Speech Recognition in Adverse Environments: a Probabilistic Approach

by

Trausti Thor Kristjansson

A thesis
presented to the University of Waterloo
in fulfilment of the
thesis requirement for the degree of
Doctor of Philosophy
in
Computer Science

Waterloo, Ontario, Canada, 2002

Abstract

In this thesis I advocate a probabilistic view of robust speech recognition. I discuss the classification of distorted features using an optimal classifier, and I show how the generation of noisy speech can be represented as a generative graphical probability model. By doing so, my aim is to build a conceptual framework that provides a unified understanding of robust speech recognition, and to some extent bridges the gap between a purely signal processing viewpoint and the pattern classification or decoding viewpoint.

The most tangible contribution of this thesis is the introduction of the Algonquin method for robust speech recognition. It exemplifies the probabilistic method and encompasses a number of novel ideas. For example, it uses a probability distribution to describe the relationship between clean speech, noise, channel and the resultant noisy speech. It employs a variational approach to find an approximation to the joint posterior distribution which can be used for the purpose of restoring the distorted observations. It also allows us to estimate the parameters of the environment using a Generalized EM method.

Another important contribution of this thesis is a new paradigm for robust speech recognition, which we call *uncertainty decoding*. This new paradigm follows naturally from the standard way of performing inference in the graphical probability model that describes noisy speech generation.

Acknowledgements

First and foremost I would like thank my advisor Brendan Frey. One of the most enjoyable aspect of the last four years has been to discuss research ideas with him. His insight into complex theoretical and technical issues never ceases to impress me.

I would like to thank Li Deng and Alex Acero who made my visits at Microsoft Research exciting and productive. I benefited greatly from their wisdom and guidance over the last two years.

I am also indebted to my thesis committee, Pedro Moreno, Dale Schuurmans, Richard Mann and Paul Fieguth, who provided me with valuable suggestions during the last stages of my thesis work.

My stay in Waterloo over the last three years was all the more fun due to my friends who always had time for an interesting discussion over a cup of coffee. I am especially lucky to count Chris Pal and Sarah Dorner among my friends.

I thank my family for their love and support through the years. My mother Jona, sister Berglind and brother Arnar for giving me a home in Iceland to return to and my father Kristjan for action filled trips to Florida.

This thesis is dedicated to Susan, for her love, support and patience.

Contents

List of Tables

List of Figures

Chapter 1

Introduction and Overview

In recent years, automatic speech recognition has started to emerge as a practical and useful technology. Applications range from large vocabulary medical dictation, and spoken document retrieval to embedded applications such voice activated dialing or song selection in a portable MP3 player.

In any speech based Human Computer Interaction (HCI) system, there will be a need for verification that the system has understood the intensions of the speaker. This process is called *grounding*[66]. People are very demanding on the overall naturalness and effectiveness of this process. Most people will not use a dictation system if it misses two words in a hundred. A user of a voice activated dialing service is likely to revert to manual dialing if the system once calls a wrong number.

Speech recognition systems are very sensitive to a mismatch between the conditions under which the system was trained and the deployment conditions[46]. Recognition performance can degrade substantially, even for small differences. As a result, most successful applications of speech recognition require the user to employ a close talking microphone or a telephone handset, since they ensure relatively high signal quality. In some environments where speech based HCI would be convenient, it is not possible to use close talking microphones. By increasing the immunity or *robustness* of the speech recognition system to environmental interference, many new applications become viable.

Although there are many contributing factors to the usefulness of a speech based HCI system, a crucial factor is the accuracy of the speech recognition component. In this thesis I look at speech recognition component in isolation from other aspects of the HCI system.

1.1 Thesis Overview

This thesis can be divided roughly into two parts. In the first part I present a general probabilistic view of the problem of classification of corrupted observations, and then formulate a graphical probabilistic model for of speech recognition in adverse environments. In the second part I present a new method for robust speech recognition called Algonquin that takes advantage of the probabilistic viewpoint. This method is studied extensively in the second part of the thesis. The subject of the individual chapters is as follows:

Chapter 2 provides a brief review of the speech recognition and the problem of speech recognition in a noisy environment. We introduce the *environment model* and discuss how the environment effects and distorts the observations.

In Chapter 3 we look at the generic problem of pattern classification when observations have been distorted. We discuss some strategies for handling distorted observations, and relate them to each other. We also relate these strategies to speech recognition of noisy speech, which is essentially a complicated version of the simple problem discussed in this chapter.

In Chapter 4 we introduce the generative graphical model that describes an HMM based speech recognizer, and expand this model to account for the generation of noisy speech. We discuss inference in this generative model, and how traditional robustness paradigms can be viewed as inference in the model. We also see that this view leads to a new paradigm called *uncertainty decoding*.

In Chapter 5 we start filling in components of the generative model discussed in Chapter 4. We discuss the Mel-Frequency Cepstrum Transform of the environment model, and how it leads to the *interaction likelihood* that describes the relationship between clean speech, noise, channel and the observed noisy speech.

In Chapter 6 the remaining components of the noisy speech model are explained, and we discuss how the *interaction likelihood* causes exact inference to be intractable.

In Chapter 7 we first discuss how recognition performance is assessed, and review some prior methods for Robust speech recognition. In particular, we discuss the VTS method.

In Chapter 8 we introduce a new method for robust speech recognition, called Algonquin. This method solves the problem of the intractability of inference, by approximating the interaction likelihood with an approximate likelihood function that is computationally attractive. We look at various aspects of the Algonquin method and discuss its advantages.

In Chapter 9 we continue to discuss the Algonquin method. In this chapter, we discuss the problem of estimating the environmental parameters, and they can be estimated using a Generalized EM method.

In Chapter 10 we discuss a new decoding strategy for robust speech recognition called *uncertainty decoding* or the *soft information* paradigm. This method is inspired by the realization of how inference should be done properly which arose in the discussion of

Chapters 3 and 4. We give results that show the promise of this new paradigm.

We conclude in Chapter 11 with a review and discussion of future directions.

Chapter 2

Speech Recognition in Adverse Environments

This chapter introduces the basic workings of a speech recognition system. We also introduce the standard environment model and discuss the effect noise and channel distortion has on the speech signal and the related observation features.

2.1 Introduction to Automatic Speech Recognition

The purpose of a speech recognition system is to convert a sequence of real valued observations $\mathbf{X} = \{\mathbf{x}_1, \dots, \mathbf{x}_T\}$ derived from the speech waveform, into a sequence of words $\mathbf{W}$ [68]. Each word corresponds unambiguously to a sequence of discrete states $\mathbf{s}$. We can therefore work with states rather than words.

In order to describe the correspondence of states to observations, we use a probability model $P(\mathbf{X}, \mathbf{s})$. Given a sequence of observations, this function is designed such that it is maximized for the state sequence corresponding to the words uttered.

Once the model $P(\mathbf{X}, \mathbf{s})$ has been constructed, the purpose of a speech recognizer is to find the state sequence that maximizes the function

$$\hat{\mathbf{s}} = \underset{\mathbf{s}}{\mathrm{argmax}} P(\mathbf{X}, \mathbf{s}). \quad (2.1)$$

There are in essence only two components of the correspondence model $P(\mathbf{X}, \mathbf{s})$; the phone, word and language model gives *transition probabilities* $p(s_i|s_{i-1})$, that assign values to the probability that one state follows another. The *acoustic models* $p(\mathbf{x}|s)$ reflect the probability that the vector valued observation $\mathbf{x}$ resulted from the state s.

From these two components we can write the speech model $P(\mathbf{X}, \mathbf{s})$,

$$P(\mathbf{X}, \mathbf{s}) = p(s_0) \prod_{i=1}^{T} p(\mathbf{x}_i|s_i) p(s_i|s_{i-1}), \quad (2.2)$$

where i indexes the time frame of the observation and state[68].

We can also view a speech recognizer from a systems perspective. Figure 2.1 shows a block diagram of a conventional speech recognition system.

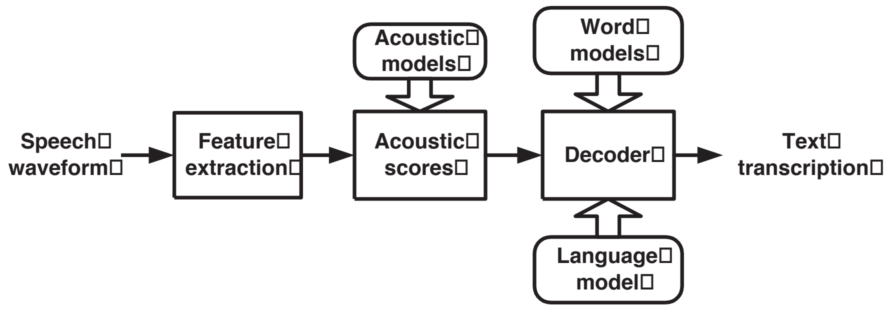


Figure 2.1: Block diagram of conventional speech recognition system.

The acoustic speech waveform is transformed into an analogue electronic signal by a microphone which is then converted to digital form by an Analog-to-Digital Converter

(ADC). Samples of the signal are taken at a fixed rate, commonly 8-16 thousand samples per second.

In the **Feature extraction** stage (also called the **Front end**), the sampled waveform is taken and converted into suitable features $\mathbf{x}$. The most common features are Mel-frequency Cepstrum Features. These will be discussed in detail in Chapter 5.

In the next block, labelled **Acoustic scores** the observed feature is compared to finite number of speech sounds. The speech sounds are represented by *acoustic models* $p(\mathbf{x}|s)$ where s designates the *state* and corresponds to a particular speech sound. There are a few ways of modelling $p(\mathbf{x}|s)$, however, the most common and effective model is the mixture of Gaussians model,

$$p(\mathbf{x}_i|s=k) = \sum_j \rho_j N(\mathbf{x}_i; \mu_{j,k}, \mathbf{\Sigma}_{j,k}), \tag{2.3}$$

where $\mathbf{x}_i$ is an observation vector at time i, and s_k designates a specific speech sound. In this thesis we will only consider Gaussian Mixture acoustic models.

The decoder finds the most probable sequence of states “that generated” the observed signal. It uses *word* and *phone* models as well as a language model that define allowable word and phone sequences. The effect of using word and phone models is to reduce the number of allowable state sequences that the recognizer has to search over.

2.2 Effects of the Environment

In this section we will introduce the effects of the environment. The traditional signal processing environment model is shown in Figure 2.2 [1].

This model shows the speech signal being convolved with a linear channel, and noise being added. The linear channel models the effect of the microphone and the room acous-

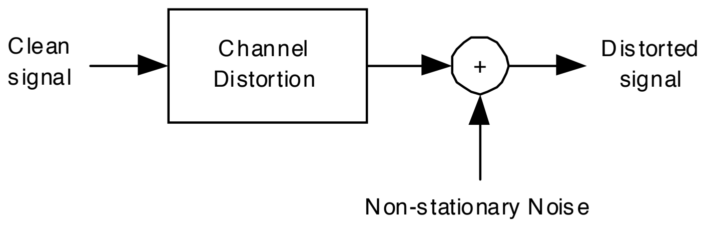


Figure 2.2: A block diagram of the noise and channel.

tics. Ambient noise is modelled as an additive process. Examples of noise processes are ventilation noise in an office, road noise in a car and the sound of a subway train.

In the time domain this can be written as[1]

$$y[m] = \sum_{k=0}^{K-1} x[m-k]h[k] + n[m], \tag{2.4}$$

where $x[m]$ is a sample of the clean speech signal, $n[m]$ is a sample of the noise signal, $h[m]$ is the impulse response of the linear channel, $y[m]$ is a sample of the noisy speech signal.

A linear time invariant channel model is used to model the effects of room, the effects of the microphone, and in the case of speech that is sent over a telecommunications channel, the effects of the channel. Collectively, these are called the channel. In most cases, physical characteristics of the channel vary slowly, and thus parameters of the channel model vary slowly.

The observed signal goes through a sequence of operations in the **Feature extraction** block, which will be discussed extensively in Chapter 5. In this block, short segments of the time domain signal are processed to produce features. The result is a set of features in the log-spectrum. In the log-spectrum domain, the relationship becomes

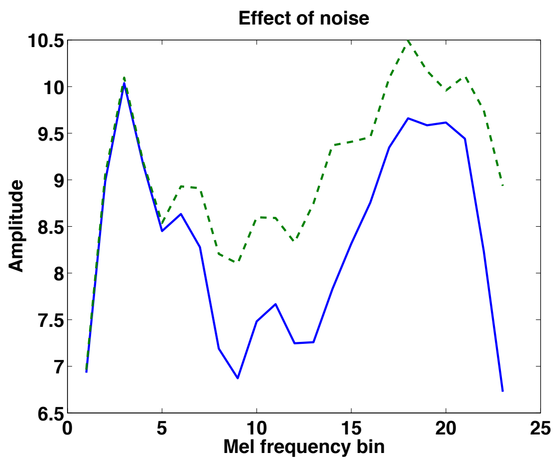
$$\mathbf{y} = \mathbf{x} + \mathbf{h} + \ln(1 + \exp(\mathbf{n} - \mathbf{x} - \mathbf{h})) + \mathbf{e}, \tag{2.5}$$

where $\mathbf{e}$ is an error term, and $\mathbf{y}$ is the log-spectrum representation of a portion of the time domain signal y etc.

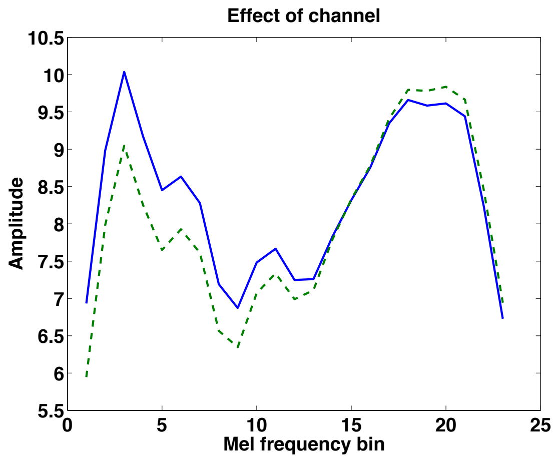
The channel and additive noise alter the log-spectrum coefficients in predictable ways. In the log-spectrum domain, the effect of the channel is mostly linear, but the effect of the noise is highly non-linear.

The effect of adding subway noise at 10dB below the signal level is shown figure 2.3(a). As can be seen, the lower components are intact while the higher components have been masked by the noise. Figure 2.3(b) shows a noise free case where the signal has been filtered to simulate the effect of a microphone with MIRS[1] frequency characteristics. Notice how the lower frequencies are attenuated, while higher frequencies are amplified.

[1]MIRS is a transfer function characteristic for telecommunications terminals as defined in the International Telecommunications Union (ITU) technical specification GSM 03.50



(a)



(b)

Figure 2.3: Plot of log-mel-spectra for a single frame. (a) Effect of noise. Dotted line shows the noisy acoustic vector. Notice how the noise masks the signal for higher frequencies. (b) Effect of using a different microphone. Dotted line shows the acoustic vector of alternate microphone. Notice how the lower frequencies are attenuated, while higher frequencies are amplified.

Chapter 3

Classifying Corrupted Observations

In this chapter we start by describing the effects of noise on a very simple pattern classifier and discuss several measures that can be used to counter the effects of noise. These include cleaning the signal by only taking bias into account, cleaning by also taking a speech prior into account and changing the classifier to classify noisy observations. We will then discuss the probability of error and find the relative effectiveness of these measures.

The problem of noise robust speech recognition is a complicated version of the simple classifier discussed here and the discussion will therefore provide a framework that various robustness methods can be related to.

3.1 Classification of Corrupted Observations

The simplest possible real value classifier takes one real valued observation and assigns it to one of two classes. Figure 3.1(a) shows the Bayesian graph[44] for this classifier. The model involves a discrete class variable s, and a continuous variable x. It is assumed that the variability in the observation is inherent in the process of generation, something that cannot be helped. As an example, we assume the two classes are fricatives and vowels $s \in \{f, v\}$, and that we have two corresponding continuous class conditional distribution for the observation x, i.e. $p(x|s = f)$ and $p(x|s = v)$ which happen to be Gaussian. This classifier will assign an observation x to the class of fricatives, if $p(x|s = f) \cdot p(s = f) > p(x|s = v) \cdot p(s = v)$ and vice versa[15].

Figure 3.1: (a) Classifier for clean observations: Graph shows real valued classifier with one discrete variable s and one continuous observed variable x. (b) Classifier for corrupted observations: Graph with one discrete variable s one continuous hidden variable x and a continuous noisy observed variable y.

If the observation has been distorted in some way, we will not observe the clean signal x, instead, we observe the noisy signal y. Figure 3.1(b) shows a Bayesian network with an additional node for the noisy feature y. Notice that x is now a hidden variable, and we assume that the distortion is independent of the class variable.

3.2 Classifying Corrupted Observations

We will now discuss several measures we can take when we know that the clean observations have been corrupted.

3.2.1 Correcting for Bias.

In the spectrum domain, the effect of noise is to introduce both bias and uncertainty. If the "noise" process introduces only bias (which is almost never the case), then we can alter the decision boundaries of our classifier and the "noise" process will not effect performance. This situation is shown in Figure 3.2. Here, the relationship between x and y is $y = x + 1$. Notice that in Figure 3.2, the area of the shaded error region has not increased from the top plot to the shifted bottom plot. Some robustness methods, such as Spectral Subtraction only take the bias component into account.

The noise process also introduces uncertainty. In this thesis, uncertainty will be represented by the variance of the distributions we use. If for example the relationship between x and y is $y = x + e$ where e is a zero mean and normally distributed, then the minimum classification error will necessarily increase, due to the increased overlap of the class-conditional distributions. This is shown in the middle plot of Figure 3.3.

3.2.2 Maximum Likelihood Point Estimates: Using $p(y|x)$

As mentioned above, many robustness methods use point estimates of the random variables involved[33]. One very common method involves finding a point estimate of the clean observation, based on the noisy observation. To do this, we can for example use

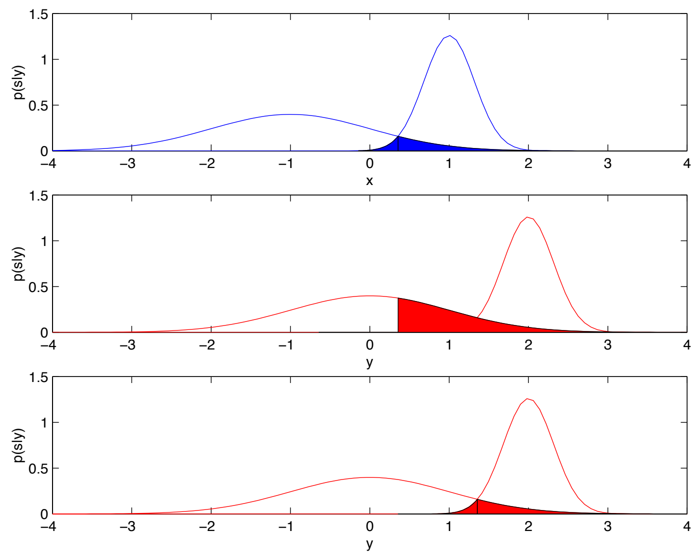


Figure 3.2: Effect of bias: (Top) The class conditional distributions for clean observations. (Middle) Effect of bias (but no increase in uncertainty). Notice that the error region has been enlarged. (Bottom) After the correcting for bias (e.g. using Maximum-Likelihood parameter estimation).

the Minimum Mean Squared Error estimate[17]

$$\hat{x} = \int x \cdot p(x|y) dx \tag{3.1}$$

and then use the clean speech classifier.

Returning to our example of a normally distributed noise distribution with zero mean, we see that $y \sim N(y; x, \sigma)$. If we use the Maximum-Likelihood strategy for parameter estimation, and assume a uniform distribution for $p(x)$ we get

$$\hat{x} = \int x \cdot \frac{p(y|x)p(x)}{p(y)} dx = \int x \cdot N(y; x, \sigma) dx = y. \tag{3.2}$$

In this case we have used all the information about how the environment distorts clean speech, but no prior information about clean speech x. As can be seen from Equation

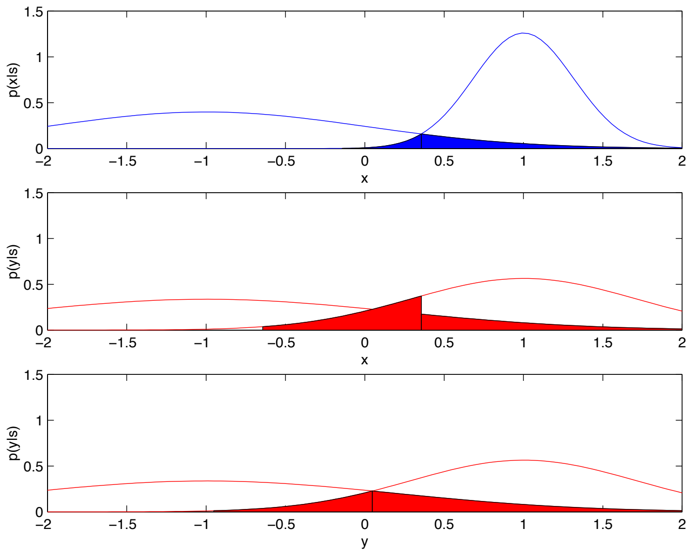


Figure 3.3: Effect of uncertainty: (Top) The class conditional distributions for clean observations. (Middle) Effect of increasing variance (but no bias). (Bottom) After finding the optimum decision boundary (i.e. classification based on $p(y|s)$).

(3.2) $\hat{x} = y$, and hence, we will not move the decision boundary. Thus, if we use this point estimate, we will introduce an additional error, as seen in the bottom plot of Figure 3.3. This is because the decision boundaries will shift, even thought the noise process introduces no bias.

3.2.3 Maximum A Posteriori Point Estimates: Using $p(y|x)$ and $p(x)$

In Equation (3.2) we did not us prior information about x. If we use the prior model for x we arrive at a different expression for $\hat{x}$,

$$\begin{aligned}
\hat{x} &= \int x p(x|y) dx = \int x \cdot \frac{p(y|x)p(x)}{p(y)} dx \qquad (3.3)\\
&= \int x \frac{N(y; x, \sigma_y) \sum_s N(x; \mu_s, \sigma_s) P(s) dx}{\sum_s \int N(y; x, \sigma_y) N(x; \mu_s, \sigma_s) P(s) dx}\\
&= \frac{p(v) N(y, \mu_v, \sigma_v + \sigma_y) \left[\frac{\sigma_v y + \sigma_y \mu_v}{\sigma_v + \sigma_y}\right] + p(f) N(y, \mu_f, \sigma_f + \sigma_y) \left[\frac{\sigma_f y + \sigma_y \mu_f}{\sigma_f + \sigma_y}\right]}{p(v) N(y; \mu_v, \sigma_v + \sigma_y) + p(f) N(y; \mu_f, \sigma_f + \sigma_y)}.
\end{aligned}$$

As we will see in the next section, using the prior model when producing a point estimate of x, is better than assuming a uniform prior on x. MMSE-VTS and MMSE-Algonquin make use of a speech model.

3.2.4 Classification Based on $p(y|s)$

Since the objective of the above methods is to restore the observations to their uncorrupted state and return a point estimate, there is always the chance of error, and the information about uncertainty is discarded. To take uncertainty into account, we should base recognition on $p(y|s)$,

$$p(y|s) = \int p(y|x, s) p(x|s) dx. \qquad (3.4)$$

This turns out to be the best we can do, i.e. to base our classification on $p(y|s)$ instead of $p(x|s)$. Although this is the optimal classification strategy from the perspective of recognition accuracy, it can be more computationally demanding.

By retraining, one can find the models $p(y|s)$ and substitute them for $p(x|s)$ into the classifier in Figure 3.1(a). *Model adaptation* techniques approximate this procedure. In other words, for each s we first find an expression $f(y) = p(y|s)$ as a function y and then evaluate $f(y_{obs})$.

In Chapter 10 we will introduce a new paradigm that we call the *soft information* or *uncertainty decoding* paradigm. In this paradigm we calculate $p(y_{obs}|s)$ in a different way. In essence, we first approximate $p(y_{obs}|x)$ for a particular observation y_{obs} and then evaluate the integral in Equation (3.4).

3.3 The Probability of Error

Robustness methods fall into two main categories, i.e. *feature domain* methods and *model domain* methods [38]. Feature domain methods attempt to clean the signal and produce a point estimate of the clean signal $\hat{x}$ given the noisy observation y.

Recall the example of a vowel/fricative classifier. Now we wish to find the probability of error of this classifier. First we look at the case where there is no distortion. The probability of error $\mathcal{E}(p_x, x)$[15] is the probability that we choose a vowel, when in fact the observation was caused by a fricative, and vice versa

$$\mathcal{E}(p_x, x) = p(v) \int_{x \in F} p(x|s = v)dx + p(f) \int_{x \in V} p(x|s = f)dx. \tag{3.5}$$

where V is the set of x values where $p(x|s = v) > p(x|s = f)$ and F is the set of x values where $p(x|s = f) < p(x|s = v)$. The argument p_x in $\mathcal{E}(p_x, x)$ represents the decision boundaries with respect to $p_x(x|s)$ and the argument x means that the input to the classifier is the clean observation x.

The graph in Figure 3.1(b) shows how we update the model if we have a noisy observation y. In the previous section we described two methods for producing a point estimate of clean speech $\hat{x}$ and using that point estimate in the clean speech classifier. In this case the probability of error is

$$\mathcal{E}(p_x, \hat{x}) = p(v) \int_{y:\hat{x} \in F} p(f(y)|s = v)dy + p(f) \int_{y:\hat{x} \in V} p(f(y)|s = f)dy, \qquad (3.6)$$

where $\hat{x} = f(y)$.

The last method we discussed involved basing the classification on $p(y|s)$. Assuming we know the exact form of $p(y|x)$ the probability of error is now

$$\begin{aligned} \mathcal{E}(p_y, y) &= p(v) \int_{y \in F} p(y|s = v)dy + p(f) \int_{y \in V} p(y|s = f)dy & (3.7) \\ &= p(v) \int_{y \in F} \int_x p(y|x)p(x|s = v)dxdy & (3.8) \\ &+ p(f) \int_{y \in V} \int_x p(y|x)p(x|s = f)dxdy. \end{aligned}$$

One may be led to think that is possible to do better by cleaning the observation and passing the cleaned observation $\hat{x}$ to the recognizer. If no additional information is used in the cleaning process, then it follows from the *data processing inequality* [10] that it is impossible to do better by processing the signal, e.g. by cleaning it. Thus we have

$$\mathcal{E}(p_y, y) \leq \mathcal{E}(p_x, \hat{x}). \qquad (3.9)$$

We also know that using clean speech will produce better results than either of the above methods, thus

$$\mathcal{E}(p_x, x) < \mathcal{E}(p_y, y) \leq \mathcal{E}(p_x, \hat{x}). \qquad (3.10)$$

3.3.1 A Numerical Example

To illustrate the relationship between these methods and their relative error rates, consider the fricative/vowel classifier again. Assume that vowels and fricatives are equally likely. Assume also that $p(x|s = v) = N(x; -1, 0.1)$ i.e. normally distributed with mean

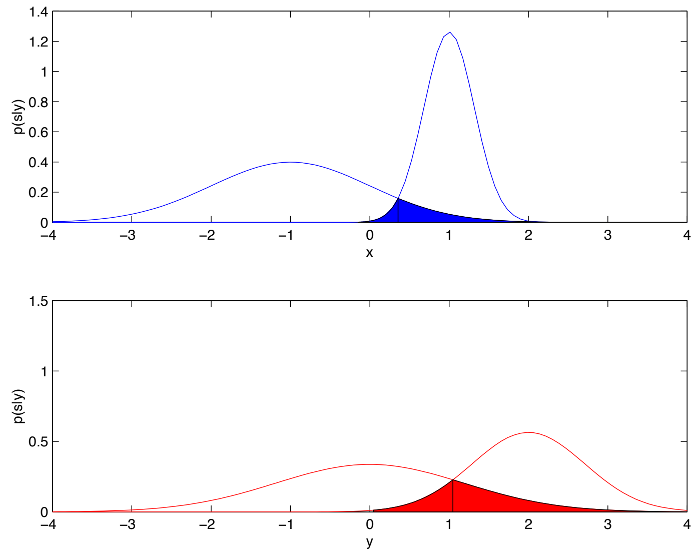


Figure 3.4: (Top) The class conditional distributions for clean observations. (bottom) Effect of bias and uncertainty $p(y|x) = N(y; x + 1, 0.4)$.

-1 and variance 0.1, and similarly, $p(x|s = f) = N(x; 1, 1)$. The probability of error under these conditions is 0.0539 and the expected classification error is therefore 5.39%.

Now we corrupt the data using $p(y|x) = N(y; x+1, 0.4)$, i.e. we shift the observation by 1 and add a little Gaussian noise (see Figure 3.4). For such a simple model, it is possible to evaluate the error integrals exactly. However, it is much simpler to use a Monte-Carlo type method[65]. If we do this, we find that the probability of error when using the noisy features without any processing is 59.1%.

If we "clean" the observation using $\hat{x} = y - 1$ then the error is reduced to 27.4%. If we use the complex expression for $\hat{x}$ of Equation (3.3) that takes the prior information about the speech models into account, the error is reduced to 24.2%. If we base the classification on $p(y|s)$ the classification error is reduced to 20.9% which is the theoretically minimum classification error for this fictional vowel/fricative classifier.

3.4 Discussion

In this chapter we have introduced three strategies for dealing with noise in the simplest of all real valued classifiers. The first is to do nothing. We looked at two possible ways to produce a point estimate of clean speech $\hat{x}$. The difference between these was the use of a prior model for speech. The last strategy discussed was to use $p(y|s)$ instead of $p(x|s)$. This turns out to be the best strategy.

Although this discussion is a simplification of the problem of noise robust speech recognition, different Robust Automatic Speech Recognition (RASR) methods can be classified based on which of these strategies they take. The problem of speech recognition in noise is essentially a complicated version of the preceding discussion. The complications arise from the non-linear way in which the environment effects the speech features, due to the Mel-frequency Cepstrum (MFC) transform, and the more complicated generative model required.

In this thesis, we will introduce new methods for robust speech recognition that allow us to perform approximate inference despite the non-linearity of the MFC transform, and explore the importance of taking uncertainty of various components of the model into account. We will also explore the importance of using prior information of the speech model.

