastakohtana edelliseen)
- Joissain tapauksissa (kaistanlaajennus) jopa sattuu korvaan. Viesti tulee perille, mutta ääni on “kova”. Seurauksena kokemus ei ole miellyttävä, mutta ääni on selkeä.
- Pahin kombinaatio on matala ääni ja tylpistetty (tylpistys viittaa kapeakaistaisiin ääniin)

---

- Toiset äänet olivat aina vähän kovempia korvaan
- (Keinotekoinen kaistanlaajennus) vähän metallinen, mutta ei minua haittaa vaan kuuntelen mieluummin
- Tunkkaiset äänet ärsyttävät, mieluummin metallinen kuin tunkkainen. Metallinen on selkeämpi. (Tunkkaisuus viittaa tässä kapeakaistaisiin ääniin.)
- 1,2,3 arvostelu vaikea. (Mistä tietää) minkä arvoinen on kolmonen? Ongelmana on se, että kun asteikko alkaa selkiytyä, onkin jo antanut esim. kakkosen sille kaikkein huonoimmalle.
- Kumulatiivisesti alkoi ärsyttämään samat piirteet. Siksi myöhemmin tuli ehkä arvosteltua ankarammin.
- Miesäänille metallisuus (kaistanlaajennus) sopii, naisäänille ei. (Metallisuus viittaa kaistanlaajennuksessa esiintyvään ylätaajuuksien ylikorostumiseen, jota esiintyy joissain laajennetuissa /s/ äänteissä.)

---

- Useinmiten se, jossa äänenlaatu tuntui olevan parempi, tuntui olevan “kovempi”. Pehmeät olivat tavallaan miellyttävämpiä, mutta epäselvempiä.

---

- Aika moni sanoo /s/:n ässävikaisesti, kuin /f/:n.
- “Kovemmat” selkeämpiä, mutta kovuus ei miellyttävää. (Testihenkilö oli kuitenkin käyttänyt selkeyttä laatukriteerinä ja pisteyttänyt siksi kaistanlaajennetut paremmin.)

---

- /s/:stä helppo huomata äänen laatu.
- *Melko sama* unohtui matkalla. (Testihenkilö ei käyttänyt melko samaa kertaakaan paritestissä, edes null-pareille.)

---

- Alkaa hakea “liikaa” eroja.
- Paremmat kirkkaampia, ei puuromaisia.

---

- Toiset äänet kuulostavat äänenvoimakkuudeltaan kovemmilta kuin toiset. (Tämä kommentti tuli useammaltakin kuuntelijalta.)

## Comments Translated to English

- In most of the samples, higher pitched sound was clearer.

---

- When the same test item came second time, the listening context was different and therefore I sometimes chose differently.
- If there was a clear difference in samples, it was hard to choose *almost the same*.

---

- The female voice talking about the train annoyed me the most. Not that I hated the sample or the speaker, it just sounded bäd. In my opinion, the other speakers were speaking quite normally/neutrally, but in this sample the speaker stressed the last word somehow weirdly. It felt like the speaker was pronouncing Finnish in the wrong way, because she was stressing the last syllable when she should have been stressing the beginning of the word. (Other subjects also had comments dealing with the style of speech speaker in some particular sample had, but as the speech style has no great relevance to the artificial bandwidth expansion, other such comments have been left out.)
- Usually /s/’s and maybe /r/’s seemed to cause the most difference between samples. One of the samples in the pair was clearly better in pairs which had these kinds of phonemes. For example in the pair where the speaker said something about beer (c8) there was no clear difference between the samples because it had no /s/ or /r/.
- Quite often one of the samples was somehow more unclear. Not that it was more silent or harder to hear, it was more like the speaker was speaking with a handkerchief etc. in front of his face. The phonemes were all equally muffled and unclear.

---

- Some of the test samples were low pitch voices. In them it was difficult to hear differences in the lower frequencies.

---

- It was extremely annoying, if “the fix” worsened the lisp in /s/.
- Enhancing discants improved clarity.
- Barrel likeness makes it (the speech) muddy fast, it is hard to make out what the speaker is saying.
- If there is too much discant, it breaks the /s/, but it is still easy to make out what the speaker is saying. (as opposed to the last item)

- In some cases it (artificial bandwidth expansion) even hurts the ear. The message is clear, but the voice is “hard”. As a consequence the experience is not pleasant, but the speech is clear.
- The worst combination is low pitched voice and bluntness (bluntness is referring to narrowband sounds)

---

- Some of the samples were always a bit “harder” (sharper) for the ear.
- (Artificial bandwidth expansion is) a bit metallic, but I don’t mind. I rather listen to it.
- Muffled sounds are annoying, I prefer metallic to muffled. Metallic is more clear. (Muffled refers to narrowband sounds here.)
- Grading on a scale of 1–3 is hard. How do you know which sound should be given a three? The problem is that when you start to have a feel what kind of scale we are on, you have already given a two for the worst sound in the test.
- Same sound features started to annoy cumulatively more and more. Therefore later samples may have gotten worse grades.
- Metallic sound (artificial bandwidth expansion) was good for male voices, but not for female voices. (The metallic sound refers to the overamplification of high frequencies, which is sometimes encountered in /s/ of the artificially expanded samples.)

---

- Most of the time the sample in which quality seemed to be higher felt “harder”. The softer samples were in a way more pleasing, but unclearer.

---

- Quite many of the (test sample) speakers say /s/ with a lisp, like an /f/.
- “Harder” sounds were more clear, but the hardness was not pleasing. (The test subject had nevertheless defined clarity to be an important part of quality and therefore given higher grades for the expanded samples.)

---

- It is easy to determine the quality of the sound from /s/’s.
- I totally forgot about the *about the same*, while doing the test. (The test subject never used *about the same* in the pair comparison test, not even for null pairs.)

---

- One tries to find differences “too hard”.
- The better samples were more clear, not muffled.

---

- Some sounds sounded louder than the others. (This comment was given by multiple listeners.)