In the discussion above, the way in which the environment effects the speech signal was assumed to be known (i.e. the parameters of $p(y|x)$). This is not the case in real world situations, and ways in which to estimate the environment model is another topic of discussion in this thesis.

Chapter 4

Graphical Probability Models for Robust ASR

In this chapter we discuss a graphical model representation of the corruption process. We will introduce a generative model for noisy speech. We then discuss the different paradigms for robust speech recognition from this perspective.

4.1 A Generative Graphical Model for Noisy Speech

Figure 4.1 shows a generative Bayes net[41, 54, 15] for the speech process. The graph shows the state sequence of the speech process $\mathbf{s}^x = s_1^x \dots s_M^x$ and the class dependent observation vectors $\mathbf{X} = \mathbf{x}_1 \dots \mathbf{x}_M$. This graph is equivalent to the traditional Hidden Markov Model with continuous observations[82].

To model the effects of environmental noise and channel distortion, we introduce two new random variables, $\mathbf{n}$ and $\mathbf{h}$. We will model these random variables by mixtures of Gaussians (see Equation 2.3) similarly to speech. Hence, we use the variables s_i^n and s_i^h

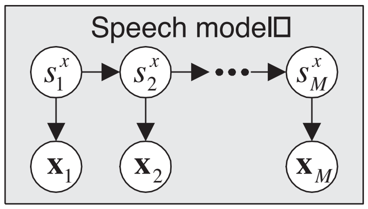


Figure 4.1: A Bayesian network of the speech generation process. The model is equivalent to the traditional speech HMM with continuous observation densities.

to denote the class of $\mathbf{n}$ and $\mathbf{h}$ respectively, at time i. We also introduce a distribution $p(\mathbf{y}|\mathbf{x}, \mathbf{h}, \mathbf{n})$ that describes how these random variables are combined to produce the observed value $\mathbf{y}$. At time step i, the combination of these three components is shown by the graph in Figure 4.2(a). The relationship between $\mathbf{y}$, $\mathbf{x}$, $\mathbf{h}$ and $\mathbf{n}$ was introduced in Equation (2.5). In Chapter 5, we will go into the details of the relationship and the associated distribution, which we call the *interaction distribution*.

Cleaning methods can employ simplified versions of the true model. For example, we can ignore the time dynamics of the speech and noise process, and a use smaller state space for speech. This is shown in Figure 4.2(b). We will use c^x to explicitly denote the state space for cleaning model, and s^x to designate the states of the true speech model.

Figure 4.3 shows a generative Bayes net for the speech process under noisy conditions. The top part of the graph is the speech model, as before. The bottom part of the graph models the noise process, where the state sequence $\mathbf{s}^n = s_1^n \dots s_M^n$ gives rise to the noise observations $\mathbf{N} = \mathbf{n}_1 \dots \mathbf{n}_M$. The speech and noise features are combined to produce the noisy speech features $\mathbf{Y} = \mathbf{y}_1 \dots \mathbf{y}_M$. The channel distortion is assumed to be stationary throughout the utterance and we omit it from the graph in Figure 4.3. In this graph, both the speech and noise are considered to be Markovian processes.

Given a sequence of clean speech observations $\mathbf{X}$, the goal of the speech decoding

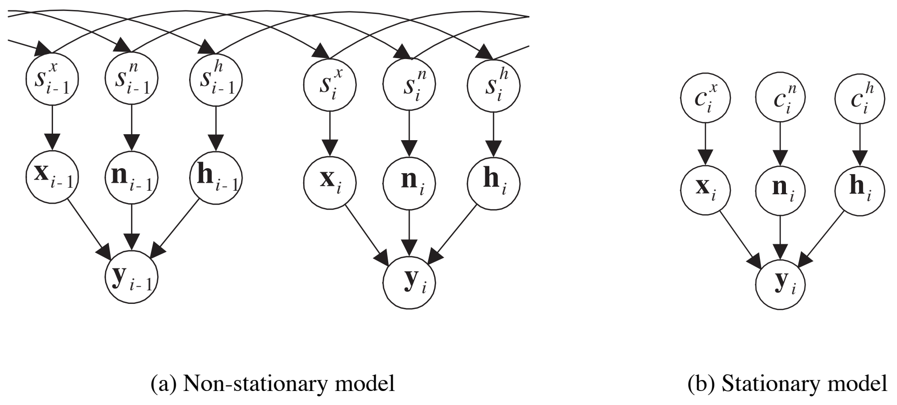


Figure 4.2: (a) Two frames of the dynamic generative model. (b) The stationary model for time frame i. This is graphical model for the joint distribution $p(\mathbf{y}, \mathbf{x}, \mathbf{n}, \mathbf{h}, c^x, c^n, c^h)$. Cleaning algorithms often employ simplified probabilistic models.

process of a speech recognition system is to find the state sequence $\mathbf{s}$ with the highest posterior probability $p(\mathbf{s}|\mathbf{X})$ among all possible state sequences[68]. The Viterbi algorithm allows this to be done efficiently[11, 80]. Similarly, given a sequence of noisy observations $\mathbf{Y}$, the goal is to find the state sequence $\mathbf{s}$ with the highest posterior probability $p(\mathbf{s}|\mathbf{Y})$. In other words, we would like to perform inference in the graph of Figure 4.3.

Although there are loops in this network, they do not cause exact inference to become intractable[50]. One can fold the network as is shown in Figure 4.4. In this network we have combined the states s^x and s^n into a combined state variable s and similarly, we have combined $\mathbf{x}$ and $\mathbf{n}$ into $\mathbf{z}$. The number of states in s is the product of the number of states s^x and s^n. For small noise state spaces, this is quite tractable. When performing inference, we can either find a single best noise state sequence[78] or sum over all noise paths, which is both theoretically more reasonable, and more amenable to

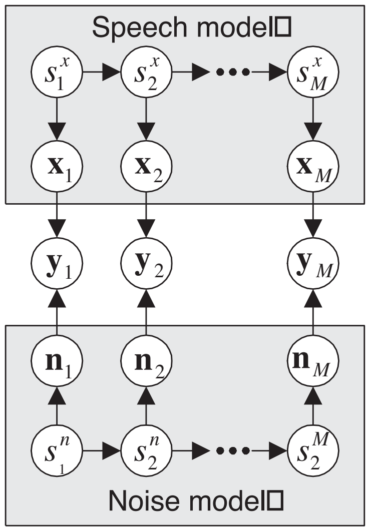


Figure 4.3: A Bayesian network of the noisy speech generation process. The upper model is equivalent to the traditional speech HMM with continuous observation densities. We model the noise process in a similar way. The noise and speech vectors, $\mathbf{n}_i$ and $\mathbf{x}_i$ combine to produce the observed noisy speech vectors $\mathbf{y}_i$.

implementation in a large vocabulary speech recognition system[50, 80].

Ideally, we would like to perform exact inference in this graph. However, exact inference is intractable due to the non-linear mixing of the speech $\mathbf{x}$, noise $\mathbf{n}$ and channel $\mathbf{h}$ to produce the observed corrupted speech $\mathbf{y}_{obs}$. Is Chapter 6 we discuss the cause of the intractability, and in Chapter 8, we introduce and analyze a method for approximate inference in this network.

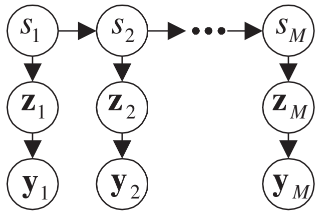


Figure 4.4: Bayesian network for the noisy speech process. The noise HMM of Figure 4.3 has been combined with the speech HMM. The new state variable s_i is the concatenation of the state variable s_i^x and s_i^n and the variable $\mathbf{z}$ is the concatenation of $\mathbf{x}$ and $\mathbf{n}$.

4.2 Approaches to Noise Robust Speech Recognition

The two environmental robustness paradigms that we have discussed, i.e. *feature cleaning* and *model adaptation*, fit well into the probabilistic graphical model viewpoint. At the end of this chapter we will also discuss the how the third paradigm i.e. *uncertainty decoding* follows from our desire to do inference proper in the graph in Figure 4.3.

4.2.1 Feature Cleaning

In the feature cleaning method[33], the goal is to restore the noisy features, such that they resemble clean features. We can use the joint conditional distribution $p(\mathbf{x}, \mathbf{n}, \mathbf{h}|\mathbf{y}_{obs})$ over clean speech $\mathbf{x}$, noise $\mathbf{n}$ and channel $\mathbf{h}$ for this purpose. In this case, we can use the Minimum Mean Squared Error (MMSE) criterion for estimating a point estimate $\hat{\mathbf{x}}$ for the speech vector $\mathbf{x}$,

$$\hat{\mathbf{x}} = \int \mathbf{x} \cdot p(\mathbf{x}|\mathbf{y}_{obs}) d\mathbf{x} = \int \mathbf{x} \cdot p(\mathbf{x}, \mathbf{n}, \mathbf{h}|\mathbf{y}_{obs}) d\mathbf{x} d\mathbf{n} d\mathbf{h}, \tag{4.1}$$

where $\mathbf{y}$ is the noise speech vector. This is equivalent to clamping $\mathbf{y}$ to $\mathbf{y}_{obs}$ and passing messages to $\mathbf{x}$ in the graph of Figure 4.2(a) (i.e. marginalizing over $\mathbf{x}$). This gives us

$p(\mathbf{x}, \mathbf{y}_{obs})$ from which we get $p(\mathbf{x}|\mathbf{y}_{obs})$ and then take the MMSE estimate. Note that when we take the MMSE estimate, we loose the information about uncertainty that is encoded in the distribution $p(\mathbf{x}|\mathbf{y}_{obs})$.

An advantage of this approach is that the complexity of the cleaning algorithm can be low. For example, we can use the time invariant graph in Figure 4.2(a), instead of the graph of the recognizer itself. Examples of methods that fit into this category are Spectral Subtraction[7] and Codebook Dependent Cepstral Normalization (CDCN)[1], which we will discuss in more detail in the Chapter 7.

4.2.2 Model Adaptation

It is known that if a speech recognizer is trained on noisy speech for a given noise type (i.e. *matched training*), the recognition rate improves substantially. The goal of model adaptation is to transform the acoustic models of the recognizer $p(\mathbf{x}|s)$ in such a way that they approximate the noisy speech models $p(\mathbf{y}|s)$, i.e. the models that would be obtained by training on speech in the current noise condition[26]. Hence the goal is to replace these models with $p(\mathbf{y}|s)$. If we have the joint class conditional distributions, we can accomplish this since

$$\hat{p}(\mathbf{y}|s) = \int \hat{p}(\mathbf{x}, \mathbf{n}, \mathbf{h}, \mathbf{y}|s) d\mathbf{x} d\mathbf{n} d\mathbf{h} \tag{4.2}$$

This is equivalent to clamping s^x and marginalizing over $\mathbf{y}$ in the graph of Figure 4.3 for each speech state s^x, and replacing the acoustic distributions in the recognizer with the resultant distributions $p(\mathbf{y}|s^x)$. In this case, the decoding is based on the noisy speech posterior $p(\mathbf{s}|\mathbf{Y})$.

Examples of methods that fall into this category include Parallel Model Combination (PMC)[29] and the Model Domain Vector Taylor Series (MD-VTS) method[62].

4.2.3 Uncertainty Decoding

Feature cleaning methods deliver a point estimate of the clean speech to the recognizer. Information about the uncertainty of the observation is lost. It is intuitively appealing to instead use a distribution that reflects the uncertainty of the cleaned observation $\hat{\mathbf{x}}$.

A third paradigm for robust speech recognition is to perform inference in the natural way suggested by the graph of Figure 4.3. If we do this, the message passed up the network from variable node $\mathbf{x}$ to variable node s^x, would be $f(\mathbf{x}) = p(\mathbf{y}_{obs}|\mathbf{x})$ which would be combined with $p(\mathbf{x}|s^x)$ to produce $p(\mathbf{y}_{obs}|s^x)$, where

$$p(\mathbf{y}_{obs}|s^x) \approx \int p(\mathbf{y}_{obs}|\mathbf{x}) p(\mathbf{x}|s^x) d\mathbf{x}. \tag{4.3}$$

The end result is that the recognizer bases its classification on $p(\mathbf{Y}|s)$. If we could perform exact inference, this would be equivalent to Model Adaptation.

However, the likelihood $f(\mathbf{x}) = p(\mathbf{y}_{obs}|\mathbf{x})$ has a very non-Gaussian form. Thus, we introduce a new method that achieves the same goal, but allows us to work with functions that are more accurately approximated by Gaussians.

In the *uncertainty decoding* paradigm, we approximate $p(s|\mathbf{y})/p(s)$ as[1]

$$p(\mathbf{s}|\mathbf{Y}) \approx p(s_0) \prod_i \frac{p(\mathbf{y}_i|s_i)}{p(\mathbf{y}_i)} \cdot p(s_i|s_{i-1}) = p(s_0) \prod_i \frac{p(s_i|\mathbf{y}_i)}{p(s_i)} \cdot p(s_i|s_{i-1}). \tag{4.4}$$

The uncertainty decoding method retains the information about the uncertainty of the observations, which feature cleaning discards due to the point estimate. We will see that this method differs from model adaptation because we are not attempting to produce a distribution over all $\mathbf{y}$ for a particular speech class. This allows us to tune the approximation to a particular observed $\mathbf{y}_{obs}$.

[1]In a second manifestation of the soft information paradigm, we approximate $p(\mathbf{x}|\mathbf{y}_{obs})/p(\mathbf{x})$.

Chapter 5

MFC Transform and the Interaction Likelihood

In this chapter we first discuss the Mel-Frequency Cepstrum (MFC) transformation. We then take the MFC transform of the environment equation, and discuss the how uncertainty about the relationship of the environmental components is introduced in the process of dimensionality reduction. This will lead to the definition of the *interaction likelihood* that describes how likely different combinations of x, n and h are once we have observed a particular value for the noisy observation y_{obs}.

5.1 The Mel Frequency Cepstrum Transform

Various acoustic features have been proposed in the literature [35]. Considerable work has been dedicated to finding an effective low dimensional representation of the speech signal. Some researchers have sought perceptually motivated features that mimic the acoustic processing of the human auditory system, while others have had other goals such as computational efficiency.

Most commercial grade speech recognition systems use a form of Mel-Frequency Cepstrum coefficients (MFCCs)[11, 38]. MFC features are both efficient to compute and share characteristics with human auditory processing[68]. MFC features have proven to perform well under noise free conditions (see Figure D.1 for a comparison of MFCC features to Log-spectrum-features). As a result, a considerable proportion of noise adaptation research has focuses on the MFC features[33].

The Mel Frequency Cepstrum (MFC) transform is used to transform a segment of the time signal into a set of features. There is a long history of using the MFC transform for speech recognition. There are biological and technical justifications for the various steps in the transform, but the most important reason for its use is its effectiveness in producing high recognition rates.

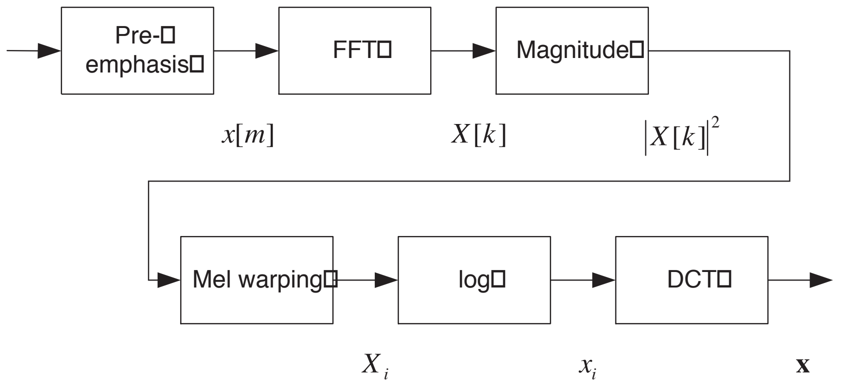


Figure 5.1: Block diagram of the Mel-Frequency Cepstrum Transformation.

There are several steps in the MFC transform as shown in Figure 5.1 [68]. The steps of the transform are:

1. Take a segment of the sampled signal. In the front end that we will use this is

25ms, or 200 samples when the signal is sampled at 8kHz. Then these samples are windowed with a Hamming window,

$$x[n] = w[n]x'[jS + n] \text{ for } 0 < n < L \tag{5.1}$$

where x' is the sampled signal, S is the step size (e.g. 10ms or 80 samples at 8kHz) and $w[n]$ is the windowing function

$$w[n] = 0.54 - 0.46 \cos\left(\frac{2\pi(n-1)}{(L-1)}\right), \tag{5.2}$$

where L is the window length (commonly 25ms or 200 samples at 8kHz).

2. Take the Fast Fourier Transform (FFT). This results in the complex Discrete Fourier transform features $X[k]$ [11].

$$X[k] = \sum_n x[n] \exp\left[\frac{-j2\pi kn}{K}\right], \tag{5.3}$$

where K is the length of the FFT, (commonly 256 samples)

3. Take the magnitude squared of the FFT features. Phase information is discarded in this step.

$$|X[k]|^2 = X[k] \cdot X^*[k]. \tag{5.4}$$

4. Mel warping. This operation averages the results of the FFT. It reduces the frequency resolution of the higher components. In the Aurora front end, this step reduced dimensionality from 256 coefficients to 23.

$$X_i = \sum_{k=0} f_i[k]|X[k]|^2 \tag{5.5}$$

where f_i is the i-th filterbank[72].

5. Take the logarithm:

$$x_i = \log(X_i). \tag{5.6}$$

6. Take the discrete cosine transform (DCT). After this step, the dimensionality is further reduced by dropping dimensions 14 and higher, leaving a 13 dimensional vector, for the original frame.
$$\mathbf{x}_c = \mathbf{C}\mathbf{x}. \tag{5.7}$$
Where the elements of the C matrix are defined as follows[72]:
$$c_{i,j} = \sqrt{\frac{2}{N}} \cos\left(\frac{\pi i}{N}(j - 0.5)\right). \tag{5.8}$$

7. Calculate time derivatives or delta and acceleration features. The delta coefficients are also 13 dimensional as are the acceleration features. The results is a 39 dimensional MFCC feature vector. The delta parameters are calculated from the base features by
$$\Delta x_t = \frac{\sum_{r=1}^{R} R \cdot (x_{t+r} - x_{t-r})}{2\sum_{r=1}^{R} R^2}, \tag{5.9}$$
where R is a parameter (generally 2 or 3). The acceleration parameters are calculated similarly from the delta parameters[72].

In addition to good performance there are several theoretical motivations behind the use of the MFC transform.

- The phase of the speech signal is discarded as it is not thought to contain much information.
- The Mel frequency warping produces frequency resolution that resembles the frequency resolution of the human ear[83].
- The log operation matches the amplitude sensitivity of the human ear[83].
- The FFT-log-DCT combination performs a blind de-convolution of the signal and separates the features associated with the shape of the vocal tract and the features that represent the voicing.

Perhaps equally important is the effect of the transform on the probability distributions of the speech features [19]. It can be shown that if the signal is a white Gaussian process in the time domain, the spectrum features will be Rayleigh distributed. The Mel-warped power spectrum features are approximately Chi-squared distributed while the log spectrum and cepstrum features are well modelled by mixtures of Gaussians. Thus, from the perspective of using parametric pdf functions to approximate these distributions, namely GMMs, the MFC transform also has advantageous effects since

- the distributions in the MFC domain are well modelled by mixtures of Gaussians,
- the gain of the signal is only reflected by the first coefficient c_0. If this coefficient is discarded, then the features are gain-invariant.

The MFC transform serves to reduce the dimensionality of the features. For example, in the Aurora reference front-end the frame is of length 200 samples. After Mel-frequency warping the dimensionality is reduced to 23. Higher quefrency components are discarded after the DCT transformation leaving a 13 dimensional vector, which is then expanded to 39 dimensions by appending time derivatives.

Dimensionality reduction is an important component of any pattern recognition system. However, from a source separation viewpoint, it is important that the features of the desired and interfering signals not overlap completely in feature space. Dimensionality reduction is a many to one mapping which may cause signals that are separable in the linear or spectrum domain to become inseparable in a cepstrum domain. In other words, the optimal feature representation for robust speech recognition may not be the same as the optimal feature representation for speech recognition in clean conditions. For example, retaining the voicing information, which is lost in the Mel frequency transformation, and the truncation of the DCT matrix, may be important to robust speech recognition.

This is the motivation behind extended representations, such as using microphone arrays, or multi-modal speech recognition (using speech and visual lip features). In this thesis, however, we will try to do the best we can with the features we already have, namely the MFC features.

5.2 The MFC Transform of the Interaction Equation

We will now derive the equations for the noisy signal $\mathbf{y}$ in terms of $\mathbf{x}$,$\mathbf{n}$ and $\mathbf{h}$ at all stages of the MFC transform. These results will be required in the discussion of the *interaction likelihood*.

The standard environment model was introduced Chapter 2 (see Figure 2.2). In the time domain the relationship is

$$y[m] = \sum_{k=0}^{K-1} x[m-k]h[k] + n[m], \tag{5.10}$$

where $x[m]$ is a sample of the clean speech signal, $n[m]$ is a sample of the noise signal, $h[m]$ is the impulse response of the linear channel, $y[m]$ are the samples of the noisy speech signal.

The first step in the MFC transform is to take the Discrete Time Fourier transform of the signal,

$$Y[k] = X[k]H[k] + N[k]. \tag{5.11}$$

Next we take the magnitude square[1],

$$|Y[k]|^2 = |X[k]|^2|H[k]|^2 + |N[k]|^2 + 2|X[k]||H[k]||N[k]|cos\theta_k, \tag{5.12}$$

[1]The Aurora reference front end uses the magnitude while the front end of the Microsoft Whisper speech recognition system uses magnitude squared.

where θ_k is the angle between $X[k]H[k]$ and $N[k]$. The phase information is lost in this operation.

We then do Mel-frequency binning, which reduces the resolution of higher frequency components,

$$\begin{aligned}\sum_k w_k^i |Y[k]|^2 &= \sum_k w_k^i |X[k]|^2 |H[k]|^2 + \sum_k w_k^i |N[k]|^2 \\ &+ 2\sum_k w_k^i |X[k]||H[k]||N[k]|cos\theta_k, \end{aligned} \tag{5.13}$$

where w_k^i are the weights of filter bank i and $\sum_k w_k^i = 1$ for all i. The Mel-binning operation can be seen as having the effect of weighting the lower components of the DFT more heavily when calculating acoustic scores.

To simplify the notation we define:

$$\begin{aligned} Y_i^2 = \sum_k w_k^i |Y[k]|^2, &\quad X_i^2 = \sum_k w_k^i |X[k]|^2, \\ N_i^2 = \sum_k w_k^i |N[k]|^2, &\quad H_i^2 = \sum_k w_k^i |H[k]|^2. \end{aligned}$$

Also define:

$$X_i^2 H_i^2 = \left(\sum_k w_k^i |X[k]|^2\right)\left(\sum_k w_k^i |H[k]|^2\right) \tag{5.14}$$

$$= \sum_k w_k^i |X[k]|^2 |H[k]|^2 - \epsilon_{CTi}. \tag{5.15}$$

The error term ϵ_{CTi} is 0 only if $H[k]$ is constant over the bins of each filterbank. If the channel is not constant, then this assumption introduces error due to the cross terms.

We then define the pseudo-normalized phase related term:

$$\alpha_i = \frac{\sum_k w_k^i |X[k]||H[k]||N[k]|cos\theta_k}{\sqrt{\sum_k w_k^i |X[k]|^2}\sqrt{\sum_k w_k^i |H[k]|^2}\sqrt{\sum_k w_k^i |N[k]|^2}}. \tag{5.16}$$

Substituting this into Equation (5.13) we get

$$Y_i^2 = X_i^2 H_i^2 + N_i^2 + 2\alpha_i |X_i||H_i||N_i| + \epsilon_{CTi}. \tag{5.17}$$

The next step is to take the $\log$. Taking the $\log$ of Equation (5.17) and using the notation $y_i = \ln Y_i^2, x_i = \ln X_i^2, h_i = \ln H_i^2, n_i = \ln N_i^2$, we arrive at

$$\begin{aligned} y_i = x_i + h_i + \ln \Big[& 1 + \exp(n_i - x_i - h_i) \\ & + 2\alpha_i \exp((n_i - x_i - h_i)/2) + \epsilon_{CTi} \exp(-x_i - h_i) \Big] \end{aligned} \tag{5.18}$$

or

$$\begin{aligned} y_i = {} & x_i + h_i + \ln(1 + \exp(n_i - x_i - h_i)) \\ & + \ln\left(1 + \frac{2\alpha_i \exp\left((n_i - x_i - h_i)/2\right) + \epsilon_{CTi} \exp(-x_i - h_i)}{1 + \exp(n_i - x_i - h_i)}\right). \end{aligned} \tag{5.19}$$

Since the dimensions are independent, we arrive at the *interaction equation* in the log spectrum domain

$$\mathbf{y} = \mathbf{x} + \mathbf{h} + \ln(1 + \exp(\mathbf{n} - \mathbf{x} - \mathbf{h})) + \mathbf{e}, \tag{5.20}$$

where $\mathbf{x} = [x_1, \ldots, x_N]^T$ and $\mathbf{e}$ is the error term

$$\mathbf{e} = \ln\left(1 + \frac{2\alpha \exp\left((\mathbf{n} - \mathbf{x} - \mathbf{h})/2\right) + \epsilon_{CT} \exp(-\mathbf{x} - \mathbf{h})}{1 + \exp(\mathbf{n} - \mathbf{x} - \mathbf{h})}\right). \tag{5.21}$$

For notational convenience, we define:

$$g(z) = \ln(1 + \exp(z)) \tag{5.22}$$

and write Equation (5.20) as

$$\mathbf{y} = \mathbf{x} + \mathbf{h} + \mathbf{g}(\mathbf{n} - \mathbf{x} - \mathbf{h}) + \mathbf{e}. \tag{5.23}$$

When modelling the relationship between $\mathbf{y}, \mathbf{x}, \mathbf{h}$ and $\mathbf{n}$, most noise robustness techniques overlook the error $\mathbf{e}$ in Equation (5.19) and assume the relationship is exact. Below we will describe two ways in which the error term can be taken into account by using a distribution over $\mathbf{y}$ given $\mathbf{x}, \mathbf{h}$ and $\mathbf{n}$.

Finally we take the DCT. Defining $\mathbf{x}_C = \mathbf{C}\mathbf{x}$ etc. we write

$$\mathbf{y}_C + \mathbf{x}_C + \mathbf{h}_C + \mathbf{C}g\left(\mathbf{C}^{-1}(\mathbf{n}_C - \mathbf{x}_C - \mathbf{h}_C)\right) + \mathbf{C}\mathbf{e}, \tag{5.24}$$

where $\mathbf{C}$ is the truncated DCT matrix.

5.3 The Interaction Likelihood

Due to the error term, the relationship between $\mathbf{y}, \mathbf{x}, \mathbf{n}$ and $\mathbf{h}$ is not exact. We will take this into account by using a probability distribution.

5.3.1 Fixed Variance Interaction Likelihood

A first approximation is to assume that the error term $\mathbf{e}$ is zero mean and Gaussian. This leads to the distribution over $\mathbf{y}$ in the log-spectrum domain

$$p(\mathbf{y}|\mathbf{x}, \mathbf{h}, \mathbf{n}) = N(\mathbf{y}; \mathbf{x} + \mathbf{h} + \ln(1 + \exp(\mathbf{n} - \mathbf{x} - \mathbf{h})), \mathbf{\Psi}). \tag{5.25}$$

In this case, the variance is fixed. We will call this the *interaction distribution*.

Since our observations are noisy speech features $\mathbf{y}_{obs}$, we are more interested in the *interaction likelihood*

$$\begin{aligned} f(\mathbf{x}, \mathbf{n}, \mathbf{h}) &= p(\mathbf{y} = \mathbf{y}_{obs}|\mathbf{x}, \mathbf{h}, \mathbf{n}) \\ &= N(\mathbf{y} = \mathbf{y}_{obs}; \mathbf{x} + \mathbf{h} + \ln(1 + \exp(\mathbf{n} - \mathbf{x} - \mathbf{h})), \mathbf{\Psi}). \end{aligned} \tag{5.26}$$

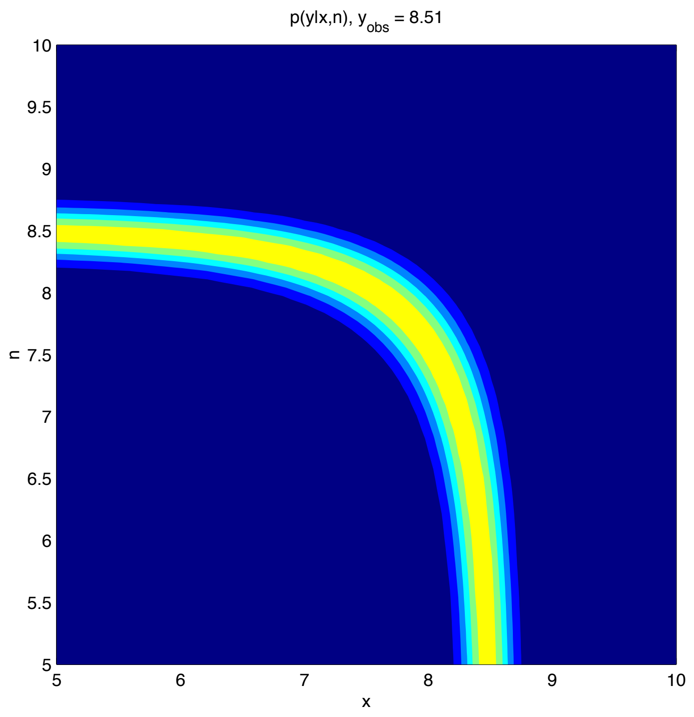


Figure 5.2: Magnitude independent interaction likelihood $f(x, n) = p(y = 8.51|x, n)$.

Figure 5.2 shows the a plot of 1 filter bank component of the interaction likelihood for $y_{obs} = 8.51$. This plot shows the combinations of noise and speech that are likely to produce the observation. For example, if speech dominates noise, then we lie on vertical part of the curve. If noise dominates speech, then we lie on the horizontal portion.

5.3.2 Interaction Likelihood with Magnitude Dependent Variance

The relative magnitude of $\mathbf{x}$ and $\mathbf{n}$ does have an effect on the size of the error in the interaction equation.

We can take this into account in the following way[2]. Recall the magnitude squared relationship in Equation (5.12),

$$|Y[k]|^2 = |X[k]|^2|H[k]|^2 + |N[k]|^2 + 2|X[k]||H[k]||N[k]|cos\theta_k. \qquad (5.27)$$

Even if we have knowledge of $|X[k]|^2$, $H[k]|^2$ and $|N[k]|^2$, uncertainty about $|Y[k]|^2$ is due to the lack of knowledge about the phase θ_k. Since $N[k]$ is independent of $X[k]$ and $H[k]$, θ_k is uniform in $(-\pi, \pi)$. Hence we introduce a random variable $r_k = \cos(\theta_k)$ which has the distribution

$$p(r) = \begin{cases} \frac{1}{\pi\sqrt{1-r^2}} & \text{if } |r| < 1 \\ 0 & \text{otherwise.} \end{cases} \qquad (5.28)$$

If $X[k]$, $H[k]$ and $N[k]$ are constant within a filter bank i, the random variable α_i is

$$\alpha_i = \sum_k w_k^i r_k. \qquad (5.29)$$

This random variable is zero mean and has probability 0 outside of (-1,1). The variance tends to 0 as the number of bins in the filter tends to infinity. Although α_i is dependent on $X[k]$, $H[k]$ and $N[k]$ we will assume it is zero mean with variance ψ_i.

Now we can write (omitting ϵ_{CT})

$$\begin{aligned} \mathbf{y} &= \mathbf{x} + \mathbf{h} + \ln(1 + e^{(\mathbf{n}-\mathbf{x}-\mathbf{h})}) + \ln\left(1 + \frac{2\alpha e^{((\mathbf{n}-\mathbf{x}-\mathbf{h})/2)}}{1 + e^{(\mathbf{n}-\mathbf{x}-\mathbf{h})}}\right) & (5.30) \\ &= \mathbf{x} + \mathbf{h} + \ln(1 + e^{(\mathbf{n}-\mathbf{x}-\mathbf{h})}) + \ln\left(1 + \frac{\alpha}{\cosh(\mathbf{n}-\mathbf{x}-\mathbf{h})}\right). & (5.31) \end{aligned}$$

The last term in equation 5.30 using the first term in the Taylor series approximation for $\ln(1 + x)$:

$$\ln\left[1 + \frac{\alpha}{\cosh(\mathbf{n}-\mathbf{x}-\mathbf{h})}\right] \approx \frac{\alpha}{\cosh(\mathbf{n}-\mathbf{x}-\mathbf{h})}, \qquad (5.32)$$

[2]Derivation due to Alex Acero

which is accurate for small values of α. We can therefore write

$$\mathbf{y} \approx \mathbf{x} + \mathbf{h} + \ln(1 + e^{(\mathbf{n}-\mathbf{x}-\mathbf{h})}) + \frac{\alpha}{\cosh(\mathbf{n}-\mathbf{x}-\mathbf{h})}. \tag{5.33}$$

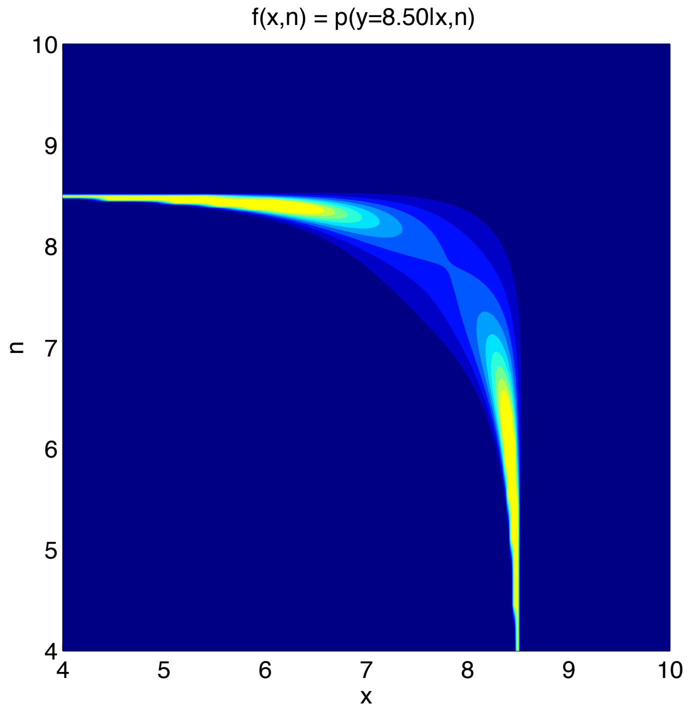


Figure 5.3: Magnitude dependent interaction likelihood $f(x,n) = p(y = 8.5|x,n)$. In contrast to the magnitude independent case (see Figure 5.2) the variance decreases as the difference in magnitude of x and n increases.

Assuming α is zero mean with variance ψ we arrive at the interaction equation with magnitude dependent variance:

$$p(\mathbf{y}|\mathbf{x},\mathbf{h},\mathbf{n}) = N\left(\mathbf{y};\mathbf{x} + \mathbf{h} + \ln(1 + e^{(\mathbf{n}-\mathbf{x}-\mathbf{h})}), \frac{\psi}{[\cosh(\mathbf{n}-\mathbf{x}-\mathbf{h})]^2}\right) \tag{5.34}$$

From the expression of the variance, we see that if $n_i \approx x_i$ and h_i is small, then the denominator will be close to 1. Thus the variance is largest around the bend in Figure 5.3. If noise dominates speech or speech dominates noise, then the denominator will be large, and the variance small.

5.3.3 Empirical plot of Interaction Likelihood

We would like to empirically assess the error in the interaction equation, for a particular observed value y_{obs}, since this would reveal the form of the interaction likelihood. To do this, we could compute the log-spectrum features of a clean file, the log-spectrum features of a noise file and the log-spectrum features of noisy speech file that results from mixing the two. We could then plot all data pairs for $\mathbf{x}$, $\mathbf{n}$ for a small range of y values (assuming no channel distortion).

In the log-spectrum domain the sample set $\{\{x_1, n_1, y_1\}, \ldots, \{x_N, n_N, y_N\}\}$ (N is the number of observations) can be viewed as a representation of the joint distribution $p(x, n, y)$. By noting that $p(x, n, y) = p(x + \Delta, n + \Delta, y + \Delta)$, we can produce a plot that is proportional to $p(x, n|y_{obs}) = p(y_{obs}|x, n)/(p(x) \cdot p(n))$. This gives us a good appreciation for the true interaction likelihood $p(y_{obs}|x, n)$, assuming $p(x) \cdot p(n)$ is relatively flat. The plot in Figure 5.4 was produced calculating the value $\Delta_i = 8.5 - y_i$ and then plotting $(x_i - \Delta_i, n_i - \Delta_i)$ for dimension 6 for data files at 20dB.

Notice how remarkably similar the magnitude dependent likelihood in Figure 5.3 is to the scatter plot in Figure 5.4.

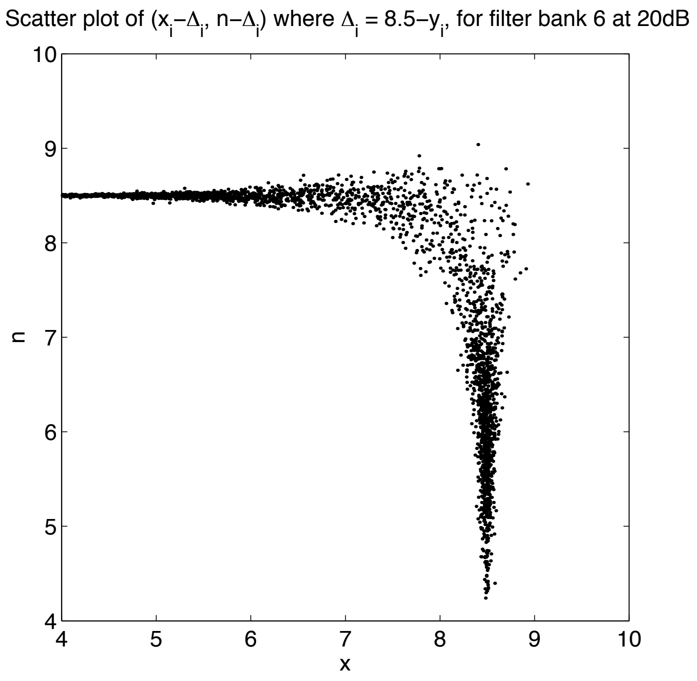


Figure 5.4: Scatter plot of $p(x, n|y_{obs}) = p(y_{obs}|x, n)/(p(x) \cdot p(n))$. This plot gives a good idea of the form of the true interaction likelihood $p(y_{obs}|x, n)$. Notice how similar this plot is to the magnitude dependent likelihood in Figure 5.3

Chapter 6

Inference in Non-Gaussian Networks

In this chapter we will discuss inference in the network of Figure 4.2(b). This is the simplified "cleaning" network where the time dynamics of the speech and noise process have been ignored. This model will be used in the MMSE-Algonquin algorithm that is the subject of Chapter 8. We will start by discussing the speech, noise and channel component models and then plot the posterior distribution over x and n for a given observed y_{obs}. This will demonstrate how the non-Gaussian interaction likelihood is the cause of intractability of inference.

6.1 The Component Models

To perform inference in the graph of Figure 4.2(b) we need to specify the components of the model. These are, the prior speech model $p(\mathbf{x})$, the prior noise model $p(\mathbf{n})$, the prior channel model $p(\mathbf{h})$ and the interaction likelihood $p(\mathbf{y}|\mathbf{x}, \mathbf{n}, \mathbf{h})$.

6.1.1 The Speech Model

Many modern speech recognizers use Gaussian mixtures to model acoustic observations[68]. We will also use a mixture of Gaussians to model speech. Thus

$$p(\mathbf{x}) = \sum_{s^x} p(s^x)p(\mathbf{x}|s^x) = \sum_{s^x} \pi_{s^x} N(\mathbf{x}; \boldsymbol{\mu}_{s^x}, \boldsymbol{\Sigma}_{s^x}). \tag{6.1}$$

The parameters of the speech model can found either from the parameters of the clean speech models of the recognizer or found by training a mixture model directly on clean speech features.

The soft information paradigm that we discuss in Chapter 10, requires the correspondence of the states of the recognizer and the classes of the speech model. For large vocabulary tasks, there can be tens of thousands of states.

The feature cleaning paradigm has the advantage that we can used a simplified speech model that has far fewer mixtures than the model constructed from the acoustic models of the recognizer.

6.1.2 The Noise Model

Similarly to the speech model, we used a mixture of Gaussians to model noise:

$$p(\mathbf{n}) = \sum_{s^n} p(s^n)p(\mathbf{n}|s^n) = \sum_{s^n} \pi_{s^n} N(\mathbf{n}; \mu_{s^n}, \boldsymbol{\Sigma}_{s^n}). \tag{6.2}$$

It some cases, it suffices to use a single mixture for the noise model e.g., for low intensity office noise. In other cases, significant gains in recognition accuracy can be achieved by using multiple mixtures. If the noise process is non-Gaussian, e.g., if it is non-stationary, then multiple noise mixtures provide a more accurate model[50].

Significant gains can be also achieved by adapting the parameters of the noise model to the current noise conditions, e.g., by using a generalized EM approach.

6.1.3 The Channel Model

The channel will be modelled in exactly the same way as the speech and noise models

$$p(\mathbf{h}) = \sum_{s^h} p(s^h)p(\mathbf{h}|s^h) = \sum_{s^h} \pi_{s^h} N(\mathbf{h}; \mu_{s^h}, \mathbf{\Sigma}_{s^h}). \tag{6.3}$$

The channel is often slowly varying or constant throughout an utterance. In many cases, a point estimate is a sufficiently good model of the channel, which is a special case of Equation (6.3) when we use a single component and the variance approaches zero. For modelling consistency we use this expression. Another reason for using this expression is that when learning the channel parameters from data, the convergence rate is highly dependent on the variance.

For notational purposes, we will denote the combined column vector $[\mathbf{x}; \mathbf{n}; \mathbf{h}]$ as $\mathbf{z}$. We can then use the shorthand

$$p(\mathbf{z}|s) = p(\mathbf{x}|s^x)p(\mathbf{n}|s^n)p(\mathbf{h}|s^h) =$$
$$N(\mathbf{z}; \boldsymbol{\mu}_s, \mathbf{\Sigma}_s) = N\left(\begin{bmatrix}\mathbf{x}\\ \mathbf{n}\\ \mathbf{h}\end{bmatrix}; \begin{bmatrix}\boldsymbol{\mu}^x_{s^x}\\ \boldsymbol{\mu}^n_{s^n}\\ \boldsymbol{\mu}^h_{s^h}\end{bmatrix}, \begin{bmatrix}\mathbf{\Sigma}^x_{s^x} & 0 & 0\\ 0 & \mathbf{\Sigma}^n_{s^n} & 0\\ 0 & 0 & \mathbf{\Sigma}^h_{s^h}\end{bmatrix}\right). \tag{6.4}$$

The joint distribution over $\mathbf{x}$,$\mathbf{n}$,$\mathbf{h}$ and $\mathbf{y}$ and the class variables s^x,s^n,s^h is thus

$$p(\mathbf{y}, \mathbf{x}, \mathbf{n}, \mathbf{h}, s^x, s^n s^h) = p(\mathbf{y}|\mathbf{x}, \mathbf{n}, \mathbf{h})p(s^x)p(\mathbf{x}|s^x)p(s^n)p(\mathbf{n}|s^n)p(s^h)p(\mathbf{h}|s^h), \tag{6.5}$$

or

$$p(\mathbf{y}, \mathbf{z}, s) = p(\mathbf{y}|\mathbf{z})p(s)p(\mathbf{z}|s). \tag{6.6}$$

6.2 Inference

The last component we need in order to perform inference in the model of Figure 4.2(b) is the interaction likelihood, which was discussed in Chapter 5.

We now have all the components of the network in Figure 4.2(b), i.e. the components required to express the joint distribution $p(\mathbf{y}, \mathbf{x}, \mathbf{n}, s^x, s^n)$, omitting the channel for clarity.

In the log-Mel-spectrum domain, the dimensions of $\mathbf{x}$, $\mathbf{n}$ and $\mathbf{y}$ decouple. Each dimension corresponds to energy over a small frequency range. We can therefore plot a single dimension or *frequency bin*.

In order to choose a representative speech model component, we use state 4 of the speech model for the word '*three*'. The observation vector is taken from the middle of the word '*three*', from a file with noise at 15dB SNR. The single component noise model $p(n_6|s^n = 1)$ is estimated from the first 20 frames of the file.

The top left plot in Figure 6.1 shows dimension 6 of the speech model $p(x_6|s^x = 4)$. The top right plot shows dimension 6 of the noise model $p(n_6|s^n = 1)$. The bottom right plot shows the interaction likelihood $f(x_6, n_6) = p(y_{6,obs}|x_6, n_6)$ for a particular observation $y_6 = 9.16$.

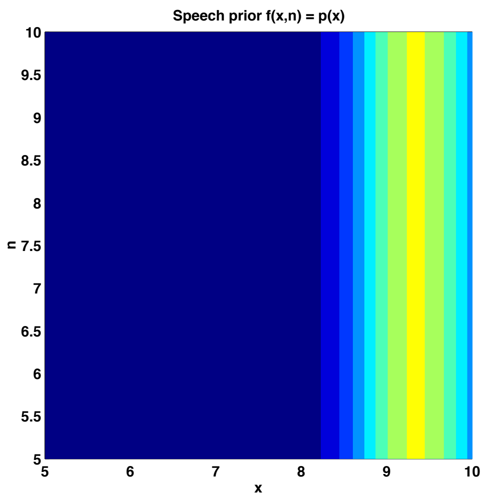
The bottom right plot in Figure 6.1 shows the joint distribution over noisy speech, clean speech and noise $p(y_6 = 9.16, x_6, n_6|s^x = 4, s^n = 1)$,

$$p(y_6 = 9.16, x_6, n_6|s^x = 4, s^n = 1) = \\ p(y_6 = 9.16|x_6, n_6)p(x_6|s^x = 4)p(n_6|s^n = 1) \quad (6.7)$$

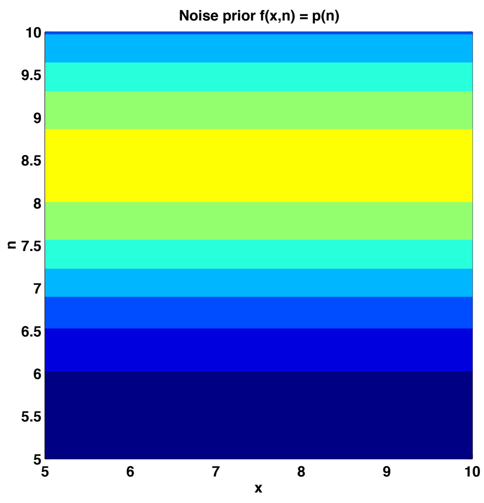
There are a few things to note about these plots. First is the relative variance of the noise and speech components. A single component noise model tends to have much greater variance than the components of the speech models.

Secondly, and more importantly, the joint distribution curves as it approaches the bend in of the interaction likelihood. This effect is the root of the intractability of exact inference in the network, because we cannot easily integrate this function to find marginal distributions.

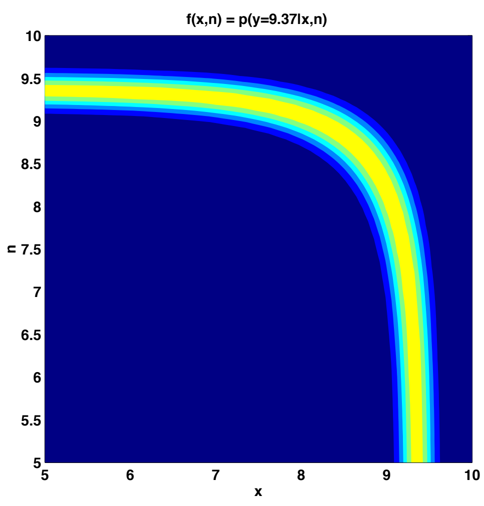
It is known that the Gaussian form is very convenient from a computational perspective. In Chapter 8 we discuss how to perform exact inference in the network, by using an approximations to the interaction likelihood. The method relies on using a linearization of the interaction likelihood. The expansion point of the linearization is iteratively improved to minimize the error in the approximation to the posterior (see Figure 8.2). The result of using this approximations is that the posterior becomes Gaussian and inference becomes tractable.



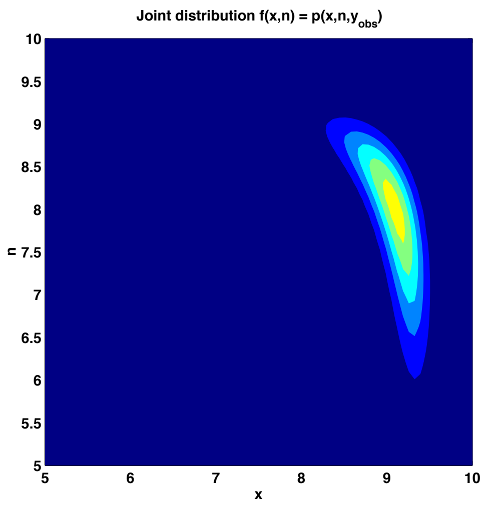
(a) Speech prior.



(b) Noise prior.



(c) Non-linear interaction likelihood.



(d) Non-linear joint distribution.

Figure 6.1: (a) The speech model $p(x_6|s^x = 4)$. (b) The noise model $p(n_6|s^n = 1)$. (c) the interaction likelihood $p(y_6 = 9.16|x_6, n_6)$. (d) The joint distribution $p(y_6 = 9.16, x_6, n_6|s^x = 4, s^n = 1)$. Notice the non-Gaussian form of the joint distribution.

Chapter 7

Performance Evaluation and the Prior Art

It is possible to calculate the probability of error of a speech recognizer in the same way that was done for the simple vowel/fricative classifier in Chapter 3, i.e. by using a Monte-Carlo method. Although this may be useful for gauging the relative error of different robustness methods, or to assess the confusion probabilities of specific models it is more common to evaluate the performance of a robustness method on real data[37].

A common way of reporting the performance of a Robustness Automatic Speech Recognition (RASR) method, is to compare it to the performance of Spectral Subtraction[7]. However, each research group has its own implementation of Spectral Subtraction, which perform differently[1].

[1]Table 7.2 shows the results of running a faithful implementation of Boll's original Spectral subtraction on Set A of the Aurora data set.

7.1 Measuring Performance

In speech recognition, the most common figure of merit is based on word accuracy. It is also possible to use other measures, such as sentence accuracy or even “meaning” accuracy, when the recognizer has to recognize a smaller set of key words that relate to the meaning of the sentence. In this thesis we will base the accuracy measures on words.

Percent Accuracy

Percent Accuracy is defined as[72]

$$\text{Percent Accuracy} = \frac{N - D - S - I}{N} \times 100\%, \tag{7.1}$$

where I is the number of inserted words, S is the number of substituted words, D is the number of deleted words and N is the total number of words in the transcription.

Word Error Rate

The Word Error Rate is defined as

$$WER = \frac{I + S + D}{N} \times 100\% = 100\% - \text{Percent Accuracy}. \tag{7.2}$$

Accuracy and WER will be the preferred way of reporting absolute performance.

Relative Reduction in WER

When assessing the relative merits of a particular method, a more informative measure is the relative reduction in WER

$$\text{Relative WER Reduction} = 100\% - \frac{WER_{\text{Method 2}}}{WER_{\text{Method 1}}} \times 100\%. \tag{7.3}$$

7.1.1 The Aurora Evaluation Framework

Until recently there was no good method for different groups working on robust speech recognition to compare the performance of their RASR methods. Different groups used different data sets with different noise types, using different ways for calculating the noise and signal levels when estimating SNR.

The Aurora 2 data set was produced by the European Telecommunications Standards Institute (ETSI) STQ-AURORA DSR Working Group [37]. The data-set includes training data, test data, a standard front end and the HTK speech recognition system[72]. The motivation for the creation of the database was to evaluate different methods for distributed speech recognition, i.e. recognition of speech from mobile phones, kiosks etc. Since the Aurora 2 is a complete recognizer with testing and training data, only the noise robustness machinery changes from one group to the next.

The Aurora data set will serve as the major means of evaluating the different methods discussed in this thesis. The database uses a subset of the TI digits database, which contains spoken digits under clean conditions. Each noise condition consists of 1001 files containing 1 to 5 spoken digits each. Noise is added to these files at different Signal to Noise Ratios (SNR). There are three test sets, A, B and C and two training sets, *clean* and *multicondition*. Sets A and B have and the multicondition training set have additive noise and are filtered to match a G.712 frequency response characteristic while set C has additive noise and different channel distortion due to being filtered to match a the MIRS frequency response characteristic.

Noise is added to the speech files at 6 different levels, (20dB, 15dB, 10dB, 5dB, 0dB and -5dB). There are various ways of calculating the SNR depending on how the signal and noise levels are computed. In Aurora, the signal level is computed from the speech portions of the file only.

	Subway	Car	Babble	Exhibit	Average
Clean	98.96	99.06	99.02	99.17	99.05
20 dB	97.02	88.33	96.06	96.39	94.45
15 dB	93.25	70.98	85.09	90.77	85.02
10 dB	78.42	46.43	57.71	72.82	63.84
5 dB	50.32	25.09	27.56	40.05	35.75
0 dB	24.04	12.18	11.36	13.95	15.38
-5 dB	12.16	7.65	8.62	8.39	9.21
Average	68.61	48.60	55.56	62.80	58.89

Table 7.1: Accuracy for Set A of the Aurora 2 database for different noise types at different SNRs. No processing has been performed on the noisy speech data.

In Set A the noises that are added to the clean speech file are:

- **Subway:** Sounds of a subway terminal and trains going by
- **Car:** Sound inside a driving car
- **Babble:** the sound of a room full of people talking simultaneously
- **Exhibit:** the sound of an art exhibit

These sounds are all non-stationary sounds where *Car* noise is the least varying, and *Subway* is the most varying. Similarly, Set B contains *Restaurant*, *Street*, *Airport* and *Station*. Set C contains two noise conditions, *Subway M* and *Street M* noise which have been filtered with the MIRS filter. Baseline results, i.e. results when the noisy files are fed directly to the recognizer, are give in Table 7.1.

Set A and B are similar in the respect that there is no mismatch between the channel in the training and testing sets. A robustness method does not need to correct for any channel mismatch. The difference between sets A and B, besides the different noises, is that for set A, it is allowable to incorporate global knowledge of the noise into a robustness method. Some robustness methods build a mapping from noisy speech to clean speech. These methods are called “stereo” based methods, in the sense that they are constructed using clean speech and the same speech after it has been distorted (i.e. not binaural). They are non-parametric and therefore do not adapt well to new noise environments. This is the reason for sets A and B, since is allowable to construct such a mapping from set A, but not for set B. SDCN[1] and SPLICE[12] are examples of “stereo” based methods. However, in this thesis we look only at parametric methods that incorporate an environment model, and use only the current file to estimate the parameters of the noise mode. Sets A and B are therefore equivalent for our purposes. The purpose of set C is to asses the effectiveness of a robustness method at handling channel distortion.

7.1.2 The Aurora-HTK Recognizer

The Aurora 2 database also describes the word models and the training procedures for the CUED-HTK[2] speech recognition system[72]. There are 11 word models (*one*, *two*, *three*, *four*, *five*, *six*, *seven*, *eight*, *nine*, *zero*, *oh*), each of which is a 16 state forward HMM. In addition there are two silence models; a 3 state model, and a single state model.

7.2 The Prior Art

In this section, the most relevant feature domain and model domain methods will be discussed, and recognition results on the Aurora 2 database will be given for some of them.

[2]Cambridge University, Engineering Department Hidden Markov Model Tool Kit

7.2.1 Feature Cleaning

These methods alter the noisy features $\mathbf{y}$ such that they resemble the clean features $\mathbf{x}$. Referring back to the block diagram of a speech recognition system in Figure 2.1, these change the features before they enter the **Acoustic scores** block. They can work in the time domain, i.e. before the **Feature extraction** block. They can work in the MFCC domain, i.e. they alter the features between the **Feature extraction** block and the **Acoustic scores** block or they replace part of this block. Methods that fall into this category include Spectral Subtraction (SS)[7], Cepstrum Mean Normalization (CMN)[25], the RelAtive SpecTrAl method (RASTA)[36], Codebook Dependent Cepstral Normalization (CDCN)[1] and SPLICE[12].

A characteristic of these methods is that they produce a point estimate of the clean features. Hence, they discard what is known about the uncertainty of the features.

Spectral Subtraction

It is informative to look at Spectral Subtraction, since it is the most straightforward and perhaps most widely used feature domain technique[7]. Various extensions to this method have been proposed[46]. Spectral Subtraction does not use a prior model for speech.

In its original form SS is a time domain method and is intended for removing the effects of additive noise $n[m]$ only, hence the interaction equation is $y[m] = x[m] + n[m]$ or using the power spectrum: $|Y(f)|^2 = |X(f)|^2 + |N(f)|^2$. Boll assumed that the effects of the noise can be modelled as a bias in the spectrum domain, i.e. the noise is assumed to be steady state noise with zero variance. The bias is estimated from a section of the signal that contains only noise (this necessitates a good speech/non-speech detector),

$$|\widehat{N(f)}|^2 = \frac{1}{M}\sum_{i=0}^{M-1} |Y_i(f)|^2. \tag{7.4}$$

The noise bias is then subtracted from the short time spectrum values. The estimate of $|X(f)|^2$ is

$$|\widehat{X(f)}|^2 = |Y(f)|^2 - |\widehat{N(f)}|^2. \tag{7.5}$$

Due to the point estimate of the noise, this method suffers from the fact that the $|\widehat{X(f)}|^2$ can become negative. This is in fact only one of 5 steps in the original spectral subtraction algorithm.

Spectral subtraction is often used as the baseline method that other methods are compared to. Results for a faithful implementation of SS are shown in Table 7.2. Notice that the performance of high SNR deteriorates compared to not processing the features (see Figure 7.1). Due to the rectification of Equation (7.5) and noise/speech classification, distortion is actually *introduced* into the signal at higher SNRs causing recognition performance to degrade. For lower SNR the performance is better than if nothing is done.

MMSE Vector Taylor Series

The Vector Taylor Series (VTS) method[62, 63] is related to the Algonquin method in that both use the Vector Taylor series to linearize the relationship between $\mathbf{y}$, $\mathbf{x}$, $\mathbf{n}$ and $\mathbf{h}$. The VTS method is in essence a model adaptation method but can be used to clean features as well as to update acoustic models. We will first discuss the method for the purpose of cleaning features.

The VTS method uses a point estimate of the noise process[3] as SS does, but it also

[3]The 0-th order version uses a point estimate for the noise, but the 1-st order version takes the variance of the noise process into account

	Subway	Car	Babble	Exhib.	Ave.
Clean	97.14	96.70	97.58	97.38	97.20
20 dB	91.65	82.65	94.30	87.04	88.91
15 dB	82.84	71.67	88.99	77.51	80.25
10 dB	68.19	56.23	79.39	61.74	66.39
5 dB	46.58	38.75	63.47	41.01	47.45
0 dB	23.79	18.95	35.82	21.69	25.06
-5 dB	11.18	9.55	13.93	10.89	11.39
Average	62.61	53.65	72.39	57.80	61.61

Table 7.2: Accuracy for Set A. Results for Spectral Subtraction. Noise level estimated from the first 20 frames.

uses a prior model for speech. Another difference is that we work in the log-spectrum domain.

The equation to find a point estimate for clean speech is

$$\hat{x} = y - \sum_{s=0}^{M-1} P(s|y) f(\mu_{x,s}, \mu_n, \mu_h), \tag{7.6}$$

where

$$f(x, n, h) = h + \log(1 + \exp(n - x - h)). \tag{7.7}$$

and $P(s|y)$ is the posterior probability of each component of the speech model. In the VTS method, the relationship between x, n, h and n is assumed to be an exact i.e. $y = x + h + \log(1 - \exp(n - x - h))$. We will explain how $P(s|y)$ is found in the next section.

Intuitively, this equation is similar to the cleaning equation of spectral subtraction i.e. Equation (7.5), but instead of subtracting a fixed noise vector, this method subtracts the "best" correction vector from the current observation. The correction vector is a weighted sum of component correction vectors $f(\mu_{x,k}, \mu_n, \mu_h)$, each of which corresponds to a speech center $\mu_{x,k}$ of the prior speech model. If the combination of a particular speech center $\mu_{x,k}$, noise μ_n and channel μ_h is a good explanation for the observed value y, then the score $P(s|y)$ will have a relatively high score (close to 1) and the resulting estimate $\hat{x}$ will be close to $\mu_{x,k}$.

Moreno evaluated MMSE VTS and found it to outperform CDCN[62] which in turn produces better results than SS[1]. Table 7.3 shows results for MMSE-VTS of order 0 and Table 7.4 shows results MMSE-VTS of order 1.

	Subway	Car	Babble	Exhibit	Average
Clean	98.99	99.12	99.05	99.26	99.11
20 dB	96.56	97.85	97.46	96.45	97.08
15 dB	92.08	95.47	94.04	91.70	93.32
10 dB	80.66	87.76	80.11	80.10	82.16
5 dB	58.00	65.15	51.54	54.83	57.38
0 dB	30.80	31.32	24.19	24.99	27.82
-5 dB	15.29	11.94	13.96	11.14	13.08
Average	71.62	75.51	69.47	69.61	71.55

Table 7.3: Accuracy for Set A. Results for 0th order VTS. A 256 mixture speech model was used. Noise level estimated from the first 20 frames.

7.2.2 Model Adaptation

Model domain methods alter the acoustic models of the recognizer, based on a model of the noise process. Methods that fall into this category include Parallel Model Combina-

	Subway	Car	Babble	Exhibit	Average
Clean	98.99	99.12	99.05	99.26	99.11
20 dB	97.27	97.61	98.18	97.38	97.61
15 dB	95.33	93.53	95.97	94.57	94.85
10 dB	87.75	79.23	85.77	86.36	84.78
5 dB	68.04	49.64	53.65	61.77	58.28
0 dB	37.61	20.41	22.79	26.32	26.78
-5 dB	15.93	5.35	10.20	10.43	10.48
Average	77.20	68.08	71.27	73.28	72.46

Table 7.4: Accuracy for Set A. Results for 1st order VTS. A 256 mixture speech model was used. Noise level estimated from the first 20 frames.

tion (PMC)[26] and the Vector Taylor Series (VTS) method[62, 3, 50]. Referring back to the block diagram of a speech recognition system in Figure 2.1, these methods swap in a different set of acoustic models.

Model Domain VTS

As noted above, the VTS method alters the models $p(x|s)$ so that they approximate $p(y|s)$. If we start with the model in Figure 4.2(b), we can write

$$p(y|s) = \int_{x,n,h} p(y|x,n,h)p(x|s)p(n)p(h). \tag{7.8}$$

The VTS method approximates $p(y|s)$ with a Gaussian distribution. As mentioned before, the relationship between x, n, h and n is assumed to be an exact i.e. $y = x + h + \log(1 - \exp(n - x - h))$ or

$$p(y|x,n,h) = \delta(y - (x + f(x,n,h))), \tag{7.9}$$

where $f(x,n,h) = h + \log(1 - \exp(n - x - h))$. The noise and channel models $p(n)$ and $p(h)$ are also replaced by delta functions $p(n) = \delta(n - n_0)$ and $p(h) = \delta(h - h_0)$, thus

$$p(y|s) = \int \delta(y - (x + f(x, n_0, h_0)))p_x(x|s)dx. \tag{7.10}$$

It is still not possible to perform the integral in Equation (7.10), due to the non-linear function $f(x, n_0, h_0)$. The approach taken in the VTS method (and Algonquin) is to linearize the f at x_0,n_0 and h_0 using the Vector Taylor Series

$$\begin{aligned} f_l(x,n,h) &= f(x_0,n_0,h_0) + \frac{d}{dx}f(x_0,n_0,h_0)(x - x_0) && (7.11)\\ &+ \frac{d}{dn}f(x_0,n_0,h_0)(n - n_0) + \frac{d}{dh}f(x_0,n_0,h_0)(h - h_0). && (7.12)\end{aligned}$$

Once this has been done, the integral can be taken without any difficulty.

Recall that $p(y|s)$ is approximated by a Gaussian. Hence, we instead find the mean and variance of $p(y|s)$ is

$$\mu_{y,s} = E(y) = \int y \int \delta(y - (x + f_l(x, n_0, h_0)))p(x|s)dxdy, \tag{7.13}$$

and

$$\Sigma_{y,s} = E(y^2) - \mu_{y,s}^2. \tag{7.14}$$

If we use only the first term in the Taylor series, then the parameters of $p(y|s)$ are

$$\mu_{y,k} = \mu_x + f(\mu_{x,k}, \mu_n, \mu_h), \tag{7.15}$$

and

$$\Sigma_y = \Sigma_x. \tag{7.16}$$

The updated acoustic models $p(\mathbf{y}|s)$ are used instead of the original acoustic models $p(\mathbf{x}|s)$. The acoustic models can contain multiple mixtures. In this case, the transformation is repeated for each mixture, independently of other mixtures.

The first order vector Taylor series can also be used, which results in slightly more complicated update equations for Σ_y[4]

$$\Sigma_y = (1 + F_x)\Sigma_x(1 + F_x)^T, \tag{7.17}$$

while μ_y remains the same. In Equation (7.17) $F_x = \frac{d}{dx}f(x_0, n_0, h_0)$ is the matrix derivative of f evaluated at x_0, n_0, h_0.

The VTS method requires an expansion point x_0, n_0, h_0. This expansion point is chosen to be the mode of the speech distribution μ_x, the mode of the noise distribution μ_n and the mode of the channel distribution μ_h.

For the MMSE version of VTS, we require $p(s|y)$, i.e. the weights used in Equation (7.6). These are found by

$$P(s|y) = \frac{p(y|s)P(s)}{p(y)}, \tag{7.18}$$

the mode of the channel model μ_h.

Acero et al.[3] compared the accuracy of VTS and PMC and found that VTS more accurately updated the model parameters of $p(y|s)$.

Since the MFC acoustic models are used and not log-spectrum models, the equations become slightly more complicated, and one has to transform the delta and acceleration models as well. This is form of the equations is reported in [3].

7.3 Discussion

Figure 7.1 shows the average results for the methods discussed in this chapter. The red line (circles) shows the results of running the recognizer directly on the noisy speech

[4]If n and h are not assumed to be point estimates, the update equation is $\Sigma_y = (1+F_x)\Sigma_x(1+F_x)^T + F_x\Sigma_n F_x^T + (1 + F_x)\Sigma_h(1 + F_x)^T$ as reported in [63]

files. The green (boxes) line shows the results for Spectral Subtraction, and the blue line (triangles) shows results for the Vector Taylor Series Method.

The first order VTS method performs considerably better than the Spectral Subtraction method. There are a number of differences between these two method, but an important difference is that the VTS cleaning algorithm uses a speech model, whereas SS does not. This result was predicted by the results in Chapter 3, where we saw that incorporating a speech model into the cleaning paradigm produced better results.

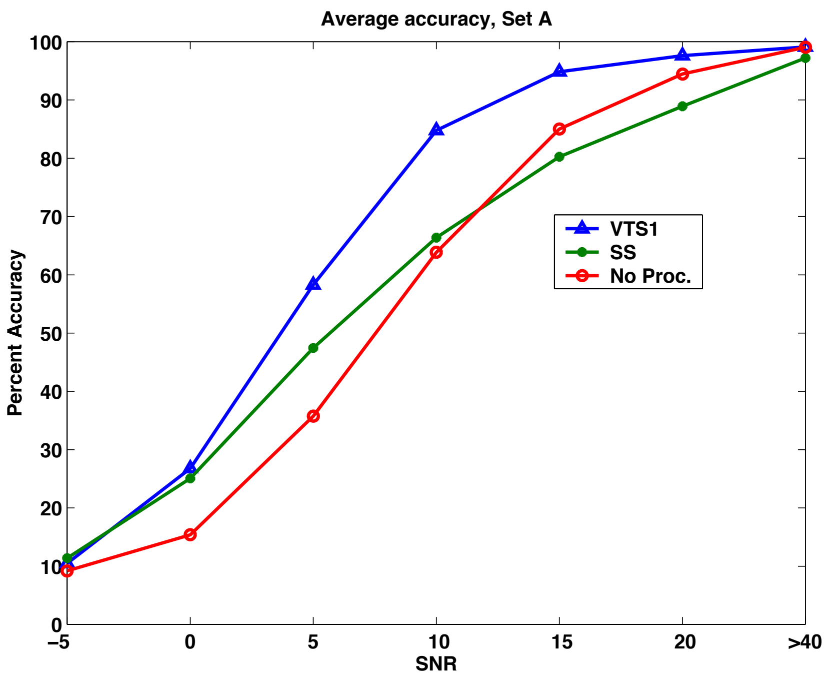


Figure 7.1: The plot shows results for the various cleaning methods discussed in this chapter. The red line (circles) shows the results of running the recognizer directly on the noisy speech files. The green (boxes) line shows the results for Spectral Subtraction, and the blue line (triangles) shows results for the Vector Taylor Series Method.

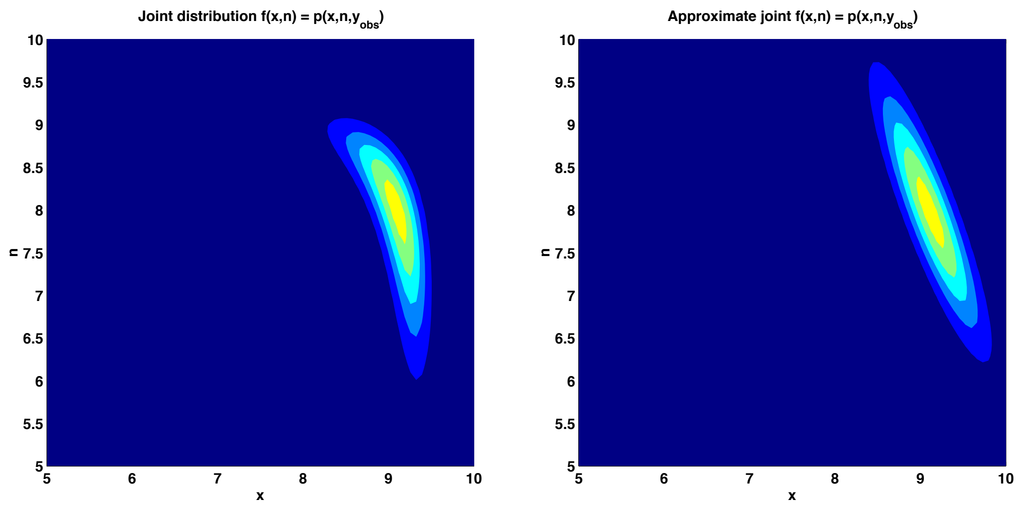
Chapter 8

The Algonquin Framework

In Chapter 4 we stated that from the probabilistic viewpoint, we wish to perform inference in the graph of Figure 4.3. In Chapter 5 we saw that the MFC transform involves taking the Log of the Mel-spectrum features before performing the DCT. The Log is a highly non-linear operation, and is the root of the computational hurdles, that need to be addressed in order to perform inference efficiently in this graphical model.

The Algonquin algorithm [23] produces a Gaussian approximation to the joint conditional distribution $p(\mathbf{x}, \mathbf{n}, \mathbf{h}|\mathbf{y}_{obs})$ (see Figure 8.1). As we saw in Chapter 6, this distribution is a mixture of components that cannot easily be used, e.g. when we wish to find a point estimate, due to the difficulty of integration. The Algonquin method allows us to work with a Gaussian approximation to the joint conditional distribution which in turn allows us to find point estimates of $\mathbf{x}, \mathbf{n}$ and $\mathbf{h}$ or approximate $p(s|\mathbf{y}_{obs})$.

The Algonquin method uses a linear approximation to the interaction equation. The leads to a Gaussian form for the Interaction likelihood which makes inference tractable. In this chapter we introduce the Algonquin method and discuss its performance.



(a) Non-linear joint distribution

(b) Linear joint distribution

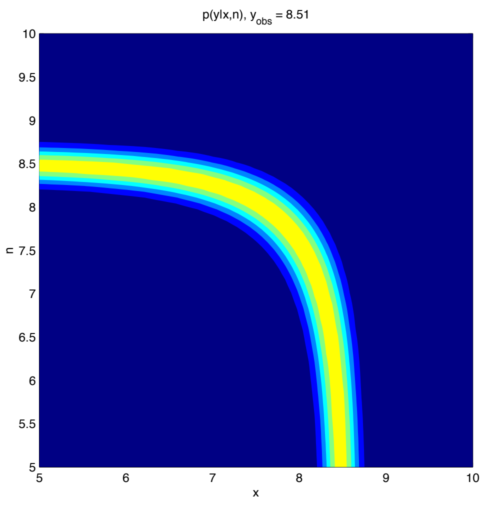
Figure 8.1: (a) Non-linear joint distribution for observed $y_{obs} = 9.37$. (b) Linear approximation to joint distribution found using the Algonquin algorithm.

8.1 The Algonquin Framework

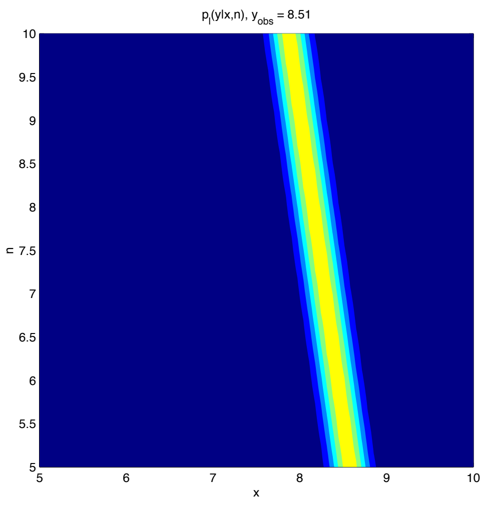
In the following sections we first discuss the linearization of the interaction equation and then go on to discuss the Algonquin framework and how it allows us to iteratively improve the linearization.

8.1.1 Linearization of the Interaction Likelihood

The Algonquin method uses a Taylor series linearization of the interaction equation. The non-linear interaction likelihood is shown in Figure 8.2(a) while the approximation to the non-linear likelihood is shown in Figure 8.2(b).



(a) Non-linear interaction likelihood



(b) Linear approximation

Figure 8.2: (a) Non-linear interaction likelihood. (b) Linear approximation to interaction likelihood.

Recall that the interaction equation in the log spectrum domain is

$$\mathbf{y} \approx \mathbf{g}\Big(\begin{bmatrix}\mathbf{x}\\ \mathbf{n}\\ \mathbf{h}\end{bmatrix}\Big) = \mathbf{x} + \mathbf{h} + \ln(\mathbf{1} + \exp(\mathbf{n} - \mathbf{x} - \mathbf{h})). \tag{8.1}$$

Define $\mathbf{z}^T = [\mathbf{x}^T \mathbf{n}^T \mathbf{h}^T]$. In order to linearize this equation using the first order Taylor series, we require the function evaluated at an expansion point $\mathbf{g}(\mathbf{z}_0)$, and it's derivative $\mathbf{G}(\mathbf{z}_0)$ at the same point

$$\mathbf{g}_l(\mathbf{z}) = \mathbf{g}(\mathbf{z}_0) + \mathbf{G}(\mathbf{z}_0)(\mathbf{z} - \mathbf{z}_0). \tag{8.2}$$

Taking the derivative of the $i-$th component of the vector g with respect to the $j-$th component of x_j we find

$$\frac{dg(x_i, n_i, h_i)}{dx_j} = \frac{1}{1 + \exp(n_i - x_i - h_i)} \tag{8.3}$$

if $i = j$ and 0 otherwise. Similarly, the derivative with respect to n_j is

$$\frac{dg(x_i, n_i, h_i)}{dn_j} = \frac{\exp(n_i - x_i - h_i)}{1 + \exp(n_i - x_i - h_i)}. \tag{8.4}$$

The derivative with respect to h_i is equal to the derivative with respect to x_i.

We define the $d \times d$ matrices, where d is the dimension of $\mathbf{x}$:

$$\mathbf{G}_x(\mathbf{x}, \mathbf{n}, \mathbf{h}) = \frac{dg(\mathbf{x}, \mathbf{n}, \mathbf{h})}{d\mathbf{x}} = diag\left[\frac{dg(x_1, n_1, h_1)}{dx_1} \dots \frac{dg(x_d, n_d, h_d)}{dx_d}\right], \tag{8.5}$$

and similarly for $\mathbf{G}_n$ and $\mathbf{G}_h$. $diag[\mathbf{x}]$ means that the elements of the vector $\mathbf{x}$ populate the diagonal of a diagonal matrix. For notational purposes we define the $3d \times d$ matrix:

$$\mathbf{G}(\mathbf{z}) = \frac{d\mathbf{g}(\mathbf{z})}{d\mathbf{z}} = \Big[\mathbf{G}_x(\mathbf{x}, \mathbf{n}, \mathbf{h}); \quad \mathbf{G}_n(\mathbf{x}, \mathbf{n}, \mathbf{h}); \quad \mathbf{G}_h(\mathbf{x}, \mathbf{n}, \mathbf{h})\Big] \tag{8.6}$$

Now we can finally write the linearized joint distribution, using the subscript l to differentiate it from the non-Gaussian joint distribution:

$$p_l(\mathbf{y}, \mathbf{z}, s) = N(\mathbf{y}; \mathbf{g}(\mathbf{z}_0) + \mathbf{G}(\mathbf{z}_0)(\mathbf{z} - \mathbf{z}_0), \mathbf{\Psi})\pi_s N(\mathbf{z}; \mu_s, \Sigma_s). \tag{8.7}$$

Note that the linearized joint distribution is a function of the linearization point $\mathbf{z}_0$.

8.1.2 The Posterior $q_{\mathbf{y}_{obs}}(\mathbf{z})$

The distribution of Equation (8.7) is Gaussian and we can potentially marginalize over any of its random variables. There are two problems with using p_l directly:

- The form of Equation (8.7) does not allows us to directly read off the mode of the distribution and the marginal $p(y_{obs}|s)$.
- p_l requires a linearization point $\mathbf{z}_0$, and using a poor choice for $\mathbf{z}_0$ can have a very negative effect on the results.

To solve the first problem we will rewrite p_l as q using an alternate parameterization. To address the second problem, we will iteratively update the Taylor series expansion point, which allows us to align the mode of the approximate posterior q to the true posterior p.

Our ultimate goal is to perform inference for a particular observation $\mathbf{y}_{obs}$. Let us therefore discuss the form of the function $q(\mathbf{z}|\mathbf{y}_{obs})$, i.e. the re-parameterized version of the linearized joint posterior function $p_l(\mathbf{z}|\mathbf{y}_{obs})$. The posterior is represented by a Gaussian mixture model

$$p_l(\mathbf{x},\mathbf{n},\mathbf{h}|\mathbf{y}_{obs}) = q(\mathbf{x},\mathbf{n},\mathbf{h}|\mathbf{y}_{obs}) = \sum_{s^x,s^n,s^h} q(s^x,s^n,s^h|\mathbf{y}_{obs})q(\mathbf{x},\mathbf{n},\mathbf{h}|s^x,s^n,s^h,\mathbf{y}_{obs}), \tag{8.8}$$

or using shorthand notation

$$q_{\mathbf{y}_{obs}}(\mathbf{z}) = \sum_s \rho_s q_{\mathbf{y}_{obs}}(\mathbf{z}|s), \tag{8.9}$$

where the subscript $\mathbf{y}_{obs}$ indicates dependence on the observations. The posterior mixing weights for classes s^x, s^n and s^h are $q_{\mathbf{y}_{obs}}(s^x,s^n,s^h) \approx p(s^x,s^n,s^h|\mathbf{y}_{obs})$. We use the shorthand $\rho_{s^xs^n,s^h}$ or ρ_s for $q_{\mathbf{y}_{obs}}(s^x,s^n,s^h)$.

The form of each mixture is

$$q_{\mathbf{y}_{obs}}(\mathbf{x},\mathbf{n},\mathbf{h}|s^x,s^n,s^h) = N\left(\begin{bmatrix}\mathbf{x}\\ \mathbf{n}\\ \mathbf{h}\end{bmatrix};\begin{bmatrix}\boldsymbol{\eta}^x_{s^xs^ns^h}\\ \boldsymbol{\eta}^n_{s^xs^ns^h}\\ \boldsymbol{\eta}^h_{s^xs^ns^h}\end{bmatrix},\begin{bmatrix}\mathbf{\Phi}^{xx}_{s^xs^ns^h} & \mathbf{\Phi}^{xn}_{s^xs^ns^h} & \mathbf{\Phi}^{xh}_{s^xs^ns^h}\\ \mathbf{\Phi}^{xn}_{s^xs^ns^h} & \mathbf{\Phi}^{nn}_{s^xs^ns^h} & \mathbf{\Phi}^{nh}_{s^xs^ns^h}\\ \mathbf{\Phi}^{xh}_{s^xs^ns^h} & \mathbf{\Phi}^{nh}_{s^xs^ns^h} & \mathbf{\Phi}^{hh}_{s^xs^ns^h}\end{bmatrix}\right), \tag{8.10}$$

or using shorthand notation

$$q_{\mathbf{y}_{obs}}(\mathbf{z}|s) = N(\mathbf{z};\boldsymbol{\eta}_s,\mathbf{\Phi}_s). \tag{8.11}$$

This form assumes that $\mathbf{x}, \mathbf{n}$ and $\mathbf{h}$ are jointly Gaussian. Compare the form of Equation (8.10) to the joint distribution in Equation (6.6) that we repeat here:

$$\begin{aligned} p(\mathbf{y}, \mathbf{z}|s) &= p(\mathbf{y}|\mathbf{x}, \mathbf{n}, \mathbf{h})p(\mathbf{x}|s^x)p(\mathbf{n}|s^n)p(\mathbf{h}|s^h) = \\ &= p(\mathbf{y}|\mathbf{x}, \mathbf{n}, \mathbf{h}) \cdot N\left(\begin{bmatrix}\mathbf{x}\\ \mathbf{n}\\ \mathbf{h}\end{bmatrix}; \begin{bmatrix}\boldsymbol{\mu}^x_{s^x s^n s^h}\\ \boldsymbol{\mu}^n_{s^x s^n s^h}\\ \boldsymbol{\mu}^h_{s^x s^n s^h}\end{bmatrix}, \begin{bmatrix}\boldsymbol{\Sigma}^x_{s^x s^n s^h} & 0 & 0\\ 0 & \boldsymbol{\Sigma}^n_{s^x s^n s^h} & 0\\ 0 & 0 & \boldsymbol{\Sigma}^h_{s^x s^n s^h}\end{bmatrix}\right). \end{aligned} \tag{8.12}$$

Note that the covariance matrix $\boldsymbol{\Phi}_s$ is now tri-diagonal (each component matrix $\boldsymbol{\Phi}^{xx}_{s^x s^n s^h}$ etc. is diagonal[1]). This reflects the fact that when $\mathbf{y}$ is observed, $\mathbf{x}, \mathbf{n}$ and $\mathbf{h}$ are no longer independent.

We now have three expressions for the posterior: the non-linear posterior $p(\mathbf{x}, \mathbf{n}, \mathbf{h}|\mathbf{y}_{obs})$, the linearized posterior $p_l(\mathbf{x}, \mathbf{n}, \mathbf{h}|\mathbf{y}_{obs})$, and the re-parameterized and (possibly factorized) posterior $q_{\mathbf{y}_{obs}}(\mathbf{x}, \mathbf{n}, \mathbf{h})$.

The linearized conditional joint distribution $p_l(\mathbf{x}, \mathbf{n}, \mathbf{h}|\mathbf{y}_{obs})$ and the un-factorized distribution $q_{\mathbf{y}_{obs}}(\mathbf{x}, \mathbf{n}, \mathbf{h})$ of Equation (8.8) are both Gaussian and can model the exact same distribution. However, the parameters of the two distributions are different. The mode of the posterior $p_l(\mathbf{x}, \mathbf{n}, \mathbf{h}|\mathbf{y}_{obs})$ is not coincident with the modes of the priors or the interaction likelihood. The parameters of $q_{\mathbf{y}_{obs}}(\mathbf{x}, \mathbf{n}, \mathbf{h})$ *are* coincident with the modes of $p_l(\mathbf{x}, \mathbf{n}, \mathbf{h}|\mathbf{y}_{obs})$. Recall that we linearized the interaction function "arbitrarily" at the modes of the noise and speech priors. Having found the mode of the posterior, we can use this mode as the new expansion point, as we will see in Section 8.1.4.

[1]In the log-spectrum domain, we can re-arrange the elements of vector $\mathbf{z}$ to be $[x_1, n_1, h_1, x_2, n_2, h_2, \ldots, x_d, n_d, h_d]$. The resultant $\boldsymbol{\Phi}$ matrix is then block diagonal, where each block is 3×3. This shows that finding its inverse of $\boldsymbol{\Phi}$ requires finding the inverse of d 3×3 matrices.

8.1.3 The Parameters of $q_{\mathbf{y}_{obs}}(\mathbf{z})$

We have now specified the form of a linearized version of the joint distribution $p_l(\mathbf{y}_{obs}, \mathbf{x}, \mathbf{y}, \mathbf{h})$ and an alternate parameterization $q_{\mathbf{y}_{obs}}(\mathbf{x}, \mathbf{y}, \mathbf{h})$ for the linearized posterior. However, we have not yet described how the parameters of $q_{\mathbf{y}_{obs}}$ are to be found.

Define $\mathbf{G} = \mathbf{G}(\mathbf{z}_0)$ and $\mathbf{g} = \mathbf{g}(\mathbf{z}_0)$. We can directly write each component of $p_l(\mathbf{x}, \mathbf{n}, \mathbf{h}, \mathbf{y}_{obs})$ by invoking matrix identity (A-10).

$$\begin{aligned} p_l(\mathbf{y}_{obs}, \mathbf{x}, \mathbf{n}, \mathbf{h}|s^x, s^n, s^h) &= p_l(\mathbf{y}_{obs}|\mathbf{x}, \mathbf{n}, \mathbf{h})p(\mathbf{x}|s^x)p(\mathbf{n}|s^n)p(\mathbf{h}|s^h) \\ &= N(\mathbf{y}_{obs}; \mathbf{g} - \mathbf{G}\mathbf{z}_0 + \mathbf{G}\mathbf{z}, \Psi)N(\mathbf{z}; \boldsymbol{\mu}_s, \boldsymbol{\Sigma}_s) \quad (8.13) \end{aligned}$$

can be written as

$$q(\mathbf{y}_{obs}|s)q(\mathbf{z}|\mathbf{y}_{obs}, s) = \gamma_s N(\mathbf{z}; \boldsymbol{\eta}_s, \boldsymbol{\Phi}_s), \quad (8.14)$$

where the mixture mode is

$$\boldsymbol{\eta}_s = \boldsymbol{\Phi}_s\big[\boldsymbol{\Sigma}_s{}^{-1}\boldsymbol{\mu}_s + \mathbf{G}^T\boldsymbol{\Psi}^{-1}(\mathbf{y}_{obs} - \mathbf{g} + \mathbf{G}\mathbf{z}_0)\big], \quad (8.15)$$

and the covariance matrix is

$$\boldsymbol{\Phi}_s = (\boldsymbol{\Sigma}_s{}^{-1} + \mathbf{G}^T\boldsymbol{\Psi}^{-1}\mathbf{G})^{-1}, \quad (8.16)$$

and

$$\begin{aligned} \gamma_s = (2\pi)^{\frac{d_z - d_y}{2}} &|\boldsymbol{\Sigma}_s|^{-1/2}|\boldsymbol{\Psi}|^{-1/2}|\boldsymbol{\Phi}_s|^{1/2} \\ &\exp\big[-\frac{1}{2}\big(\boldsymbol{\mu}_s^T\boldsymbol{\Sigma}_s{}^{-1}\boldsymbol{\mu}_s + (\mathbf{y}_{obs} - \mathbf{g} + \mathbf{G}\mathbf{z}_0)^T\boldsymbol{\Psi}^{-1}(\mathbf{y}_{obs} - \mathbf{g} + \mathbf{G}\mathbf{z}_0) \\ &\qquad - \boldsymbol{\eta}_s^T\boldsymbol{\Phi}_s{}^{-1}\boldsymbol{\eta}_s)\big)\big]. \quad (8.17) \end{aligned}$$

Note that $\gamma_s = p_l(\mathbf{y}_{obs}|s)$. In order to find the weights of the components of the posterior $q_{\mathbf{y}_{obs}}$ we multiply by the priors and normalize

$$\rho_s = p_l(s|\mathbf{y}_{obs}) = \frac{p_l(\mathbf{y}_{obs}|s)p(s)}{\sum_i p_l(\mathbf{y}_{obs}|i)p(i)} = \frac{\gamma_s \pi_s}{\sum_i \gamma_i \pi_i}. \quad (8.18)$$

To summarize,

$$p_l(\mathbf{x}, \mathbf{n}, \mathbf{h}|\mathbf{y}_{obs}) = q_{\mathbf{y}_{obs}}(\mathbf{z}) = \sum_s \rho_s N(\mathbf{z}; \boldsymbol{\eta}_s, \boldsymbol{\Phi}_s), \tag{8.19}$$

where ρ_s, $\boldsymbol{\eta}_s$ and $\boldsymbol{\Phi}_s$ are defined above.

8.1.4 Updating the Linearization Point

As was noted previously, the Taylor series is expanded at $\mathbf{z}_0$. This point needs to be determined somehow. If we use the previously calculated $\boldsymbol{\eta}_s$ as an expansion point and iterate the above equations, then the $\boldsymbol{\eta}_s$ will converge to a mode of the true posterior. In some special cases, $\boldsymbol{\eta}_s$ will oscillate around the true mode. We will discuss the convergence behavior below.

Define $\boldsymbol{\eta}^{(0)_s}$ as the initial expansion point, and $\boldsymbol{\eta}_s^{(i)}$ as the value of $\boldsymbol{\eta}_s$ after i iterations. Also, $\mathbf{G}(\mathbf{z}_0)$ and $\mathbf{g}(\mathbf{z}_0)$ are functions of the expansion point. Define $\mathbf{G}^{(i)} = \mathbf{G}(\boldsymbol{\eta}_s^{(i)})$ and $\mathbf{g}^{(i)} = \mathbf{g}(\boldsymbol{\eta}_s^{(i)})$.

We can now write the iterative formula for the mode $\boldsymbol{\eta}_s^{(i)}$ each mixture s

$$\begin{aligned}\boldsymbol{\eta}_s^{(i+1)} = \left(\boldsymbol{\Sigma}_s^{-1} + \mathbf{G}^{(i)T}\boldsymbol{\Psi}^{-1}\mathbf{G}^{(i)}\right)^{-1} \cdot \\ \left[\boldsymbol{\Sigma}_s^{-1}\boldsymbol{\mu}_s + \mathbf{G}^{(i)T}\boldsymbol{\Psi}^{-1}(\mathbf{y}_{obs} - \mathbf{g}^{(i)} + \mathbf{G}^{(i)}\boldsymbol{\eta}_s^{(i)})\right],\end{aligned} \tag{8.20}$$

or, alternately (add $\boldsymbol{\Sigma}_s^{-1}\boldsymbol{\eta}_s^i - \boldsymbol{\Sigma}_s^{-1}\boldsymbol{\eta}_s^i$ term and re-arrange) to get

$$\boldsymbol{\eta}_s^{(i+1)} = \boldsymbol{\eta}_s^{(i)} + \boldsymbol{\Phi}_s^{(i)}\left[\boldsymbol{\Sigma}_s^{-1}(\boldsymbol{\mu}_s - \boldsymbol{\eta}_s^{(i)}) + \mathbf{G}^{(i)}\boldsymbol{\eta}_s^{(i)}\boldsymbol{\Psi}^{-1}(\mathbf{y} - \mathbf{g}^{(i)})\right]. \tag{8.21}$$

Notice that this equation represents a "tug-of-war" between the priors $\boldsymbol{\mu}$ and the observation $\mathbf{y}_{obs}$. The term $\boldsymbol{\Sigma}^{-1}(\boldsymbol{\mu} - \boldsymbol{\eta}^{(i)})$ pulls the mode of the posterior towards the prior, while the term $\mathbf{G}^{(i)}\boldsymbol{\eta}^{(i)}\boldsymbol{\Psi}^{-1}(\mathbf{y} - \mathbf{g}^{(i)})$ pulls towards the observation. Similarly, the covariance matrix for iteration i is

$$\boldsymbol{\Phi}_s^{(i)} = \left[\boldsymbol{\Sigma}_s^{-1} + \mathbf{G}(\boldsymbol{\eta}_s^{(i)})^T\boldsymbol{\Psi}^{-1}\mathbf{G}(\boldsymbol{\eta}_s^{(i)})\right]^{-1}. \tag{8.22}$$

```
• Initialization:
    – $\mathbf{z}_0^{(0)} \leftarrow \boldsymbol{\mu}_s$
• For each mixture $s$, iteratively update of Taylor series
  expansion point:
    – calculate $\mathbf{g}(\mathbf{z}_0^{(i)})$ according to Equation (8.2)
    – calculate $\mathbf{G}(\mathbf{z}_0^{(i)})$ according to Equation (8.6)
    – calculate $\boldsymbol{\Phi}_s^{(i)}$ according to Equation (8.22)
    – calculate $\boldsymbol{\eta}_s^{(i)}$ according to Equation (8.21)
    – $\mathbf{z}_0^{(i+1)} \leftarrow \boldsymbol{\eta}_s^{(i)}$
• Calculate Mixture weights.
    – calculate $q_y(s)$ according to Equation (8.23)
```

Figure 8.3: The Algonquin Algorithm.

This equation represents the combination of the variance of the prior $\boldsymbol{\Sigma}$ and the variance of interaction likelihood $\boldsymbol{\Psi}$.

As shown in Figure 8.3, we iteratively evaluate Equations (8.21) and (8.22) until convergence. Once the mixture component means and their variances have been iteratively refined, we need to find the weights of the individual mixtures. We use the equivalent but more efficient form (see derivation in Appendix C.1) of the equation for the mixture weights:

$$q_{y_{obs}}(s) = \rho_s = \frac{\exp(L_s)}{\sum_j \exp(L_j)} \tag{8.23}$$

where[2]

$$\begin{aligned} L_s &= \ln \pi_s - \frac{1}{2}\ln|2\pi\boldsymbol{\Sigma}_s| + \frac{1}{2}\ln|2\pi\boldsymbol{\Phi}_s| \\ &- \frac{1}{2}\left\{(\boldsymbol{\mu}_s - \boldsymbol{\eta}_s)^T\boldsymbol{\Sigma}_s^{-1}(\boldsymbol{\mu}_s - \boldsymbol{\eta}_s)\right\} \\ &- \frac{1}{2}\left[(\mathbf{y} - \mathbf{g})^T\boldsymbol{\Psi}^{-1}(\mathbf{y} - \mathbf{g})\right]. \end{aligned} \tag{8.24}$$

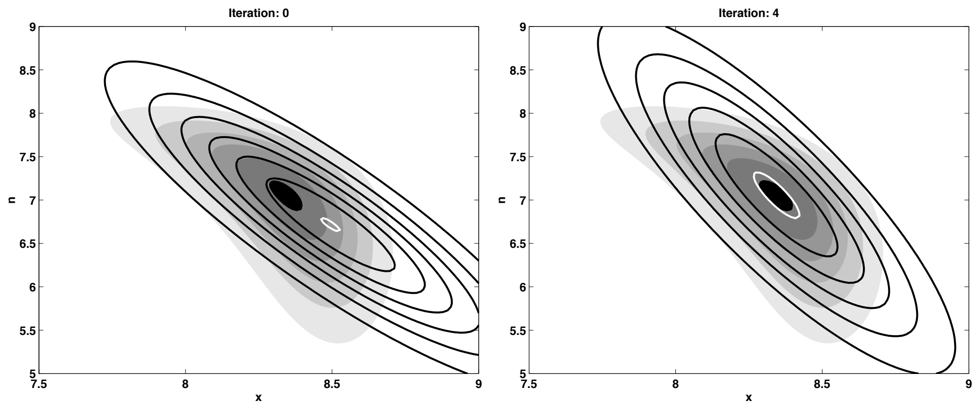
If we wish to use q for feature cleaning, the ρ_s are interpreted as mixture weights of the GMM. If we wish to use the soft information paradigm we let the class variables correspond to the states of the HMM and use the $\log(q_{y_{obs}}(s))$ directly, as we will see in Chapter 10.

8.1.5 Convergence Properties of the Algorithm

As was noted before, the expansion point of the Taylor Series approximation is iteratively improved through the use of the posterior means $\boldsymbol{\eta}_s$ as expansion points in subsequent iterations. Intuitively, it is reasonable that we should linearize around the mode of the true posterior, as this should lead to the minimum error in evaluation of marginal distributions. It can be shown that once the algorithm has found the true mode of the posterior, it will not deviate from that mode. Figure 8.4 shows how the approximate linear posterior converges from the initial expansion point to the true mode in 4 iterations.

Empirical studies of the convergence of the approximate posterior modes $\boldsymbol{\eta}_s$ show that in some cases the approximate mode repeatedly overshoots the true mode in a back and forth convergence trajectory. This behavior is more likely to be observed when the speech noise combination is a poor fit to the observed noisy speech feature $\mathbf{y}_{obs}$.

[2]This is Equation (A-56) after assuming that $\boldsymbol{\eta}_s^{(i)} = \boldsymbol{\eta}_s^{(i-1)}$ i.e. $\mathbf{z}_0 = \boldsymbol{\eta}_s$.



(a) Iteration 0 (b) Iteration 4

Figure 8.4: Plot (a) shows the true posterior and the approximate posterior for iteration 0, i.e. when the Taylor series is expanded at the speech and noise prior means. Plot (b) shows how the approximate posterior has aligned to the mode of the true posterior after 4 iterations.

It is possible to introducing a damping factor into the update equation for $\boldsymbol{\eta}_s$ in Equation (8.21) to reduce the overshoot and ensure convergence. However, recognition results show that it only takes 2-3 iterations for the recognition accuracy reach its maximum using the undamped algorithm. This indicates that convergence for the important components of the approximate posterior is fast and accurate. Figure 8.5 shows the average accuracy for the Aurora data set B (see Section 8.2) as a function of iteration for the standard algorithm and the algorithm with a damping factor of 0.5. Notice that the algorithm has reached its maximum of 85.67% in just 2 iterations, while the damped algorithm takes longer to converge. The recognition rate jumps from 78.41% when the prior means are used as an expansion point to 85.63% after the first update.

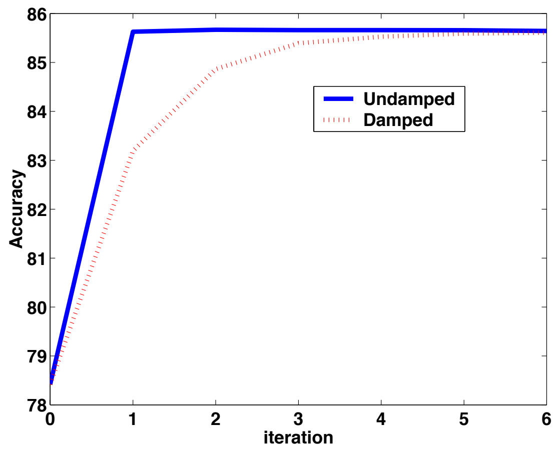


Figure 8.5: Solid line shows the accuracy as a function of iteration for Algonquin. Dashed line shows accuracy for Algonquin with damping factor.

8.1.6 Variational Inference

We now discuss an alternate way of deriving the update equations for q, using the Variational framework. This view will be required for the discussion in the next Chapter.

The basic idea of variational inference[44] is to construct a function q that is a simplified version of the distribution we are actually interested in i.e. p, and minimize the discrepancy between these two functions.

When $\mathbf{y}$ is observed, the other random variables, $\mathbf{x}$, $\mathbf{n}$, $\mathbf{h}$, s^x, s^n, s^h are no longer independent, which can make inference computationally expensive. A potential benefit of the Variational method is to allow us to work with a factorized version of q. This comes at the cost of increased approximation error, and reduced recognition accuracy.

The goal of variational inference [43, 10] is to minimize the relative entropy

(Kullback-Leibler divergence) between q and p

$$\mathcal{K} = \sum_{\{s^x,s^n,s^h\}} \int_{\{\mathbf{x},\mathbf{n},\mathbf{h}\}} q_{y_{obs}}(\mathbf{x},\mathbf{n},\mathbf{h},s^x,s^n,s^h) \cdot \ln \frac{q_{y_{obs}}(\mathbf{x},\mathbf{n},\mathbf{h},s^x,s^n,s^h)}{p(\mathbf{x},\mathbf{n},\mathbf{h},s^x,s^n,s^h|\mathbf{y}_{obs})}. \quad (8.25)$$

The KL distance is usually defined as $\int_x p(x) \log p(x)/q(x)$. The advantage of the above formulation of the KL distance is that we can use a factorized q function that allows more efficient inference.

8.1.7 The Negative Relative Entropy $\mathcal{F}$

Notice that the denominator in Equation (8.25) is the posterior, whereas we have an expression for the joint distribution in Equation(6.5). Herein lies one of the advantages of this cost function, because minimizing $\mathcal{K}$ is equivalent to maximizing

$$\mathcal{F} = \ln p(\mathbf{y}) - \mathcal{K} = \sum_{\{s^x,s^n,s^h\}} \int_{\{\mathbf{x},\mathbf{n},\mathbf{h}\}} q_{y_{obs}}(\mathbf{x},\mathbf{n},\mathbf{h},s^x,s^n,s^h) \cdot \ln \frac{p(\mathbf{x},\mathbf{n},\mathbf{h},s^x,s^n,s^h,\mathbf{y}_{obs})}{q_{y_{obs}}(\mathbf{x},\mathbf{n},\mathbf{h},s^x,s^n,s^h)}. \quad (8.26)$$

If we plug in the expressions for $p_l(\mathbf{x},\mathbf{n},\mathbf{h},s^x,s^n,s^h,\mathbf{y}_{obs})$ and $q_{\mathbf{y}_{obs}}(\mathbf{x},\mathbf{n},\mathbf{h},s^x,s^n,s^h)$ into equation (8.26), and simplify, we arrive at an expression for the free energy (see Appendix C.1),

$$\begin{aligned}
\mathcal{F} = & -\frac{1}{2}\sum_s \rho_s \ln|2\pi\mathbf{\Psi}| \\
& -\sum_s \rho_s\Big((\mathbf{y}_{obs} - \mathbf{g}(\mathbf{z}_0) + \mathbf{G}(\mathbf{z}_0)(\mathbf{z}_0 - \boldsymbol{\eta}_s))^T \\
& \qquad \mathbf{\Psi}^{-1}((\mathbf{y}_{obs} - \mathbf{g}(\mathbf{z}_0) + \mathbf{G}(\mathbf{z}_0)(\mathbf{z}_0 - \boldsymbol{\eta}_s)\Big) & (8.27) \\
& +\sum_s \rho_s Tr[\mathbf{G}(\mathbf{z}_0)^T\mathbf{\Psi}^{-1}\mathbf{G}(\mathbf{z}_0)\mathbf{\Phi}_s] & (8.28) \\
& +\sum_s \rho_s \ln \pi_s & (8.29) \\
& -\frac{1}{2}\sum_s \rho_s \ln|2\pi\mathbf{\Sigma}_s| \\
& -\frac{1}{2}\sum_s \rho_s \left\{(\boldsymbol{\mu}_s - \boldsymbol{\eta}_s)^T\mathbf{\Sigma}_s^{-1}(\boldsymbol{\mu}_s - \boldsymbol{\eta}_s) + Tr[\mathbf{\Sigma}_s^{-1}\mathbf{\Phi}_s]\right\} & (8.30) \\
& -\sum_s \rho_s \ln \rho_s & (8.31) \\
& +\frac{1}{2}\sum_s \rho_s \ln|2\pi\mathbf{\Phi}_s| - 3d. & (8.32)
\end{aligned}$$

The derivation of this equation is given in Appendix C.1. To find the estimation formulas for the parameters of q (i.e. the means $\boldsymbol{\eta}_s$ and variances $\mathbf{\Phi}_s$ and the mixtures weights $q_{\mathbf{y}}(s)$), we maximize $\mathcal{F}$ by differentiating with respect to the parameters and equating to zero. When q is un-factorized, this leads to the same equations as we found before. The derivations can be found in Appendix C.1.

8.2 MMSE Algonquin Results on Aurora Database

In order to assess the effectiveness of the the Algonquin method we apply it in the feature cleaning paradigm. Once we have estimated the approximate posterior q we find the

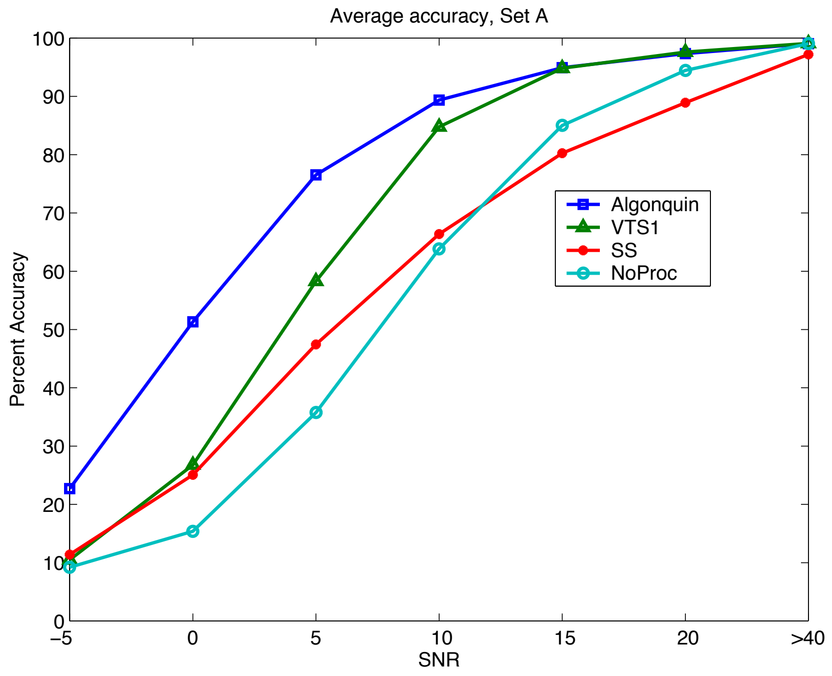


Figure 8.6: The figure shows the a comparison of the Algonquin methods to the VTS1 and SS.

MMSE estimate of the clean speech[16]

$$\hat{\mathbf{x}} = \int \mathbf{x} p(\mathbf{x}|\mathbf{y}_{obs}) d\mathbf{x} \tag{8.33}$$

$$\approx \int \mathbf{x} q(\mathbf{x}) d\mathbf{x} = \int \mathbf{z}^x \sum_s q(s) q(\mathbf{z}^x|s) d\mathbf{z}^x = \sum_s \rho_s \boldsymbol{\eta}_s^x, \tag{8.34}$$

where $\mathbf{z}^x$ is the x component of $\mathbf{z}$. The cleaned speech vector $\hat{\mathbf{x}}$ is simply the weighted sum of the mixture means of q. We give results for the using the Algonquin method on set A of the Aurora data set [37]. Table 8.1 shows the recognition accuracy when using the Algonquin in the MMSE paradigm.

The noise model contained a single Gaussian, which was estimated from the first 20 frames of each file.

The average accuracy over all conditions is 82.22%. This is a reduction in relative word error rate over spectral subtraction [7] of 60.35% and an reduction of 34.45% over the first order Vector Taylor Series method (see Figure 8.6).

The increased recognition accuracy of Algonquin over VTS may be attributed to a few factors. First is that we used a distribution for the interaction likelihood. This may have some advantages since we take into account the uncertainty in the relationship between $\mathbf{x}, \mathbf{n}, \mathbf{h}$ and $\mathbf{y}$.

The other major difference between Algonquin and VTS is that Algonquin adjusts the approximation for *each* observation $\mathbf{y}_{obs}$, whereas VTS finds a general distribution over all $\mathbf{y}$ i.e. $f(\mathbf{y}) = p(\mathbf{y}|s^x)$.

8.3 Effect of Speech Model Size

The computational complexity of the algorithm scales linearly with the number of mixtures in the speech model $p(\mathbf{x})$. One motivation for using the feature cleaning paradigm rather than the model adaptation paradigm or soft information paradigm is that the computation complexity can be much lower. If we use the model adaptation paradigm, we need to update the distributions of each and every acoustic model. A large vocabulary speech recognizer uses tens of thousands of Gaussians [3]. Thus we would like to know how the algorithm performs as a function of the size of the speech model. Figure 8.7 shows performance for a speech model of 4, through 256 mixtures. In the case of 4 through 256 mixtures, the models were trained directly on the clean speech training sets. Notice that the accuracy has levelled off at around 64 mixtures, which is about 1/10 of the mixtures used in the recognizer.

[3]The standard Aurora recognizer uses only 552 Gaussian mixtures

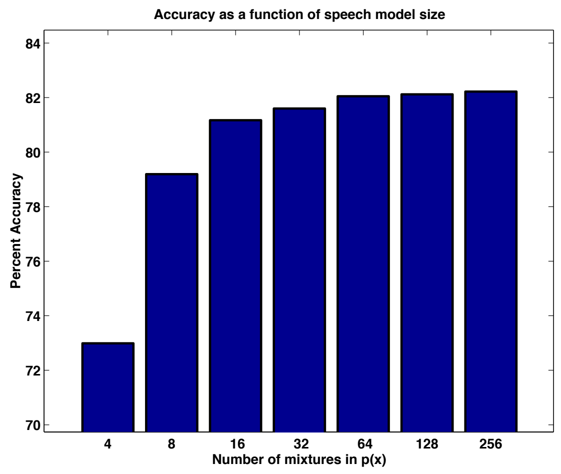
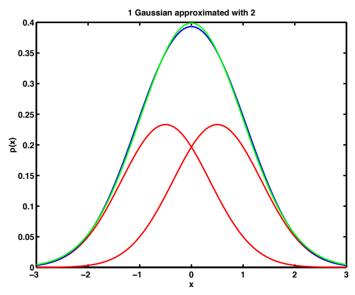


Figure 8.7: Accuracy as a function of number of components in the speech model $p(x)$.

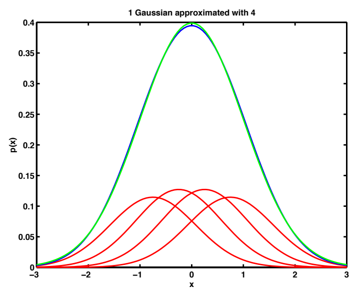
8.4 Effect of Noise Model Size

In Figure 6.1 we saw that the variance of the noise distribution is larger than the variance of the speech distribution. Figure 8.2 shows how we approximated the non-linear interaction likelihood with a linearized interaction likelihood. If we approximate the noise distribution with multiple mixtures (see Figure 8.8), we have effectively expanded the approximation of the likelihood at multiple points, and therefore done a better job of approximating the curving posterior distribution. The result of doing this is shown in Figure 8.9.

When using two mixtures, the Word Error Rate is reduced by 9.3% and 10.0% when four mixtures are used (see Tables D.10 and D.11). Notice that we have not improved how well the true noise is modelled, since we estimate a single mean and variance of the noise from the first 20 frames. The reduction in word error rate is due to increasing the precision of the approximation.



(a) 2 mixture approximation



(b) 4 mixture approximation

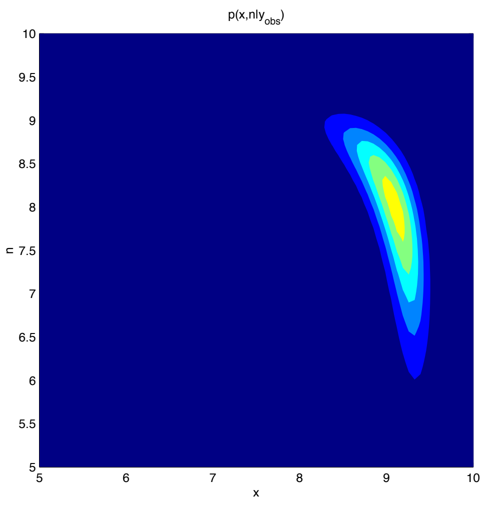
Figure 8.8: (a) Approximation of a single Gaussian with a mixture of 2 Gaussians. (b) Approximation of a single Gaussian with a mixture of 4 Gaussians.

8.5 Using Factorized Versions of q

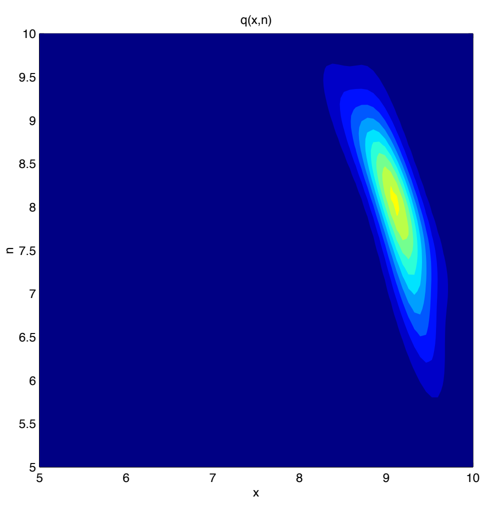
For some probabilistic models, the variational framework allows one to use a q distribution that is more computationally efficient to work with. Generally, one can increase computational efficiency when computing marginals of a joint distribution by factoring that distribution.

Figure 8.10 shows two factorizations that were investigated. The first factorization, shown in Figure 8.10(b) involves discarding the the link between n and x. This is equivalent to forcing each component of the q distribution to be an axis aligned Gaussian. We can achieve this by dropping the diagonal matrices of the Covariance distribution of q in Equation (8.10). This leads to savings in the computation of the Covariance matrix in Equation (8.22). However, the number of component factors in the q distribution remains the same. A large part of the computational burden is in computing g and its derivatives, in Equation (8.2).

Figure 8.10(c) shows a different factorization that involves constraining the modes of



(a) Non-linear posterior



(b) 2 mixture approximation

Figure 8.9: (a) The non-linear posterior. (b) Linear approximation to using 2 mixtures to represent noise model.

the q distribution. The number of free parameters is reduced from $|s^x| \times |s^n|$ to $|s^x| + |s^n|$ (where $|s^x|$ is the number of states or classes in the speech model). Despite this, $\mathbf{g}$ and its derivatives have to be computed as often as before. These two factorizations therefore do not lead to significant computational savings. They do, however, adversely effect recognition performance.

8.6 Discussion

In this chapter we introduced the Algonquin algorithm, which allows us approximate the joint conditional distribution $p(\mathbf{x}, \mathbf{n}, \mathbf{h}|\mathbf{y}_{obs})$.

We saw that this is a very effective method for feature cleaning and outperforms its

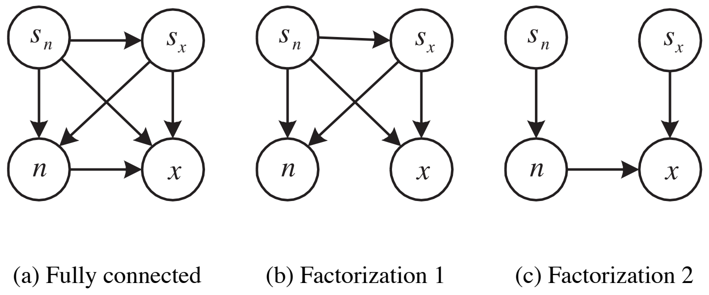


Figure 8.10: Different q function factorizations

closest cousin, the VTS method by a substantial margin. It was argued that this was due to the better approximation allowed by adapting to each observation individually.

The idea of defining an interaction likelihood and then constructing a tractable approximation is novel in itself and suggests other possible methods such as using Gaussian basis functions to approximate the interaction likelihood [67], or efficient sampling.

We saw that increasing the size of the speech model of the cleaning algorithm was helpful, but levelled off when the number of components reached 1/10th of the models size of the recognizer. We also saw that the Gaussian approximation could be enhanced by increasing the number of components in the noise model.

In this chapter, we used a noise model that was estimated from the first 20 frames of the speech file. In the next chapter we introduce a method for learning the parameters of the noise and channel models.

	Subway	Car	Babble	Exhib.	Ave.
Clean	98.93	99.12	98.99	99.32	99.09
20 dB	96.13	97.88	98.36	97.01	97.34
15 dB	92.54	95.65	97.08	94.57	94.96
10 dB	84.80	90.21	93.32	89.39	89.43
5 dB	68.74	75.15	84.16	80.07	77.03
0 dB	43.63	46.07	59.53	60.07	52.33
-5 dB	18.42	16.69	26.69	33.14	23.73
Average	77.17	80.99	86.49	84.22	82.22

Table 8.1: Accuracy for Set A. MMSE-Algonquin Algorithm after 2 iterations. The speech model contained 256 mixtures. A single component noise model was used which was estimated from the first 20 frames of each file.

Chapter 9

Learning Environmental Parameters

In the preceding chapters, we used a simple way to estimate the parameters of the noise model, we simply took the first 20 frames of each file and calculated a mean and variance. This works because there is a short pause before the onset of speech in each file, but the method gives a sub-optimal estimate of the parameters of the noise model, and is not applicable to real world applications.

It is clear that the better the noise and channel models are, the better we expect a RASR method will perform, e.g. if we knew the exact noise signal, we could simply subtract it off. In Chapter 8 we saw that increasing the number of components in the GMM noise model enhances the precision of the approximation. Increasing the number of components also allows us to model the noise more precisely, and thus reduce the classification ambiguity. For non-stationary noise and noise that does not manifest itself as a Gaussian in the log-spectrum domain, we expect that using a multi-component noise model could be beneficial.

We can get an estimate of the upper limit of performance by providing the algorithm with a "perfect" estimate. This estimate can be found from the noise component alone of a noisy speech file. If we do this using a single component noise model for Set A, we get a WER of 14.04% (85.96% accuracy) which is a reduction in WER from the baseline WER of 17.78% (82.22% accuracy) of 21.03%. Thus, we would like a better way to estimate the parameters of the noise and channel models.

For real world applications, we could use a speech/non-speech classifier to find frames for use in the noise estimate[7, 75]. A disadvantage of this approach is that any classifier is prone to errors, and in addition, we discard information about the noise contained in frames that also contain speech. Alternatively, it is possible to update the parameters without the use of an explicit speech/non-speech detector[1, 5].

In this chapter we discuss how the parameters of the noise and channel models can be estimated from the complete data, using a Generalized EM method[24, 52].

9.1 Joint Learning of Noise and Channel Distortion

Recall that the goal of variational inference is to minimize the relative entropy[10] (Kullback-Leibler divergence) between q and p_l

$$\mathcal{K} = \sum_{\{s^x, s^n, s^h\}} \int_{\{\mathbf{x},\mathbf{n},\mathbf{h}\}} q(\mathbf{x}, \mathbf{n}, \mathbf{h}, s^x, s^n, s^h) \cdot \ln \frac{q(\mathbf{x}, \mathbf{n}, \mathbf{h}, s^x, s^n, s^h)}{p_l(\mathbf{x}, \mathbf{n}, \mathbf{h}, s^x, s^n, s^h | \mathbf{y}_{obs})}. \tag{9.1}$$

Before we used this loss function to learn the parameters of q. We will now use it to learn the parameters of p When we estimate the parameters of the noise and channel models, we assume that the noise is stationary, and that we can use all the frames of a speech file[1]. Thus, when evaluating the negative relative entropy we sum over observation frames also.

[1]Notice that this is not the same as trying to adapt to the noise conditions online, e.g. using a window of preceding frames or a forgetting factor[5].

In addition, we assume that $\ln p_l(\mathbf{Y}) \approx \ln p(\mathbf{Y})$. Writing the negative relative entropy as

$$\mathcal{F} = \ln p_l(\mathbf{Y}) - \mathcal{K} = \sum_t \sum_{\{s^x,s^n,s^h\}} \int_{\{\mathbf{x},\mathbf{n},\mathbf{h}\}} q(\mathbf{x},\mathbf{n},\mathbf{h},s^x,s^n,s^h) \cdot \ln \frac{p_l(\mathbf{x},\mathbf{n},\mathbf{h},s^x,s^n,s^h,\mathbf{y})}{q(\mathbf{x},\mathbf{n},\mathbf{h},s^x,s^n,s^h)}, \quad (9.2)$$

we see that it is comprised of two components; $\ln p_l(\mathbf{Y})$ expresses how well the noisy data is modelled and $\mathcal{K}$ expresses how well q matches p_l. $\mathcal{K}$ is always positive or 0 and will only be 0 if p_l exactly matches q. If we maximize $\mathcal{F}$ with respect to the parameters of p_l, and assume that $\mathcal{K}$ remains constant then we are maximizing a lower bound on $\ln p_l(\mathbf{Y})$

$$\mathcal{F} \leq \ln p_l(\mathbf{Y}) \approx \ln p(\mathbf{Y}). \quad (9.3)$$

In words, we are maximizing a lower bound on the log-probability of the observed data under the model. Clearly, $\mathcal{K}$ will increase if the parameters of p_l are changed, because we are pulling p_l away from q. As a consequence, for each iteration of the algorithm, $\ln p_l(\mathbf{Y})$ increases more than the increase in $\mathcal{F}$, i.e. it will increase by $\Delta\mathcal{F} + \Delta\mathcal{K}$.

9.1.1 A Generalized EM Method for Parameter Adaptation

We can learn the parameters of the noise and speech model using a generalized EM method[65]. The algorithm alternates between:

1. Updating the variational parameters $\rho^{(t)}_{s^x s^n s^h}$, $\eta^{(t)}_{s^x s^n s^h}$, $\boldsymbol{\Sigma}^{(t)}_{s^x s^n s^h}$ for each frame $t = 1,\ldots,T$, as discussed in Chapter 8, and
2. Maximizing $\mathcal{F}$ with respect to the noise model parameters π^n, $\boldsymbol{\mu}^n$ and $\boldsymbol{\Sigma}^n$ and channel model parameters π^h, $\boldsymbol{\mu}^h$ and $\boldsymbol{\Sigma}^h$.

The complete algorithm is shown in Figure 9.7. In the next two sections we derive the re-estimation formulas for the parameters of $p(\mathbf{n})$ and $p(\mathbf{h})$.

9.1.2 Learning $p(\mathbf{n})$

To re-estimate the parameters of the models, we can use a generalized EM algorithm. Recall the form of $q(\mathbf{z}^{(t)}, s)$ and $p(\mathbf{z}^{(t)}, s, \mathbf{y}^{(t)})$ where we use s as a shorthand for $\{s_x, s_n, s_h\}$:

$$q(\mathbf{z}^{(t)}, s) = \sum_s \rho_s^{(t)} N(\mathbf{z}^{(t)}; \eta_s^{(t)}, \Phi_s^{(t)}) \tag{9.4}$$

$$p(\mathbf{z}^{(t)}, s, \mathbf{y}^{(t)}) = \sum_s N(\mathbf{y}^{(t)}; g(\mathbf{z}^{(t)}), \Psi)\pi_s N(\mathbf{z}^{(t)}; \mu_s, \Sigma_s). \tag{9.5}$$

The algorithm derived in in Chapter 8 corresponds to the E step in the Generalized EM algorithm. The M step involves finding π^n, $\boldsymbol{\mu}^n$ and $\boldsymbol{\Sigma}^n$ that maximize the negative relative entropy

$$\begin{aligned}\{\hat{\pi}_n, \hat{\boldsymbol{\mu}}_n, \hat{\boldsymbol{\Sigma}}_n\} &= \operatorname*{argmax}_{\pi,\boldsymbol{\mu},\boldsymbol{\Sigma}} \sum_t \sum_{s_x,s_n,s_h} \int_z q(\mathbf{z}^{(t)}, s) \ln \frac{p(\mathbf{z}^{(t)}, s, \mathbf{y}^{(t)})}{q(\mathbf{z}^{(t)}, s)} \\ &= \operatorname*{argmax}_{\pi_n,\boldsymbol{\mu}_n,\boldsymbol{\Sigma}_n} \sum_t \sum_{s_x,s_n,s_h} \int_{\mathbf{z}} q(\mathbf{z}^{(t)}, s) \ln p(\mathbf{z}^{(t)}, s, \mathbf{y}^{(t)}),\end{aligned}$$

since the term $q(\mathbf{z}^{(t)}, s) \ln q(\mathbf{z}^{(t)}, s)$ is not dependent on the parameters of $p(\mathbf{z})$. Filling in the form of $p(\mathbf{z}^{(t)}, s, \mathbf{y}^{(t)})$ we get

$$\begin{aligned}&\{\hat{\pi}_n, \hat{\boldsymbol{\mu}}_n, \hat{\boldsymbol{\Sigma}}_n\} \\ &= \operatorname*{argmax}_{\pi_n,\boldsymbol{\mu}_n,\boldsymbol{\Sigma}_n} \sum_t \sum_{s_x,s_n,s_h} \int_{\mathbf{z}} \rho_s^{(t)} N(\mathbf{z}^{(t)}; \boldsymbol{\eta}_s^{(t)}, \boldsymbol{\Phi}_s^{(t)}) \ln N(\mathbf{z}^{(t)}; g(\mathbf{z}^{(t)}), \boldsymbol{\Psi})\pi_s N(\mathbf{z}^{(t)}; \boldsymbol{\mu}_s, \boldsymbol{\Sigma}_s) \\ &= \operatorname*{argmax}_{\pi_n,\boldsymbol{\mu}_n,\boldsymbol{\Sigma}_n} \sum_t \sum_{s_x,s_n,s_h} \int_{\mathbf{z}} \rho_s^{(t)} N(\mathbf{z}^{(t)}; \boldsymbol{\eta}_s^{(t)}, \boldsymbol{\Phi}_s^{(t)}) \ln \pi_s N(\mathbf{z}^{(t)}; \boldsymbol{\mu}_s, \boldsymbol{\Sigma}_s),\end{aligned}$$

again, since $N(\mathbf{z}^{(t)}; g(\mathbf{z}^{(t)}), \mathbf{\Psi})$ is not dependent on the parameters. Continuing,

$$\begin{aligned}
&\{\hat{\pi}_n, \hat{\boldsymbol{\mu}}_n, \hat{\mathbf{\Sigma}}_n\} \\
&= \underset{\pi_n, \mu_n, \Sigma_n}{\operatorname{argmax}} \sum_t \sum_{s_x, s_n, s_h} \rho_s^{(t)} \ln \pi_s \\
&+ \sum_t \sum_{s_x, s_n, s_h} \int_{\mathbf{z}} \rho_s^{(t)} N(\mathbf{z}^{(t)}; \boldsymbol{\eta}_s^{(t)}, \mathbf{\Phi}_s^{(t)}) \left[-\frac{1}{2} |2\pi \mathbf{\Sigma}_s| - \frac{1}{2} (\mathbf{z}^{(t)} - \boldsymbol{\mu}_s)^T \mathbf{\Sigma}_s^{-1} (\mathbf{z}^{(t)} - \boldsymbol{\mu}_s) \right]
\end{aligned} \tag{9.6}$$

and finally

$$\begin{aligned}
\{\hat{\pi}_n, \hat{\boldsymbol{\mu}}_n, \hat{\mathbf{\Sigma}}_n\} &= \underset{\pi_n, \boldsymbol{\mu}_n, \mathbf{\Sigma}_n}{\operatorname{argmax}} \sum_t \sum_{s_x, s_n, s_h} \rho_s^{(t)} \ln \pi_s - \frac{1}{2} \rho_s^{(t)} \ln |2\pi \mathbf{\Sigma}_s| \\
&- \frac{1}{2} \sum_t \sum_{s_x, s_n, s_h} \rho_s^{(t)} \left[(\boldsymbol{\mu}_s - \boldsymbol{\eta}_s^{(t)})^T \mathbf{\Sigma}_s^{-1} (\boldsymbol{\mu}_s - \boldsymbol{\eta}_s^{(t)}) + Tr[\mathbf{\Sigma}_s^{-1} \mathbf{\Phi}_s^{(t)}] \right].
\end{aligned}$$

We now find the re-estimation formulas for $\hat{\pi}_n$, $\hat{\boldsymbol{\mu}}_n$ and $\hat{\mathbf{\Sigma}}_n$ in turn.

To maximize with respect to π_{s_n}, we use a Lagrange multiplier and arrive at

$$\pi_{s_n} = \frac{\sum_t \sum_{s_x, s_h} \rho_{s_x, s_n, s_h}^{(t)}}{T}, \tag{9.7}$$

where T is the number of frames. In words, the update of π_{s_n} is simply the average over time of the corresponding q distribution weights.

Maximizing w.r.t. $\boldsymbol{\mu}_s$ we arrive at

$$\boldsymbol{\mu}_{s_n} = \frac{\sum_t \sum_{s_x, s_h} \rho_{s_x, s_n, s_h}^{(t)} \boldsymbol{\eta}_{s_x, s_n, s_h}^{n,(t)}}{\sum_t \sum_{s_n, s_h} \rho_{s_x, s_n, s_h}^{(t)}}, \tag{9.8}$$

where we have used the additional superscript n in $\boldsymbol{\eta}_{s_x, s_n, s_h}^{n,(t)}$ to designate the n component of the mean vector $\boldsymbol{\eta} = [\boldsymbol{\eta}^{xT} \boldsymbol{\eta}^{nT} \boldsymbol{\eta}^{h^T}]^T$. This expressions is simply a weighted sum of noise component of the the mixture means $\boldsymbol{\eta}_{s_x, s_n, s_h}^{(t)}$ of the q distribution. In the following derivations, we omit the superscript n.

Maximizing w.r.t. $\mathbf{\Sigma}_s^{-1}$ is a little more tricky. We first need to use the following identity:

$$|X|^{-1} = |X^{-1}| \tag{9.9}$$

and write

$$\begin{aligned}
\hat{\mathbf{\Sigma}}_{s_n}^{-1} &= \underset{\mathbf{\Sigma}_s^{-1}}{\operatorname{argmax}} \sum_t \sum_{s_x, s_h} \rho_s^{(t)} \ln \pi_s - \frac{1}{2} \rho_s^{(t)} \ln |2\pi \mathbf{\Sigma}_s| \\
&- \frac{1}{2} \sum_t \sum_{s_x, s_n, s_h} \rho_s^{(t)} \left[(\boldsymbol{\mu}_s - \boldsymbol{\eta}_s^{(t)})^T \mathbf{\Sigma}_s^{-1} (\boldsymbol{\mu}_s - \boldsymbol{\eta}_s^{(t)}) + Tr[\mathbf{\Sigma}_s^{-1} \mathbf{\Phi}_s^{(t)}] \right] \\
&= \underset{\mathbf{\Sigma}_s^{-1}}{\operatorname{argmax}} \sum_t \sum_{s_x, s_n, s_h} \rho_s^{(t)} \left[\ln |\mathbf{\Sigma}_s^{-1}| - (\boldsymbol{\mu}_s - \boldsymbol{\eta}_s^{(t)})^T \mathbf{\Sigma}_s^{-1} (\boldsymbol{\mu}_s - \boldsymbol{\eta}_s^{(t)}) + Tr[\mathbf{\Sigma}_s^{-1} \mathbf{\Phi}_s^{(t)}] \right].
\end{aligned} \tag{9.10}$$

Next we take the derivative w.r.t. Σ_s^{-1}. We need the following identities:

$$\frac{\delta}{\delta \mathbf{X}} Tr[\mathbf{X}\mathbf{A}] = \mathbf{A} \tag{9.11}$$

$$\frac{\delta}{\delta \mathbf{X}} |\mathbf{X}| = |\mathbf{X}| \mathbf{X}^{-T} \tag{9.12}$$

$$\frac{\delta}{\delta \mathbf{x}} \mathbf{a^T x b} = \mathbf{b^T a} \tag{9.13}$$

Now we can continue

$$\begin{aligned}
0 &= \frac{\delta}{\delta \mathbf{\Sigma}_{s_n}^{-1}} \sum_t \sum_{s_x, s_h} \rho_s^{(t)} \left[\ln |\mathbf{\Sigma}_s^{-1}| - (\boldsymbol{\mu}_s - \boldsymbol{\eta}_s^{(t)})^T \mathbf{\Sigma}_s^{-1} (\boldsymbol{\mu}_s - \boldsymbol{\eta}_s^{(t)}) + Tr[\mathbf{\Sigma}_s^{-1} \mathbf{\Phi}_s^{(t)}] \right] \\
&= \sum_t \sum_{s_x, s_h} \rho_s^{(t)} \left[\mathbf{\Sigma}_s - (\boldsymbol{\mu}_s - \boldsymbol{\eta}_s^{(t)})(\boldsymbol{\mu}_s - \boldsymbol{\eta}_s^{(t)})^T + \mathbf{\Phi}_s^{(t)} \right],
\end{aligned} \tag{9.14}$$

so

$$\sum_t \sum_{s_x, s_h} \rho_s^{(t)} \mathbf{\Sigma}_{s_n} = \sum_t \sum_{s_x, s_h} \rho_s^{(t)} \left[(\boldsymbol{\mu}_s - \boldsymbol{\eta}_s^{(t)})(\boldsymbol{\mu}_s - \boldsymbol{\eta}_s^{(t)})^T + \mathbf{\Phi}_s^{(t)} \right] \tag{9.15}$$

and finally we find the update equation

$$\mathbf{\Sigma}_{s_n}^n = \frac{1}{\sum_t \sum_{s_x, s_h} \rho_s^{(t)}} \sum_t \sum_{s_x, s_h} \rho_s^{(t)} \left[(\boldsymbol{\mu}_s^n - \boldsymbol{\eta}_s^{n,(t)})(\boldsymbol{\mu}_s^n - \boldsymbol{\eta}_s^{n,(t)})^T + \mathbf{\Phi}_s^{nn,(t)} \right] \tag{9.16}$$

9.1.3 Learning $p(\mathbf{h})$

The re-estimation equations for $p(\mathbf{h})$ are analogous. The update for the mixture weights is

$$\pi_{s_h} = \frac{\sum_t \sum_{s_x, s_n} \rho^{(t)}_{s_x, s_n, s_h}}{T}. \tag{9.17}$$

The update equation for the mixture means is

$$\boldsymbol{\mu}^h_{s_h} = \frac{\sum_t \sum_{s_x, s_n} \rho^{(t)}_{s_x, s_n, s_h} \boldsymbol{\eta}^{h,(t)}_{s_x, s_n, s_h}}{\sum_t \sum_{s_n, s_n} \rho^{(t)}_{s_x, s_n, s_h}}, \tag{9.18}$$

and the update equation for the mixture variances is

$$\boldsymbol{\Sigma}^h_{s_h} = \frac{1}{\sum_t \sum_{s_x, s_n} \rho^{(t)}_s} \sum_t \sum_{s_x, s_n} \rho^{(t)}_s \left[(\boldsymbol{\mu}^h_s - \boldsymbol{\eta}^{h,(t)}_s)(\boldsymbol{\mu}^h_s - \boldsymbol{\eta}^{h,(t)}_s)^T + \boldsymbol{\Phi}^{hh,(t)}_s \right]. \tag{9.19}$$

9.2 Convergence Properties

In order to disentangle and learn the parameters of the noise and channel models, the algorithm relies on an accurate speech model $p(\mathbf{x})$ as well as a model for how speech, noise and the channel are combined $p(\mathbf{y}|\mathbf{x}, \mathbf{n}, \mathbf{h})$.

In order to assess the susceptibility of the algorithm to pathological behavior such as slow or stalled convergence, local minima, and saddle points, the algorithm was run on synthetic data. The data was generated by first choosing reasonable parameters for the noise model and estimating the channel parameters for the MIRS frequency response characteristic. Next samples were generated from the speech model, noise model and channel model and combined according to Equation (5.19). The goal of the algorithm was to learn the chosen parameters for the noise and channel models, from the synthetic data.

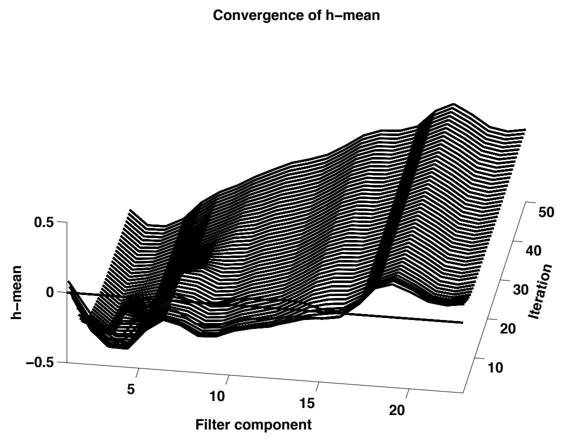


Figure 9.1: Convergence of μ_h as function of iteration, for joint estimation of noise and channel distortion (simulated data). This shows the rapid convergence of the noise model parameters (within 3 iterations).

Figure 9.1 shows the value of μ_h as a function of iteration. In this case, the algorithm learns the channel model quickly, i.e. within about 3-5 iterations.

Figure 9.2 shows a pathological case for learning (i.e. recovering) the noise model parameters. The true noise model has a relatively flat characteristic with a value of around 6. The noise model was initialized with $\boldsymbol{\mu}_n = 0$ and $\boldsymbol{\Sigma}_n = 10$. After 50 iterations the μ_n values for the higher log-spectrum coefficients are still close to 0.

This is due to the algorithm finding and incorrect but plausible "explanation" of the observed signal. The combination of the model for the sound /s/ and the noise model at iteration 50 (see Fig. 9.2) produces a relatively good fit to the observed output. The algorithm eventually recovers and learns the correct model. This shows that the algorithm is susceptible to poor initialization.

Such pathological behavior was not observed when the algorithm was run on real speech data and the noise model was initialized with the mean and variance of the first

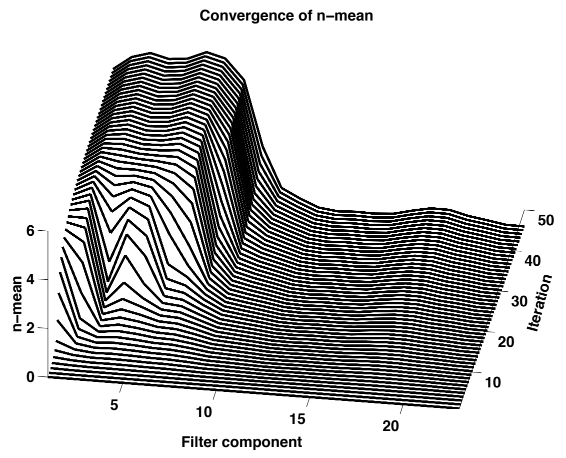


Figure 9.2: Convergence of μ_n as function of iteration, for joint estimation of noise and channel distortion (simulated data). This shows a pathological case where the noise model has been poorly initialized. Such behavior was not observed for real speech data.

20 frames of the speech file.

The convergence rate is greatly dependent on the variance of the initial noise and channel models. Convergence is much slower if the initial variance is set to a small value.

9.3 Results

To assess the performance of the learning algorithm, we report results for sets A and C. In the case of set A the only the parameters of the noise model were. For set C, both the parameters of the noise and channel models were updated.

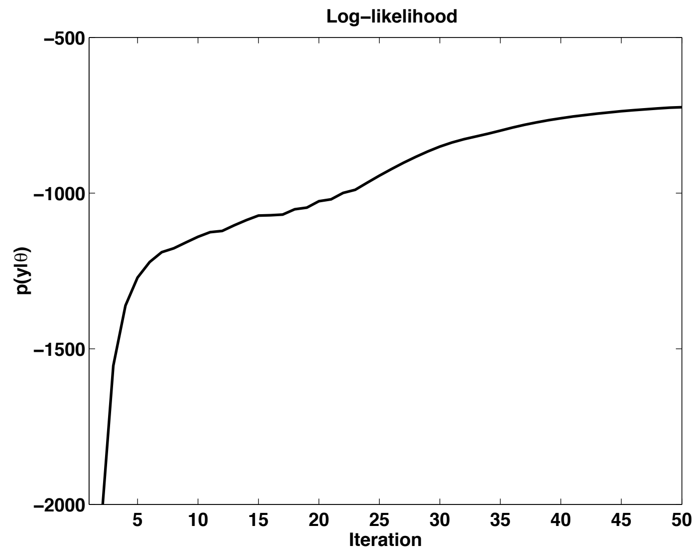


Figure 9.3: Convergence of log-likelihood as a function of iteration for the data shown in Figure 9.2.

9.3.1 Set A: Learning $p(n)$

Accuracy as a function of iteration for set A Figure is shown in Figure 9.4. A single mixture noise model was used. The noise mean and variance were estimated from the first 20 frames, and then the variance was multiplied by 3 in order to avoid local minima and speed up convergence.

The word error rate goes from 17.78%WER (82.22% accuracy) to 15.85%WER (84.15% accuracy) at iteration 4 which is a relative reduction in WER of 10.85%. This is a considerable improvement over using the 20 frame estimate. However, we do not reach the "theoretical minimum" WER of 14.04% that was obtained by estimating the noise model on the true noise.

Notice that the algorithm reaches a maximum in accuracy after 4 iterations and then

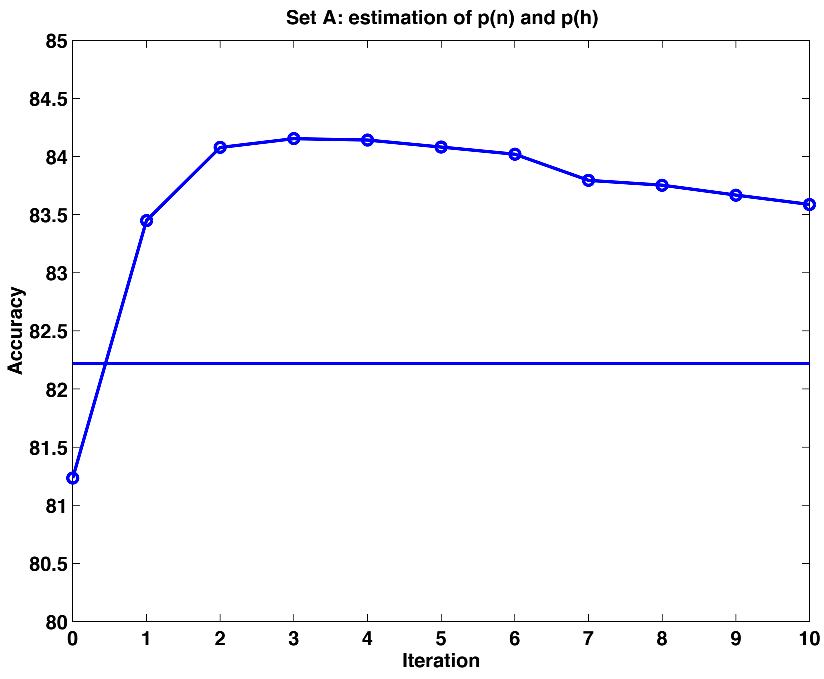


Figure 9.4: Recognition accuracy as a function of iteration for Set A using a single mixture noise model. The horizontal line is the non-adaptive result (82.22%) of using the first 20 frames of the speech file to estimate the noise model.

starts a slow decent. Since the algorithm is attempting to maximize the log probability of the data under the model, this interesting behavior is perhaps attributable to the algorithm using the noise model to better account for deficiencies in the speech model or variation between speakers.

9.3.2 Set C: Joint Learning of $p(n)$ and $p(h)$

We used set C of the Aurora task to evaluate the performance and convergence characteristics of the algorithm to *simultaneously* learn the noise and channel model. Recall that set C has been filtered to simulate a MIRS frequency response characteristic.

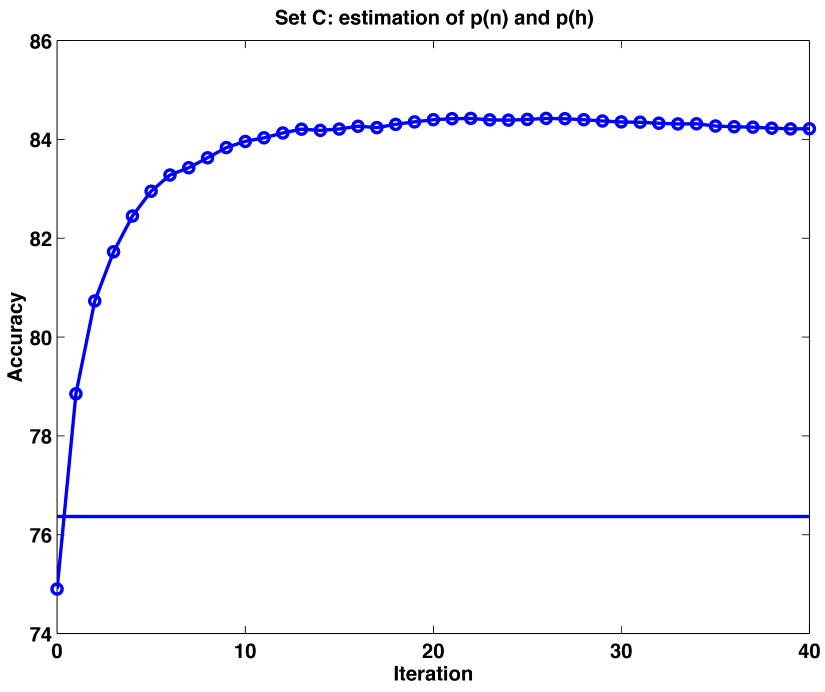


Figure 9.5: Recognition accuracy as a function of iteration for Set C using a single mixture noise model and single mixture channel model. The baseline accuracy is 76.37% for the non-adaptive algorithm, while the adaptive algorithm reaches a maximum of 84.42% at iteration 27.

Figure 9.5 shows the results for jointly learning the noise and channel models. Notice that the convergence rate is slower in this case. This is due to the initialization of the channel model, that will be described in grater detail below. The algorithm reaches a maximum of accuracy 84.42% at iteration 27. The baseline of 23.63% WER (76.37% accuracy) was achieved using the standard method of using 20 frames to estimate a noise model, but no compensation for channel mismatch was used[2]. The algorithm algorithm

[2]This is perhaps an unfair comparison, since using CMN or RASTA would provide a more reasonable

reaches a maximum of 84.42% at iteration 27, which is a relative reduction in WER of 32.92%.

In order to asses the relative importance of the two distortion sources i.e. noise and channel, and the performance of the algorithm, we ran three experiments. In the first, the algorithm was constrained to update only the channel model, in the second the algorithm was constrained to update only the noise model and in the third, both channel and noise models were learned.

Figure 9.6 shows the accuracy results for Subway Noise at 10dB SNR. Results are shown for adaptation of the channel model alone (diamonds), the noise model alone (triangles) and joint estimation of noise and channel distortion (squares). In each case the initial noise model was estimated from the first 20 frames of the a speech file. The initial channel distortion was set to $\mu_h = 0$ with $\sigma_h^2 = 1$. In these experiments, the noise and channel models were single multivariate Gaussians.

First note the results when the algorithm was constrained to adapt only the channel model (i.e. noise model was estimated from first 20 frames and not adapted). In this case, the channel model was initialized to $\mu_h = 0$ and $\sigma_h^2 = 1$. The accuracy goes from 74.42% to a maximum of 86.09%. The non-adaptive algorithm that does not take into account the channel distortion ($\mu_h = 0, \sigma_h^2 = 1 \cdot 10^{-4}$) achieves accuracy of 84.36% for this condition.

A second case was run where only the noise model was adapted. The initial noise model was estimated from the first 20 frames, and the variance was multiplied by 3, in order to speed up convergence. The channel model was set to $\mu_h = 0$ and $\sigma_h^2 = 1 \cdot 10^{-4}$. The recognition accuracy goes from 81.95% to a maximum of 87.23% at iteration 19. The recognition rate declines after iteration 19. This interesting effect may be due to the

baseline

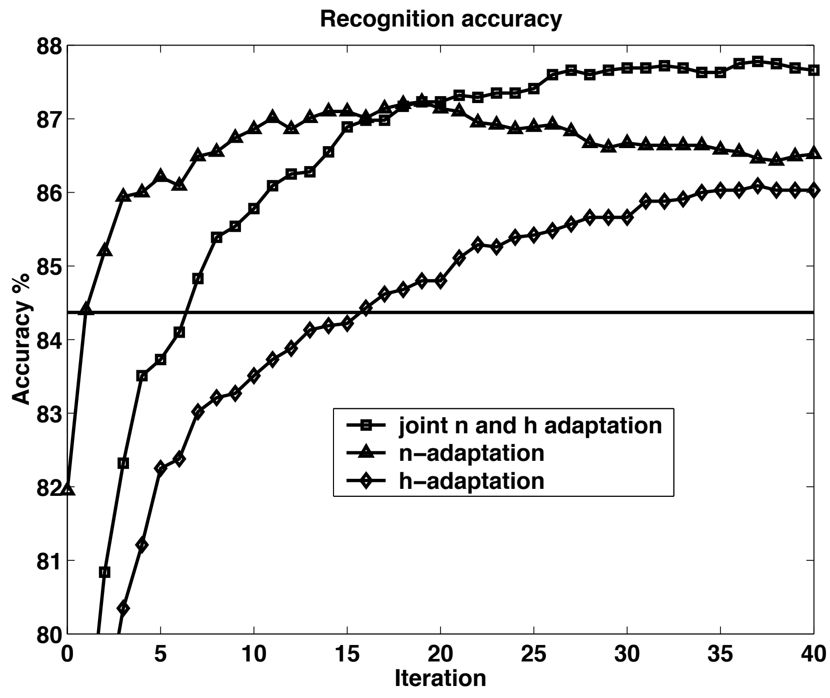


Figure 9.6: Recognition accuracy as a function of iteration. Diamond-line shows accuracy when adapting $\mathbf{h}$ alone, triangle-line shows $\mathbf{n}$-adaptation and square-line shows accuracy for joint $\mathbf{n}$ and $\mathbf{h}$ adaptation. Horizontal line shows result for non-adaptive algorithm.

algorithm attempting to compensate for the channel mismatch with the noise model.

The third case shown in Figure 9.6 is that of joint adaptation of noise and channel. In this case, the accuracy goes from 73.01% to a maximum of 87.78% at iteration 37. This is 0.55% higher than the accuracy for noise adaptation alone at iteration 19 (87.23%). In comparison to the non-adaptive algorithm, the absolute drop in word error rate is 3.4% and the relative drop of is 21.8%. This illustrates well the effectiveness of joint noise and channel adaptation.

These results indicate that the algorithm can successfully and simultaneously learn the additive distortion due to the channel and the non-linear distortion due the noise, and thus successfully untangle these two types of distortion.

9.4 Discussion

When there is no channel distortion, the algorithm reached a maximum quickly, i.e. within 4-5 iterations. When both the noise model and channel models had to be learned, the convergence rate was slower. This is attributable to the naive initialization of the channel model. More clever methods can be used to start the algorithm off closer to the convergence point.

In this chapter we have shown how the noise and channel models can be simultaneously learned from the complete data. By doing this, we were able to improve the results over the baseline considerably and get relatively close to the "theoretical minimum". One of the advantages of this method is that we do not need to classify frames into speech/non-speech frames in order to estimate the environment models.

```
Iterate:
E step
  • Initialization:
      – $\mathbf{z}_0^{(0)} \leftarrow \boldsymbol{\mu}$
  • Iterative update of Taylor series expansion point:
      – calculate $\mathbf{g}(\mathbf{z}_0^{(i)})$, Equation (8.2)
      – calculate $\mathbf{G}(\mathbf{z}_0^{(i)})$, Equation (8.6)
      – calculate $\mathbf{\Phi}_s^{(i)}$, Equation (8.22)
      – calculate $\boldsymbol{\eta}_s^{(i)}$ according to Equation (8.21)
      – $\mathbf{z}_0^{(i+1)} \leftarrow \boldsymbol{\eta}_s^{(i)}$
  • Calculate Mixture weights.
      – calculate $q_y(s)$ according to equation(8.23)
M step
  • calculate $\pi_{s_n}$, $\mu_{s_n}$ and $\Sigma_{s_n}$ according to Equations (9.7),
    (9.8) and (9.16).
  • calculate $\pi_{s_h}$, $\mu_{s_h}$ and $\Sigma_{s_h}$ according to Equations (9.17),
    (9.18) and (9.19).
```

Figure 9.7: The Generalized EM Algorithm for learning environment parameters.

Chapter 10

Taking Uncertainty into Account

Methods that attempt to clean the features, such as Spectral Subtraction, MMSE VTS and MMSE-Algonquin return a point estimate $\hat{X}$ of the clean speech feature for a particular frame. It is an intuitively appealing idea to use a distribution instead of a point estimate[51].

For example, if the corrupting noise is a telephone ring-tone, the ring will corrupt a subset of the features, i.e. those with frequencies corresponding to the frequency of the ring-tone. We would like the recognizer to discount or even overlook those corrupted feature components.

An algorithm such as MMSE-Algonquin attempts to repair those features that have been masked by the ring, by employing the prior information contained in the speech model and the noise model. The speech model of the cleaning algorithm is a less accurate version of the speech (language and acoustic) models of the recognizer. Since the true features are masked, the repaired features may be in error.

10.1 Distributions as Observations

Instead of repairing the corrupted features we would like to pass to the recognizer information about the severity of the distortion of each component of the feature vector e.g. using the variance of the estimate of $\mathbf{x}$. Instead of passing $\hat{\mathbf{x}} = \int_{\mathbf{x}} \mathbf{x} p(\mathbf{x}|\mathbf{y}_{obs})$, we could pass $f(\mathbf{x}) = p(\mathbf{x}|\mathbf{y}_{obs})$. The front end of the recognizer would then calculate:

$$score = \int_{\mathbf{x}} p(\mathbf{x}|\mathbf{y}_{obs}) p(\mathbf{x}|s) = E[p(\mathbf{x}|\mathbf{y})]. \tag{10.1}$$

However, this is difficult do justify.

When discussing inference in the generative model for noisy speech, in Chapter 4, we noted that the message that gets passed from the x to s in the graph of Figure 4.3, is the function $f(\mathbf{x}) = p(\mathbf{y}_{obs}|\mathbf{x})$. One approach to taking uncertainty into account would therefore be to pass to the recognizer $f(\mathbf{x}) = p(\mathbf{y}_{obs}|\mathbf{x})$, instead of $\hat{\mathbf{x}}$, where $\mathbf{y}_{obs}$ is the noisy observation. The end result is that instead of using $p(\mathbf{x}|s)$ as the observation likelihood, we use $p(\mathbf{y}|s)$, since

$$p(\mathbf{y}|s) = \int_{\mathbf{x}} p(\mathbf{y}_{obs}|\mathbf{x}) p(\mathbf{x}|s). \tag{10.2}$$

The form of $f(\mathbf{x}) = p(\mathbf{y}_{obs}|\mathbf{x})$ is shown in Figures 10.1 and 10.2 for two different values of the variance of the noise.

We can use the message passing paradigm to find the exact form of $p(y_{obs}|x)$ in the log-spectrum domain. Assuming the relationship between y, n and x is exact, i.e. $n = \log(\exp(y) - \exp(x))$ so $p(y|x,n) = \delta(n - \log(\exp(y) - \exp(x)))$ then

$$p(y_{obs}|x) = \frac{\exp(y_{obs})}{\exp(y_{obs}) - \exp(x)} N(\log(\exp(y_{obs}) - \exp(x)); \mu_n, \sigma_n). \tag{10.3}$$

In the log-spectrum domain, the message $f(x) = p(y_{obs}|x)$ that needs to be sent to the recognizer is shown in Figures 10.1 for noise with large variance ($\sigma_n = 1$) and in 10.2

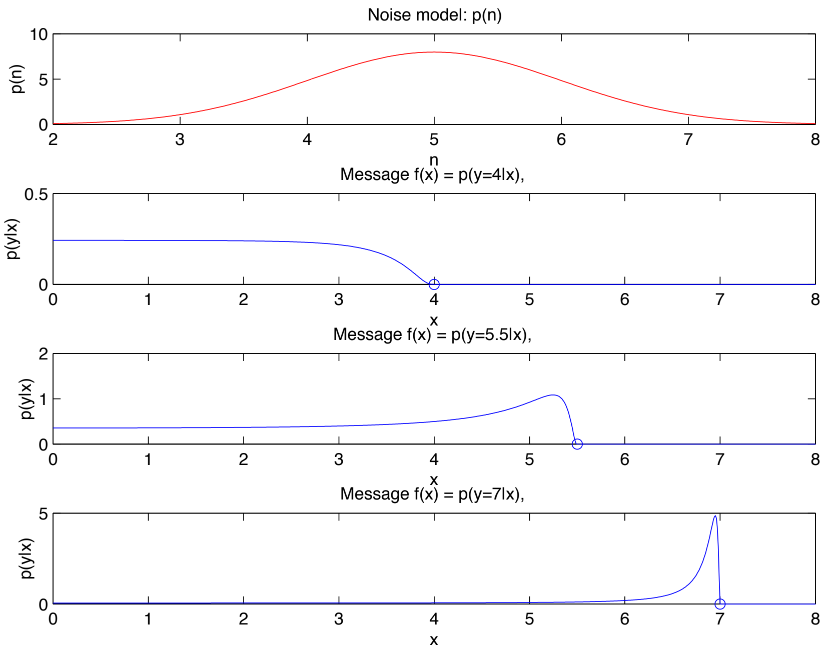


Figure 10.1: Messages $f(x) = p(y|x)$ for y equal to 4, 5.5 and 7. The noise distribution has mean 5 and standard deviation 1

for noise with smaller variance ($\sigma_n = 0.2$). These messages are clearly not Gaussian. The most “non-Gaussian” aspect is that the tail that goes to $-\infty$. The figures suggest that the shape of the message falls into two or three regimes, depending on the relative values of y_{obs} and μ_n.

If y is larger than n by some margin, the message becomes approximately Gaussian. The integral over this Gaussian goes to 1. Thus, as the observation gets larger than the noise, the message approximates a delta function at y. In this region, using a point estimate for x, e.g. by spectral subtraction would produce a similar result.

When y is smaller than μ_n the message approximates a Heavyside function, with a smooth roll-off. Thus, any two speech distribution who’s mass is substantially below the

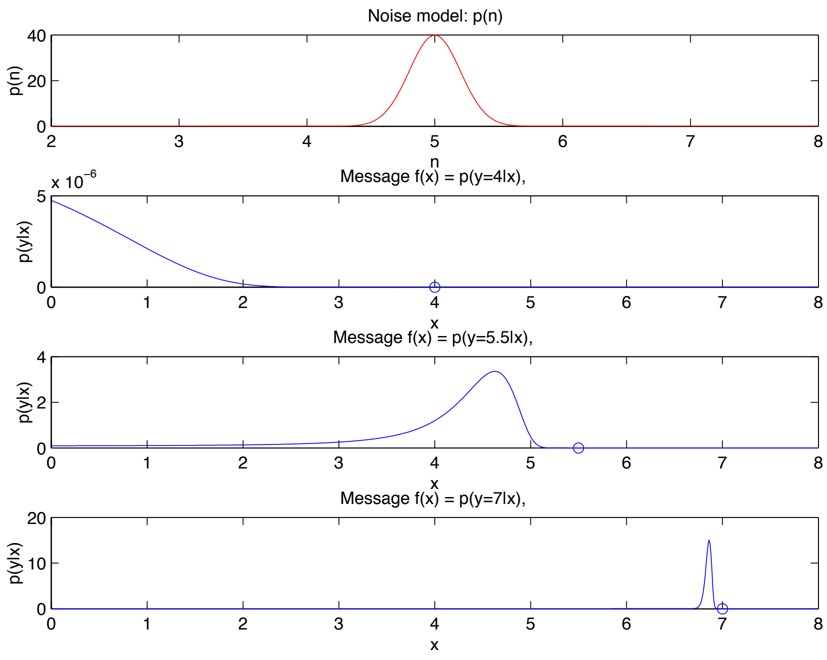


Figure 10.2: Messages $f(x) = p(y|x)$ for y equal to 4, 5.5 and 7. The noise distribution has mean 5 and standard deviation 0.2

noise mean, will produce a similar score when convolved with this message. In this case, we do not give preference to different distributions that are below the noise level.This is precisely the essence of this method, and the potential strength over that of SS and other methods that produce point estimates.

10.2 The Effect of Uncertainty

When there is noise and channel distortion in the environment, we observe corrupted features $\mathbf{Y}$ instead of $\mathbf{X}$. The environmental noise process introduces both bias and fundamental uncertainty. Bias shifts the classification boundaries, but can be accounted for.

However uncertainty increases the overlap of class conditional likelihood distributions, and thus the classification error increases. As was discussed in Chapter 3, the optimal classification strategy is based on using the posterior of the noisy speech $p(\mathbf{s}|\mathbf{Y})$. By the data processing inequality[10] it is impossible to gain more information about s by manipulating $\mathbf{y}$ e.g. by cleaning $\mathbf{y}$ to produce $\hat{\mathbf{x}}$.

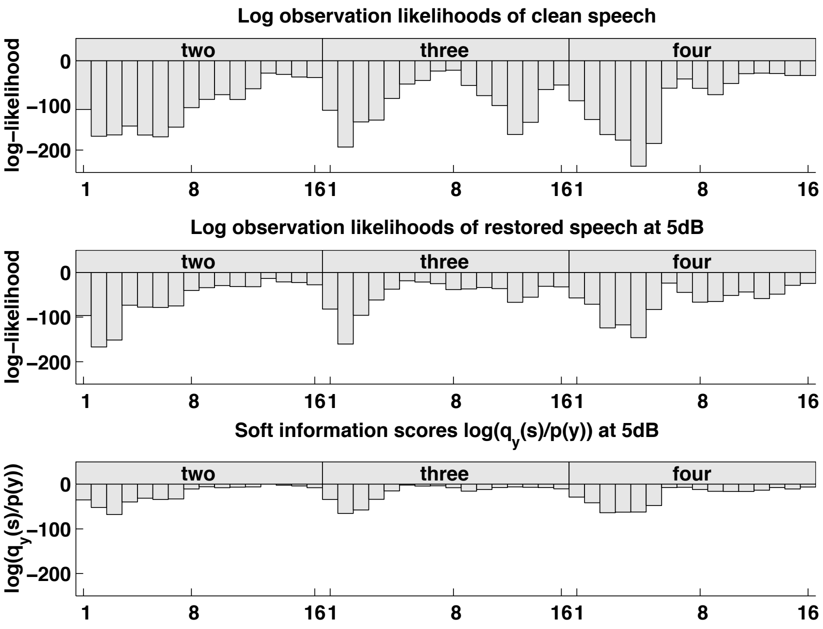


Figure 10.3: The plots show the log observation likelihood for the states s in the word models for 'two','three' and 'four' for frame 30 of file MAH_3A. This file contains the utterance 'three'. The plots show clean speech $\log(p(\mathbf{x}|s))$, cleaned speech $\log(p(\hat{\mathbf{x}}|s))$ and the soft information score which is approximately equal to $\log(p(\mathbf{y}|s)) + const$.

Intuitively, the effect of noise is to reduce our certainty that an observation belongs to one class rather than the other. Figure 10.3 shows observation scores for a particular speech frame. At the top, the log observation likelihoods $\log(p(\mathbf{x}|s))$ of the clean speech frame are shown. The plot shows the scores for each of the 16 states of the models 'two', 'three' and 'four'. The observation frame is taken from the middle of the word 'three'. The bottom two plots show observation scores for the same frame with noise at 5dB SNR. The middle plot shows the log observation likelihood $\log(p(\hat{\mathbf{x}}|s))$ of the cleaned speech frame and the bottom plot shows the soft information score which is approximately equivalent to $\log(p(\mathbf{y}|s))$[1]. Notice that $p(\hat{\mathbf{x}}|s)$ looks more like $p(\mathbf{x}|s)$, but the range of scores for the soft information case is smaller. This is in accordance with our intuition that the "specificity" of the feature is smaller when there is noise in the observation. Instead of remaining neutral as to which class the observation belongs, a cleaning method is forced to choose a single $\hat{\mathbf{x}}$, and it may amplify the error in the case of a wrong choice.

10.3 Uncertainty Decoding

Figure 10.1 shows the form of $f(\mathbf{x}) = p(\mathbf{y}_{obs}|\mathbf{x})$. As noted before, if we have two acoustic classes with distributions that lie below the noise level, both distributions will be convolved with the tail of the likelihood, and thus give similar scores. This is exactly what we would like, since the recognizer should not use this information to differentiate between the two speech classes[2]. A problem with this approach is that the form of $f(\mathbf{x}) = p(\mathbf{y}_{obs}|\mathbf{x})$ is very non-Gaussian, and it is therefore computationally expensive to evaluate the integral in Equation (10.2) numerically. Computationally attractive approximations are possible, e.g. using splines or a mixture of Gaussian.

[1]The plot shows $log(q_{\mathbf{y}}(s)/p(s))$ which is approximately equal to $log(p(\mathbf{y}|s)/p(\mathbf{y})) = log(p(\mathbf{y}|s)) + const.$

[2]The *Noise Masking* technique [49] had a similar motivation, but was formulated in a non-probabilistic way.

Alternatively, we can estimate $p(\mathbf{x}_i|\mathbf{y}_{i,obs})/p(\mathbf{x}_i)$, since

$$\begin{aligned}
p(\mathbf{s}|\mathbf{Y}) &= p(\mathbf{Y})^{-1}p(s_0)\prod_i p(\mathbf{y}_{i,obs}|s_i)\cdot p(s_i|s_{i-1}) && (10.4)\\
&\approx p(s_0)\prod_i \frac{p(\mathbf{y}_{i,obs}|s_i)\cdot p(s_i|s_{i-1})}{p(\mathbf{y}_{i,obs})} && (10.5)\\
&= p(s_0)\prod_i \frac{\int p(\mathbf{y}_{i,obs}|\mathbf{x}_i)p(\mathbf{x}_i|s_i)d\mathbf{x}}{p(\mathbf{y}_{i,obs})}\cdot p(s_i|s_{i-1}) && (10.6)\\
&= p(s_0)\prod_i \int \frac{p(\mathbf{x}_i|\mathbf{y}_{i,obs})}{p(\mathbf{x}_i)}p(\mathbf{x}_i|s_i)d\mathbf{x}\cdot p(s_i|s_{i-1}). && (10.7)
\end{aligned}$$

The approximation above is a result of assuming that $p(Y) \approx \prod_i p(\mathbf{y}_{i,obs})$. In this case, the goal is to estimate $p(\mathbf{x}_i|\mathbf{y}_{i,obs})/p(\mathbf{x}_i)$ in a form that allows for the integral to be calculated easily, e.g. in a Gaussian form. Some noise cleaning methods e.g. MMSE-Algonquin, employ Gaussian speech priors $p(\mathbf{x}_i)$ and estimate a Gaussian posterior $p(\mathbf{x}_i|\mathbf{y}_{i,obs})$, and can thus be used in this context since $p(\mathbf{x}_i|\mathbf{y}_{i,obs})/p(\mathbf{x}_i)$ is also Gaussian. Note however that the true form of $p(\mathbf{x}_i|\mathbf{y}_{i,obs})/p(\mathbf{x}_i)$ is proportional to $f(\mathbf{x}_i) = p(\mathbf{y}_{i,obs}|\mathbf{x}_i)$, shown in Figure 10.1.

In Chapter 3, a second alternative to the model adaptation method was proposed that also preserve the information about the uncertainty of the observations. This method relies on estimating the *soft information score* $p(s_i|\mathbf{y}_{i,obs})/p(s_i)$[3] and returning this value to the recognizer. The recognizer then calculates

$$p(\mathbf{s}|\mathbf{Y}) \approx p(s_0)\prod_i \frac{p(\mathbf{y}_{i,obs}|s_i)}{p(\mathbf{y}_{i,obs})}\cdot p(s_i|s_{i-1}) = p(s_0)\prod_i \frac{p(s_i|\mathbf{y}_{i,obs})}{p(s_i)}\cdot p(s_i|s_{i-1}). \quad (10.8)$$

Note that we divide by $p(s_i)$ in Equation (10.8). Speech recognition systems employ complex speech models including language models and word or phone HMMs that encode the transition probabilities $p(s_i|s_{i-1}\ldots s_0)$ between states. In the case of Algonquin, speech is modelled by a Gaussian Mixture Model. By deferring the hard decision

[3] In fact, Equation (8.17) gave $p_l(\mathbf{y}_{obs}|s)$ directly which is an approximation to $p(\mathbf{y}_{obs}|s)$.

to the decoding step of the recognizer, we avoid making a decision based on the much weaker state transition model of the cleaning algorithm. The MMSE version of Algonquin uses a GMM to model speech and therefore does not use state transition probabilities. We remove the effect of the "language model" of the cleaning algorithm by dividing by $p(s)$ in Equation (10.8).

Thus, if we can approximate $p(s_i|\mathbf{y}_{i,obs})/p(s_i)$ we can preserve information about uncertainty in the decoder. The Algonquin framework allows us to do this.

The two paradigms for robust speech recognition that have been mentioned before i.e. the *feature domain* paradigm and the *model domain* paradigm, were characterized by their relation to the block diagram of a speech recognizer (see Figure 2.1). If we wish to use soft information scores, we need to alter or replace the **Acoustic scores** block.

10.4 Estimate of $p(s|\mathbf{y})/p(s)$

The Algonquin framework is well suited to demonstrate the importance of retaining uncertainty information in the decoding of speech. While some noise robustness methodologies, such as spectral subtraction, use point estimates for the noise process, Algonquin used Gaussian mixture models to model both speech and noise. The uncertainty introduced by the noise process is captured in the variance parameters of the noise model.

Algonquin uses a variational method to produce an approximation $q_{\mathbf{y}}(\mathbf{x})$ to the posterior $p(\mathbf{x}|\mathbf{y}_{obs})$. The approximate posterior is used to calculate a point estimate of the clean speech features $\hat{\mathbf{x}}$ through an MMSE estimate:

$$\hat{\mathbf{x}} = \int \mathbf{x} p(\mathbf{x}|\mathbf{y}_{obs}) d\mathbf{x} \approx \int \mathbf{x} \sum_{\mathbf{s}} q_{\mathbf{y}}(s) q_{\mathbf{y}}(\mathbf{x}|s) d\mathbf{x} \tag{10.9}$$

Using the cleaned features $\hat{\mathbf{x}}$, the observation likelihood calculated by the recognizer is thus $p(\hat{\mathbf{x}}|s)$. In order to use soft information, we require the evaluation of

$$\frac{p(s|\mathbf{y}_{obs})}{p(s)} \approx \frac{q_{\mathbf{y}}(s)}{p(s)}, \tag{10.10}$$

which is substituted for $p(\hat{\mathbf{x}}|s)$ in the recognizer. As we will see below $p(s|\mathbf{y}_{obs}) \approx q_{\mathbf{y}}(s)$ so all the components required to calculate the soft information score in Equation (10.10) are available from the calculation of the point estimate in Equation (10.9).

The Algonquin framework employs Gaussian mixture models to model the speech and noise in the log-spectrum domain. Recall the joint distribution over noisy speech $\mathbf{y}$, speech $\mathbf{x}$, speech class s_x, noise $\mathbf{n}$, noise class s_n is

$$\begin{aligned} p(\mathbf{y}, \mathbf{x}, \mathbf{n}, s^x, s^n) &= p(\mathbf{y}|\mathbf{x}, \mathbf{n}) p(s^x) p(\mathbf{x}|s^x) p(s^n) p(\mathbf{n}|s^n) p(s^h) p(\mathbf{h}|s^h) \\ &= \mathcal{N}(\mathbf{y}; \mathbf{g}\big(\big[\mathbf{x}^T \mathbf{n}^T \mathbf{h}^T\big]^T\big), \boldsymbol{\Psi}) \cdot \pi^x_{s^x} \mathcal{N}(\mathbf{x}; \boldsymbol{\mu}^x_{s^x}, \boldsymbol{\Sigma}^x_{s^x}) \cdot \pi^n_{s^n} \mathcal{N}(\mathbf{n}; \boldsymbol{\mu}^n_{s^n}, \boldsymbol{\Sigma}^n_{s^n}) \cdot \pi^h_{s^h} \mathcal{N}(\mathbf{h}; \boldsymbol{\mu}^h_{s^h}, \boldsymbol{\Sigma}^h_{s^h}). \end{aligned} \tag{10.11}$$

For the current frame of noisy speech $\mathbf{y}$, Algonquin approximates the posterior using a simpler, parameterized distribution, q:

$$p(\mathbf{x}, \mathbf{n}, \mathbf{h}, s^x, s^n, s^h|\mathbf{y}_{obs}) \approx q_{\mathbf{y}}(\mathbf{x}, \mathbf{n}, \mathbf{h}, s^x, s^n, s^h). \tag{10.12}$$

The q function is a mixture of Gaussians:

$$q_{\mathbf{y}}(\mathbf{x}, \mathbf{n}, \mathbf{h}, s^x, s^n, s^h) = \sum_{\{s^x, s^n, s^h\}} q_{\mathbf{y}}(s^x, s^n, s^h) q_{\mathbf{y}}(\mathbf{x}, \mathbf{n}, \mathbf{h}|s^x, s^n, s^h), \tag{10.13}$$

where the $q_{\mathbf{y}}(s^x, s^n, s^h)$s serve as mixture weights. Note that

$$p(s^x|\mathbf{y}) \approx q_{\mathbf{y}}(s^x) = \sum_{s^n, s^h} q_{\mathbf{y}}(s^x, s^n, s^h) \tag{10.14}$$

which is used in the calculation of the soft information score in Equation (10.10).

The variational parameters of q are adjusted to make this approximation accurate, and then q is used as a surrogate for the true posterior when computing $\hat{\mathbf{x}}$ and calculating the soft information score $q_{\mathbf{y}}(s)/p(s)$.

In Chapter 8 we derived the expression for $q_{y_{obs}}(s)$:

$$q_{y_{obs}}(s) = \rho_s = \frac{\exp(L_s)}{\sum_j \exp(L_j)} \tag{10.15}$$

where

$$\begin{aligned} L_s &= \ln \pi_s - \frac{1}{2}\ln|2\pi\mathbf{\Sigma}_s| + \frac{1}{2}\ln|2\pi\mathbf{\Phi}_s| \\ &- \frac{1}{2}\left\{(\boldsymbol{\mu}_s - \boldsymbol{\eta}_s)^T\mathbf{\Sigma}_s^{-1}(\boldsymbol{\mu}_s - \boldsymbol{\eta}_s)\right\} \\ &- \frac{1}{2}\left[(\mathbf{y}-\mathbf{g})^T\mathbf{\Psi}^{-1}(\mathbf{y}-\mathbf{g})\right]. \end{aligned} \tag{10.16}$$

This term can be viewed as the integral of over $\mathbf{x}$, $\mathbf{n}$ and $\mathbf{h}$ of the joint posterior distribution $p(\mathbf{x}, \mathbf{n}, \mathbf{h}, s|\mathbf{y}_{obs})$ and is an approximation to $p(s|\mathbf{y}_{obs})$. This expression resembles the expression for the scaling term in the multiplication of two Gaussians (see A-6), which we would get using a Laplace type method. We are in fact evaluating

$$q_{\mathbf{y}_{obs}}(s) = \int \left[p_l(\mathbf{y}_{obs}|\mathbf{x}, \mathbf{n}, \mathbf{h})\right] \cdot \left[p(\mathbf{x}, \mathbf{n}, \mathbf{h}, s)\right] / p(\mathbf{y}_{obs}) d\mathbf{x} d\mathbf{n} d\mathbf{h} \tag{10.17}$$

the integral over the multiplication of the interaction likelihood $p_l(\mathbf{y}_{obs}|\mathbf{x}, \mathbf{n}, \mathbf{h})$, which is a "one dimensional" Gaussian and the prior $p(\mathbf{x}, \mathbf{n}, \mathbf{h}, s)$, which is a multidimensional Gaussian.

In the noise free case ($\mathbf{y}_{obs} = \mathbf{x}$), we would like the soft information score to be equal to the acoustic score for clean observations. The **Acoustic score** block in Figure 2.1 calculates the log-likelihood acoustic scores:

$$f(s) = \log(p(\mathbf{x}|s)) = -\frac{1}{2}\log|2\pi\mathbf{\Sigma}_s^x| - \frac{1}{2}(\boldsymbol{\mu}_s^x - \mathbf{x})^T\mathbf{\Sigma}_s^{x-1}(\boldsymbol{\mu}_s^x - \mathbf{x}) \tag{10.18}$$

For the noise free case, posterior modes are equal to the observation and thus $\boldsymbol{\eta}_s \approx \mathbf{x}$. Also, $\mathbf{y} \approx \mathbf{g}$ and $\mathbf{\Phi}_s \approx 0$ so

$$L_{s,\text{noise free}} \approx -\frac{1}{2}\ln|2\pi\mathbf{\Sigma}_s| - \frac{1}{2}(\boldsymbol{\mu}_\mathbf{s} - \boldsymbol{\eta}_\mathbf{s})^\mathbf{T}\mathbf{\Sigma}^{-1}(\boldsymbol{\mu}_\mathbf{s} - \boldsymbol{\eta}_\mathbf{s}) + \ln \pi_s \tag{10.19}$$

which shows that in the noise free case, the normalized score is the same as we would get by passing the clean feature to the recognizer.

10.5 Experiments

The Aurora data set is supplied with a standard Mel-frequency Cepstrum Coefficient (MFCC) front end and the CU-HTK speech recognizer. For the experiments reported here, we used filter-bank parameters without delta or acceleration features. These features were produced by altering the standard front end such that it writes out the log-Mel-spectrum values just prior to taking the DCT. It is known that MFCC parameters perform considerably better than filter-bank parameters (see Figure D.1).

The standard HTK recognizer accepts observation features $\mathbf{x}$ and calculates acoustic scores internally based on the acoustic models $p(\mathbf{x}|s)$. Experiments based on feature cleaning and model adaptation can be performed without altering the recognizer, i.e. by supplying $\hat{\mathbf{x}}$ or altering $p(\mathbf{x}|s)$ respectively. However, the uncertainty decoding paradigm relies on the fusion of the noise adaptation stage and calculation of acoustic scores. Thus, the HTK recognizer had to be altered to accept the scores $q_{\mathbf{y}_i}(s_i)/p(s_i)$ calculated by the algorithm. These scores were substituted for $p(\mathbf{x}_i|s_i)$ in the recognizer.

Twelve speech models are used in the Aurora task, 'zero' though 'nine', 'oh' and silence. Each model has 16 states and the silence model has 3 states, for a total of 179 states. Thus, $q_{\mathbf{y}_i}(s_i)/p(s_i)$ had to be calculated for each state s_i for each frame i, as described above.

The MMSE-Algonquin algorithm requires a GMM speech model $p(\mathbf{x})$. $p(\mathbf{x})$ was constructed from the HMM models trained by HTK. The mixture means $\boldsymbol{\mu}^x_{s^x}$ and variance $\boldsymbol{\Sigma}^x_{s^x}$ (see Equation (10.11)) were copied directly from the acoustic models of the

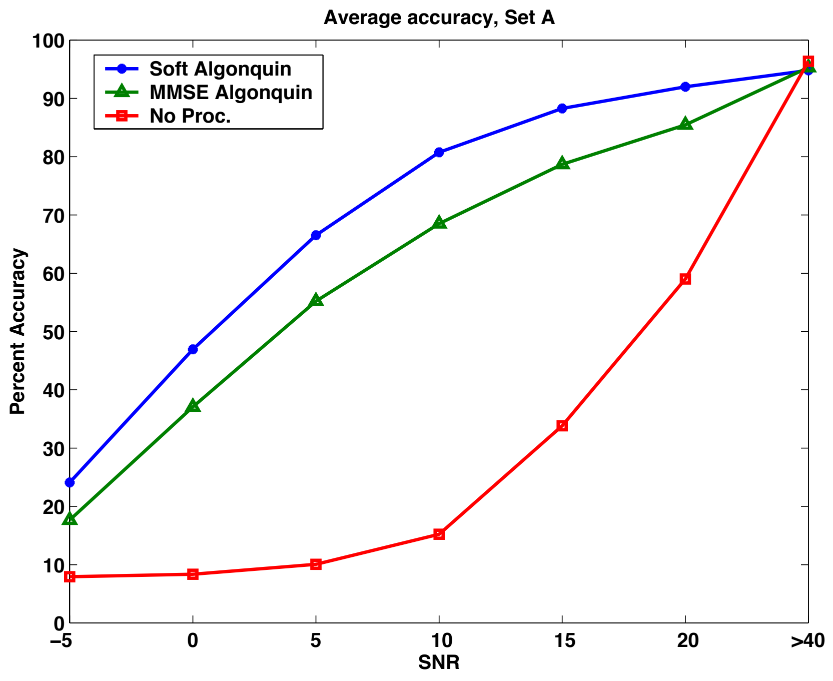


Figure 10.4: Average recognition accuracy as a function of signal to noise ratio. The three conditions shown are *soft information* i.e. using $q_{\mathbf{y}_i}(s_i)/p(s_i)$ in the recognizer, using MMSE Algonquin for feature cleaning and no processing of the corrupted features.

recognizer. To find the mixture weights $\pi^x_{s^x}$, a 179×179 state transition matrix was first constructed from the language model and the transition matrices of the HTK HMM word models. Then the stationary distribution of the transition matrix was found and multiplied by the mixture weights of the Gaussian components of the acoustic models. This resulted in a 552 component Gaussian mixture model.

The noise model consisted of a single component multivariate Gaussian. A different model was estimated for each utterance, from the first 20 frames of that file.

10.6 Results

The calculation of the soft information score in Equation (10.10) and the point estimate Equation (10.9) share almost all of the same steps. We can therefore provide a comparison that differs only in this aspect (i.e. point estimate vs. soft information), while holding all other aspects constant, such as methodology, approximation errors, speech and noise models etc.

Figure 10.4 shows a comparison of the techniques of passing a point estimate of clean speech to that of taking uncertainty into account by using $q_{\mathbf{y}_{i,obs}}(s_i)/p(s_i)$. As can be seen recognition accuracy improves considerably for all SNRs except for clean speech where it is slightly reduced. For example, at 15dB, the average accuracy goes from 78.71% accuracy to 88.29% which is an increase in accuracy of 9.58% and a relative drop in Word Error Rate (WER) of 45.01%. As expected, the use of soft information is most effective at intermediate SNRs. For clean speech, there is a drop in accuracy of 0.49% or 10.62% relative WER. At "infinite" SNR, we should ideally leave the parameters unchanged. Approximation error and error in estimation of the noise parameters seems to have a greater adverse effect on the soft information method.

The WER for MMSE-Algonquin is 35.88% while for soft-Algonquin the WER is reduced to 25.1%. The relative reduction in WER is shown in Table 10.1 (see Tables D.13 and D.14 for absolute WER). The relative reduction in average WER for noise conditions 20dB through 0dB is 28.31%.

Recall that the WER was 17.78% for the MMSE-Algonquin algorithm when using a recognizer with cepstrum domain acoustic models. This shows the effectiveness of the DCT and time derivatives and the importance of using the soft information paradigm in conjunction with these linear transforms.

	Subway	Car	Babble	Exhibition	Average
clean	-17.78%	-8.52%	-5.95%	-11.26%	-10.62%
20dB	48.53%	34.42%	44.37%	47.20%	44.91%
15dB	45.41%	37.37%	48.81%	47.00%	45.01%
10dB	35.69%	38.21%	43.28%	40.53%	38.88%
5dB	18.76%	27.19%	31.85%	26.69%	25.27%
0dB	7.77%	13.37%	26.63%	18.80%	15.70%
-5dB	3.10%	2.40%	15.92%	11.41%	7.84%
Average	24.65%	24.71%	35.04%	31.18%	28.31%

Table 10.1: Relative reduction in word error rate in percent on Set A of the Aurora 2 data set. Soft information method compared to using a point estimate of clean speech. See Tables D.13 and D.14 for absolute WER

10.7 Discussion

In this chapter we introduced a new paradigm for robust speech recognition. This paradigm allows us to take into account the uncertainty of observations introduced by noise.

In Chapter 3 we saw that the optimal classification strategy involves using $p(\mathbf{Y}|\mathbf{s})$ rather than $p(\hat{\mathbf{X}}|\mathbf{s})$, and gave results for a simple classifier. Here we have shown conclusively that this holds for the more complex problem of robust speech recognition.

We used the Algonquin framework to show the promise of this method. However, any method capable of estimating $p(\mathbf{x}|\mathbf{y})/p(\mathbf{x})$, $p(s|\mathbf{y})/p(s)$ or $p(\mathbf{y}|s)$ can be used.

An important topic of future research is how to use the soft information paradigm in conjunction with linear transforms across time and frequency.

Chapter 11

Conclusion

11.1 Summary and Contributions

In this thesis I advocated a probabilistic view of robust speech recognition. I discussed the problem of classification of distorted features using an optimal classifier, and how the generation of noisy speech can be represented as a generative graphical probability model. By doing so, my aim was to build a conceptual framework that provides a unified understanding of robust speech recognition, and to some extent bridges the gap between a purely signal processing viewpoint and the pattern classification or decoding viewpoint.

The most tangible contribution of this thesis is the introduction of the Algonquin method for robust speech recognition. It exemplifies the probabilistic method and encompasses a number of novel ideas. For example, it uses a probability distribution to describe the relationship between clean speech, noise, channel and the resultant noisy speech. It employs a Variational approach to find an approximation to the joint posterior distribution which can be used for the purpose of restoring the distorted observations. It also allows us to estimate the parameters of the environment using a Generalized EM method.

Another important contribution of this thesis is a new paradigm for robust speech recognition, which we call *uncertainty decoding*. This new paradigm follows naturally from the standard way of performing inference in the graphical probability model that describes noisy speech generation.

To summarize:

- A general probabilistic view of speech recognition in adverse environments was formulated that incorporates noise and channel distortion. This was done using the formalism of graphical probability models.
- Standard methods for robust speech recognition were related to performing inference in this graphical probability model.
- A new view of the relationship between noisy speech, clean speech, noise and channel in the log-spectrum and MFCC domains was presented. We showed how uncertainty is introduced as a consequence of dimensionality reduction.
- As a result of this new view, the *interaction likelihood* was formulated, which describes the relationship between noisy speech, clean speech, noise and channel in a probabilistic way.
- In order to perform inference the graphical model for noisy speech a new robustness method, Algonquin, based on variational inference was introduced. A number of novel ideas are involved:
    - Approximation of the interaction likelihood via the use of the Vector Taylor Series.

  - Iteratively improving the approximation by updating the expansion point of the Vector Taylor Series based on the mode of the approximate posterior.
  - Approximating the joint conditional distribution with a Gaussian distribution.

  The Algonquin framework was shown to perform considerably better than previous methods.

- A novel method for learning the environmental parameters was introduced. The negative relative entropy loss function of the variational method, lead to Generalized EM method to update the parameters of the prior models.

- Finally, we introduce the *soft information* paradigm and showed how the Algonquin method can be used within this paradigm.

11.2 Future Extensions

There are a number of issues yet to be explored in the Algonquin framework. The use of time dynamics in the noise model may be important for dealing with noise with strong temporal structure. Delta and acceleration features were not taken advantage of, although they are known to be important. This is an important area of study. The method described for learning the parameters of the environment in Chapter 9 is applicable to batch processing of speech files. Although there are many applications that require batch processing, more applications require online adaptation. This has yet to be explored fully.

One of the key insights that lead to the Algonquin method was to approximate the posterior $p(\mathbf{x}, \mathbf{n}, \mathbf{h}|\mathbf{y}_{obs})$. It is possible to approximate the posterior using a number of other methods. We mentioned the use of Gaussian Basis Functions and efficient sampling, but there are probably many more.

In the last Chapter we introduced the uncertainty decoding paradigm. One can envision a whole new field of study within robust speech recognition that takes advantage of this new viewpoint.

Appendix A

Notational Conventions

$x, X, \mathbf{x}, \mathbf{X}$	–	Variable associated with clean speech. The variable can appear in various domains. In this thesis the conversion is to use x to designate the time domain signal, $\mathbf{x}$ to designate a log-spectrum feature vector and $\mathbf{x}_c$ to designate an MFCC domain feature vector.
$y, Y, \mathbf{y}, \mathbf{Y}$	–	Variable associated with corrupted speech.
$n, N, \mathbf{n}, \mathbf{N}$	–	Variable associated with noise.
$h, H, \mathbf{h}$	–	Variable associated with the channel.
$\mathbf{z}$	–	A concatenated vector $\mathbf{z} = [\mathbf{x}^T \mathbf{n}^T \mathbf{h}^T]^T$.
$x[m], y[m], n[m]$	–	Time domain sample of the corrupted signal, clean signal and noise signal respectively.
$h[m]$	–	Impulse response of the channel model.
$\mathbf{y}, \mathbf{x}, \mathbf{n}, \mathbf{h}$	–	Log-spectrum domain feature vectors for the corrupted signal, clean signal and noise signal and channel respectively.
$g(z)$	–	Shorthand for $x + h + \ln(1 + \exp(n - x - h))$

G_x	–	Matrix derivative of g w.r.t. $\mathbf{x}$, used in the Taylors Series Linearization of the interaction equation.
$p(\mathbf{x})$	–	Speech model distribution, i.e. the speech prior.
$p(\mathbf{n})$	–	Noise model distribution, i.e. the noise prior.
$p(\mathbf{h})$	–	Channel model distribution, i.e. the channel prior.
p	–	Usually joint or joint conditional distribution of $\mathbf{x}, \mathbf{n}, \mathbf{h}$.
p_l	–	Linearized joint or joint conditional distribution of $\mathbf{x}, \mathbf{n}, \mathbf{h}$.
q	–	Approximate joint distribution of $\mathbf{x}, \mathbf{n}, \mathbf{h}$ used in Variational Inference.
s	–	Designates the state of an HMM or the mixture component of a Gaussian Mixture Model.
η	–	The mean vector of the q distribution.
ρ	–	The component weights of the q distribution.
$\mathbf{\Phi}$	–	The covariance matrix of a component of the q distribution.
μ	–	The mean vector of a distribution, usually of $p(x)$ and or $p(n)$.
π	–	Mixture weight of a distribution, usually of $p(x)$ and or $p(n)$.
$\mathbf{\Sigma}$	–	The covariance of a distribution, usually of $p(x)$ and or $p(n)$.
$\mathbf{\Psi}$	–	The covariance of the interaction equation.
$\mathcal{F}$	–	The negative relative entropy.
$\mathcal{K}$	–	Relative entropy.

Appendix B

Useful Identities

B.1 Gaussian Quadratic Integral Forms

The expectation of a quadratic form under a Gaussian is another quadratic form [70]. If $\mathbf{x}$ is Gaussian distributed with mean $\boldsymbol{\eta}$ and variance $\boldsymbol{\Phi}$ then,

$$\int_x (\mathbf{x}-\boldsymbol{\mu})^T \boldsymbol{\Sigma}^{-1}(\mathbf{x}-\boldsymbol{\mu}) N(\mathbf{x};\boldsymbol{\eta},\boldsymbol{\Phi}) = (\boldsymbol{\mu}-\boldsymbol{\eta})^T \boldsymbol{\Sigma}^{-1}(\boldsymbol{\mu}-\boldsymbol{\eta}) + Tr[\boldsymbol{\Sigma}^{-1}\boldsymbol{\Phi}] \quad \text{(A-1)}$$

If the original quadratic form has a linear function of $\mathbf{x}$, the result it still simple:

$$\int_x (\mathbf{W}\mathbf{x}-\boldsymbol{\mu})^T \boldsymbol{\Sigma}^{-1}(\mathbf{W}\mathbf{x}-\boldsymbol{\mu}) N(\mathbf{x};\boldsymbol{\eta},\boldsymbol{\Phi})$$
$$= (\boldsymbol{\mu}-\mathbf{W}\boldsymbol{\eta})^T \boldsymbol{\Sigma}^{-1}(\boldsymbol{\mu}-\mathbf{W}\boldsymbol{\eta}) + Tr[\mathbf{W}^T\boldsymbol{\Sigma}^{-1}\mathbf{W}\boldsymbol{\Phi}] \quad \text{(A-2)}$$

B.2 Multiplication of Two Gaussians

The multiplication of two Gaussians $N(\mathbf{a},\mathbf{A})$ and $N(\mathbf{b},\mathbf{B})$ is:

$$N(\mathbf{a},\mathbf{A}) \cdot N(\mathbf{b},\mathbf{B}) = z_c N(\mathbf{c},\mathbf{C}) \quad \text{(A-3)}$$

where

$$\mathbf{C} = (\mathbf{A}^{-1} + \mathbf{B}^{-1})^{-1} \tag{A-4}$$

$$\mathbf{c} = \mathbf{C}\mathbf{A}^{-1}\mathbf{a} + \mathbf{C}\mathbf{B}^{-1}\mathbf{b} \tag{A-5}$$

and

$$z_c = (2\pi)^{-d/2}|\mathbf{C}|^{+1/2}|\mathbf{A}|^{-1/2}|\mathbf{B}|^{-1/2}\exp\left[-\frac{1}{2}\left(\mathbf{a}^T\mathbf{A}^{-1}\mathbf{a} + \mathbf{b}^T\mathbf{B}^{-1}\mathbf{b} - \mathbf{c}^T\mathbf{C}^{-1}\mathbf{c}\right)\right] \tag{A-6}$$

where d is the dimension of $\mathbf{a}$. In the univariate case, we rewrite succinctly:

$$N(y;a,A)N(y;b,B) = N(y;c,C)N(a;b,A+B) \tag{A-7}$$

B.3 Change of Variables

The Gaussian random variable $\mathbf{y}$ with mean $\mathbf{d} + \mathbf{D}\mathbf{z}$ can be rewritten as a distribution over $\mathbf{z}$:

$$N(\mathbf{y};\mathbf{d}+\mathbf{D}\mathbf{z},\mathbf{E}) = K \cdot N\left[\mathbf{z};(\mathbf{D}^T\mathbf{E}^{-1}\mathbf{D})^{-1}\mathbf{D}^T\mathbf{E}^{-1}(\mathbf{y}-\mathbf{d}),(\mathbf{D}^T\mathbf{E}^{-1}\mathbf{D})^{-1}\right] \tag{A-8}$$

where

$$K = |2\pi\mathbf{E}|^{-1/2} \cdot |2\pi(\mathbf{D}^T\mathbf{E}\mathbf{D})^{-1}|^{1/2} \tag{A-9}$$

Note that $\mathbf{D}^T\mathbf{E}^{-1}\mathbf{D}$ will be singular if the dimensions of $\mathbf{z}$ are higher than the dimensions of $\mathbf{y}$, i.e. this expression is ill defined.

B.4 Multiplication of Linear Likelihood and Gaussian

The following identity allows us to rewrite the product of the Gaussian speech prior $p(x)$, Gaussian noise prior $p(n)$ and the Gaussian likelihood $p(y|x,n)$ as a singe Gaussian times a scaling factor.

$$N(\mathbf{z};\mathbf{a},\mathbf{A})N(\mathbf{y};\mathbf{d}+\mathbf{D}\mathbf{z},\mathbf{E}) = L\cdot N(\mathbf{z};\mathbf{c},\mathbf{C}) \tag{A-10}$$

where

$$\mathbf{C} = (\mathbf{A}^{-1}+\mathbf{D}^T\mathbf{E}^{-1}\mathbf{D})^{-1} \tag{A-11}$$

$$\mathbf{c} = \mathbf{C}(\mathbf{A}^{-1}\mathbf{a}+\mathbf{D}^T\mathbf{E}^{-1}(\mathbf{y}-\mathbf{d})) \tag{A-12}$$

and

$$L = (2\pi)^{\frac{d_z-d_y}{2}}|\mathbf{E}|^{-1/2}|\mathbf{C}|^{1/2}|\mathbf{A}|^{-1/2} \\ \exp\left[-\frac{1}{2}\left(\mathbf{a}^T\mathbf{A}^{-1}\mathbf{a}+(\mathbf{y}-\mathbf{d})^T\mathbf{E}^{-1}(\mathbf{y}-\mathbf{d})-\mathbf{c}^T\mathbf{C}^{-1}\mathbf{c}\right)\right] \tag{A-13}$$

where d_z is the dimension of $\mathbf{z}$ and d_y is the dimension of $\mathbf{y}$.

B.5 Matrix Calculus Results

The following identities are useful when taking derivatives of the negative relative entropy.

Matrix derivative of trace:

$$\frac{\delta}{\delta\mathbf{X}}Tr[\mathbf{X}\mathbf{A}] = \mathbf{A} \tag{A-14}$$

Matrix derivative of determinant:

$$\frac{\delta}{\delta\mathbf{X}}|\mathbf{X}| = |\mathbf{X}|\mathbf{X}^{-T} \tag{A-15}$$

Vector derivative:

$$\frac{\delta}{\delta\mathbf{x}}\mathbf{a^T x b} = \mathbf{b^T a} \tag{A-16}$$

Vector derivative of quadratic form:

$$\frac{\delta}{\delta\mathbf{x}}(\mathbf{A}\mathbf{x}+\mathbf{b})^T\mathbf{C}(\mathbf{D}\mathbf{x}+\mathbf{e}) = \mathbf{A}^T\mathbf{C}(\mathbf{D}\mathbf{x}+\mathbf{e})+\mathbf{D}^T\mathbf{C}^T(\mathbf{A}\mathbf{x}+\mathbf{b}) \tag{A-17}$$

For symmetric matrices (e.g. covariance matrices) then:

$$\frac{\delta}{\delta \mathbf{x}}(\mathbf{A}\mathbf{x}+\mathbf{b})^T\mathbf{C}(\mathbf{A}\mathbf{x}+\mathbf{b}) = 2\mathbf{A}^T\mathbf{C}(\mathbf{A}\mathbf{x}+\mathbf{b}) \tag{A-18}$$

Appendix C

Derivations

C.1 Log-Spectrum Domain Algonquin

Here we derive the Algonquin algorithm in the log-spectrum domain using the variational approach. This derivations is for and unfactorized q. Derivations for factorized versions of q follow easily. Derivations of Algonquin on the spectrum domain as well as the cepstrum domain are completely analogous.

We first define the $d \times 1$ column vectors $\mathbf{x}, \mathbf{n}, \mathbf{h}$ and $\mathbf{y}$ as the feature vectors for clean speech, noise, the transfer function and corrupted speech, respectively.

The joint distribution over the noisy speech, clean speech, noise, channel distortion, speech class and noise class is:

$$\begin{aligned} &p(\mathbf{y}, \mathbf{x}, \mathbf{n}, \mathbf{h}, s^x, s^n, s^h) \\ &\qquad = N(\mathbf{y}; g(\mathbf{z}), \boldsymbol{\Psi})\pi_{s^x} N(\mathbf{x}; \boldsymbol{\mu}^s_{s^x}, \boldsymbol{\Sigma}^s_{s^x})\pi_{s^n} N(\mathbf{n}; \boldsymbol{\mu}^n_{s^n}, \boldsymbol{\Sigma}^n_{s^n})\pi_{s^h} N(\mathbf{h}; \boldsymbol{\mu}^h_{s^h}, \boldsymbol{\Sigma}^h_{s^h}) \quad \text{(A-1)} \end{aligned}$$

where $\boldsymbol{\mu}^x_{s^x}$ is a $d \times 1$ vector of speech means, and $\boldsymbol{\Sigma}^s_{s^x}$ is a (diagonal) $d \times d$ covariance matrix for speech, and similarly for the noise and channel function. π_{s^x} is that scalar mixture weight for speech.

Defining $3d \times 1$ vector $\mathbf{z} = [\mathbf{x}^T\ \mathbf{n}^T\ \mathbf{h}^T]^T$ and $s = \{s^x, s^n, s^h\}$ we write succinctly:

$$p(\mathbf{y}, \mathbf{z}, s) = N(\mathbf{y}; g(\mathbf{z}), \boldsymbol{\Psi})\pi_s N(\mathbf{z}; \boldsymbol{\mu}_s, \boldsymbol{\Sigma}_s) \qquad \text{(A-2)}$$

Since we will have to integrate over $\mathbf{z}$ we linearize the $d \times 1$ function $\mathbf{g}(\mathbf{z})$ using the first order Taylor series.

We require the function and it's derivatives:

$$\mathbf{g}(\mathbf{x}, \mathbf{n}, \mathbf{h}) = \mathbf{x} + \mathbf{h} + \ln(1 + \exp(\mathbf{n} - \mathbf{x} - \mathbf{h})) \qquad \text{(A-3)}$$

Taking the derivative of the $i-$th component of the vector g with respect to the $j-$th component of x_j we find

$$\frac{dg(x_i, n_i, h_i)}{dx_j} = \frac{1}{1 + \exp(n_i - x_i - h_i)} \qquad \text{(A-4)}$$

if $i = j$ and 0 otherwise. Similarly, the derivative with respect to n_j is:

$$\frac{dg(x_i, n_i, h_i)}{dx_j} = \frac{\exp(n_i - x_i - h_i)}{1 + \exp(n_i - x_i - h_i)} \qquad \text{(A-5)}$$

The derivative with respect to h_i is equal to the derivative with respect to x_i.

We define the $d \times d$ matrices:

$$\mathbf{G}_x(\mathbf{x}, \mathbf{n}, \mathbf{h}) = \frac{d\mathbf{g}(\mathbf{x}, \mathbf{n}, \mathbf{h})}{d\mathbf{x}} = diag\left[\frac{dg(x_1, n_1, h_1)}{dx_1} \dots \frac{dg(x_d, n_d, h_d)}{dx_d}\right] \qquad \text{(A-6)}$$

$$\mathbf{G}_n(\mathbf{x}, \mathbf{n}, \mathbf{h}) = \frac{d\mathbf{g}(\mathbf{x}, \mathbf{n}, \mathbf{h})}{d\mathbf{n}} = diag\left[\frac{dg(x_1, n_1, h_1)}{dn_1} \dots \frac{dg(x_d, n_d, h_d)}{dn_d}\right] \qquad \text{(A-7)}$$

$$\mathbf{G}_n(\mathbf{x}, \mathbf{n}, \mathbf{h}) = \frac{d\mathbf{g}(\mathbf{x}, \mathbf{n}, \mathbf{h})}{d\mathbf{h}} = diag\left[\frac{dg(x_1, n_1, h_1)}{dh_1} \dots \frac{dg(x_d, n_d, h_d)}{dh_d}\right] \qquad \text{(A-8)}$$

and for notational purposes:

$$\mathbf{G}(\mathbf{z}) = \frac{d\mathbf{g}(\mathbf{z})}{d\mathbf{z}} = \Big[\mathbf{G}_x(\mathbf{x}, \mathbf{n}, \mathbf{h});\quad \mathbf{G}_n(\mathbf{x}, \mathbf{n}, \mathbf{h});\quad \mathbf{G}_h(\mathbf{x}, \mathbf{n}, \mathbf{h})\Big] \qquad \text{(A-9)}$$

which is $d \times 3d$.

Thus, the vector Taylor series expansion of $\mathbf{g}(\mathbf{z})$ is:

$$\mathbf{g}_l(\mathbf{z}) = \mathbf{g}(\mathbf{z}_0) + \mathbf{G}(\mathbf{z}_0)(\mathbf{z} - \mathbf{z}_0) \tag{A-10}$$

Now we can finally write the linearized posterior function:

$$p_l(\mathbf{y}, \mathbf{z}, s) = N(\mathbf{y}; \mathbf{g}(\mathbf{z}_0) + \mathbf{G}(\mathbf{z}_0)(\mathbf{z} - \mathbf{z}_0), \mathbf{\Psi})\pi_s N(\mathbf{z}; \mu_s, \Sigma_s) \tag{A-11}$$

note that the linearized version of the posterior is a function of the linearization point $\mathbf{z}_0$.

The most general form of the approximate posterior is:

$$q(\mathbf{x}, \mathbf{n}, \mathbf{h}|s^x, s^n, s^h) = N\left(\begin{bmatrix}\mathbf{x}\\ \mathbf{n}\\ \mathbf{h}\end{bmatrix}; \begin{bmatrix}\boldsymbol{\eta}^x_{s^x,s^n,s^h}\\ \boldsymbol{\eta}^n_{s^x,s^n,s^h}\\ \boldsymbol{\eta}^h_{s^x,s^n,s^h}\end{bmatrix}, \begin{bmatrix}\mathbf{\Phi}^{xx}_{s^x,s^n,s^h} & \mathbf{\Phi}^{xn}_{s^x,s^n,s^h} & \mathbf{\Phi}^{xh}_{s^x,s^n,s^h}\\ \mathbf{\Phi}^{xn}_{s^x,s^n,s^h} & \mathbf{\Phi}^{nn}_{s^x,s^n,s^h} & \mathbf{\Phi}^{nh}_{s^x,s^n,s^h}\\ \mathbf{\Phi}^{xh}_{s^x,s^n,s^h} & \mathbf{\Phi}^{nh}_{s^x,s^n,s^h} & \mathbf{\Phi}^{hh}_{s^x,s^n,s^h}\end{bmatrix}\right) \tag{A-12}$$

or

$$q(\mathbf{z}|s) = N(\mathbf{z}; \boldsymbol{\eta}_s, \mathbf{\Phi}_s) \tag{A-13}$$

This form assumes that $\mathbf{x}$, $\mathbf{n}$ and $\mathbf{h}$ are jointly Gaussian, and is exactly equivalent to the linearized posterior. Cutting links in the factor graph for the q function produces "factorized" versions of the posterior.

The posterior mixing proportions for classes s^x, s^n and s^h are $q(s^x, s^n, s^h) = \rho_{s^x s^n, s^h}$ or succinctly ρ_s. The approximate posterior is given by $q(\mathbf{x}, \mathbf{n}, \mathbf{h}) = \sum_{s^x,s^n,s^h} q(s^x, s^n, s^h)q(\mathbf{x}, \mathbf{n}, \mathbf{h}|s^x, s^n, s^h)$, i.e. $q(\mathbf{z}) = \sum_s \rho_s q(\mathbf{z}|s)$.

Having defined the exact form of the approximate posterior q and the linearized posterior p_l we can write the free energy:

$$F = \sum_s \int q(\mathbf{z}, s) \ln p_l(\mathbf{y}, \mathbf{z}, s) d\mathbf{z} - \sum_s \int q(\mathbf{z}, s) \ln q(\mathbf{z}, s) d\mathbf{z} \tag{A-14}$$

In order to derive the re-estimation formulas, we need to take the derivative of F with respect to the parameters of q. But we first need to integrate over $\mathbf{z}$. We rewrite Equation (A-14) and look at each of the five terms separately.

$$\begin{aligned} F &= \sum_s \int_z q(\mathbf{z}, s) \ln N(\mathbf{y}; \mathbf{g}(\mathbf{z}_0) + \mathbf{G}(\mathbf{z}_0)(\mathbf{z} - \mathbf{z}_0), \mathbf{\Psi}) && \text{(A-15)} \\ &+ \sum_s \int_z q(\mathbf{z}, s) \ln \pi_s && \text{(A-16)} \\ &+ \sum_s \int_z q(\mathbf{z}, s) \ln N(\mathbf{z}; \mu_s, \Sigma_s) && \text{(A-17)} \\ &- \sum_s \int_z q(\mathbf{z}, s) \ln \rho_s && \text{(A-18)} \\ &- \sum_s \int_z q(\mathbf{z}, s) \ln N(\mathbf{z}; \eta_s, \Phi_s) && \text{(A-19)} \end{aligned}$$

C.1.1 First Term: Equation (A-15)

Let's look at the first term:

$$\begin{aligned} &\sum_s \int q(\mathbf{z}, s) \ln N(\mathbf{y}; \mathbf{g}(\mathbf{z}_0) + \mathbf{G}(\mathbf{z}_0)(\mathbf{z} - \mathbf{z}_0), \mathbf{\Psi}) d\mathbf{z} && \text{(A-20)} \\ &\quad = -\frac{1}{2} \sum_s \rho_s \ln |2\pi \mathbf{\Psi}| && \text{(A-21)} \\ &\quad -\frac{1}{2} \sum_s \int \Big[\rho_s N(\mathbf{z}; \boldsymbol{\eta}_s, \mathbf{\Phi}_s) \cdot \\ &\quad (\mathbf{y} - \mathbf{g}(\mathbf{z}_0) - \mathbf{G}(\mathbf{z}_0)(\mathbf{z} - \mathbf{z}_0))^T \mathbf{\Psi}^{-1} (\mathbf{y} - \mathbf{g}(\mathbf{z}_0) - \mathbf{G}(\mathbf{z}_0)(\mathbf{z} - \mathbf{z}_0))) \Big] d\mathbf{z} && \text{(A-22)} \end{aligned}$$

Now we look at (A-22) which contains an integral of a Gaussian and a quadratic:

$$\begin{aligned} &\int \Big[N(\mathbf{z}; \boldsymbol{\eta}_s, \mathbf{\Phi}_s) \cdot \\ &\quad (\mathbf{y} - \mathbf{g}(\mathbf{z}_0) - \mathbf{G}(\mathbf{z}_0)(\mathbf{z} - \mathbf{z}_0))^T \mathbf{\Psi}^{-1} (\mathbf{y} - \mathbf{g}(\mathbf{z}_0) - \mathbf{G}(\mathbf{z}_0)(\mathbf{z} - \mathbf{z}_0)) \Big] d\mathbf{z} && \text{(A-23)} \end{aligned}$$

$$\int \Big[N(\mathbf{z}; \boldsymbol{\eta}_s, \boldsymbol{\Phi}_s) \cdot$$
$$(\mathbf{G}(\mathbf{z}_0)\mathbf{z} - (\mathbf{y}_{obs} - \mathbf{g}(\mathbf{z}_0) + \mathbf{G}(\mathbf{z}_0)\mathbf{z}_0))^T \boldsymbol{\Psi}^{-1} (\mathbf{G}(\mathbf{z}_0)\mathbf{z} - (\mathbf{y}_{obs} - \mathbf{g}(\mathbf{z}_0) + \mathbf{G}(\mathbf{z}_0)\mathbf{z}_0) \Big] d\mathbf{z} \tag{A-24}$$

and using identity A-2 we get

$$((\mathbf{y}_{obs} - \mathbf{g}(\mathbf{z}_0) + \mathbf{G}(\mathbf{z}_0)(\mathbf{z}_0 - \boldsymbol{\eta}_s))^T \boldsymbol{\Psi}^{-1} ((\mathbf{y}_{obs} - \mathbf{g}(\mathbf{z}_0) + \mathbf{G}(\mathbf{z}_0)(\mathbf{z}_0 - \boldsymbol{\eta}_s))$$
$$+ Tr[\mathbf{G}(\mathbf{z}_0)^T \boldsymbol{\Psi}^{-1} \mathbf{G}(\mathbf{z}_0) \boldsymbol{\Phi}_s] \tag{A-25}$$

We can now plug this in to Equation (A-22) to complete the first term:

$$\sum_s \int q(\mathbf{z}, s) \ln N(\mathbf{y}; \mathbf{g}(\mathbf{z}_0) + \mathbf{G}(\mathbf{z}_0)(\mathbf{z} - \mathbf{z}_0), \boldsymbol{\Psi}) d\mathbf{z}$$
$$= -\frac{1}{2} \sum_s \rho_s \ln |2\pi \boldsymbol{\Psi}|$$
$$- \sum_s \rho_s ((\mathbf{y}_{obs} - \mathbf{g}(\mathbf{z}_0) + \mathbf{G}(\mathbf{z}_0)(\mathbf{z}_0 - \boldsymbol{\eta}_s))^T \boldsymbol{\Psi}^{-1} ((\mathbf{y}_{obs} - \mathbf{g}(\mathbf{z}_0) + \mathbf{G}(\mathbf{z}_0)(\mathbf{z}_0 - \boldsymbol{\eta}_s))$$
$$+ \sum_s \rho_s Tr[\mathbf{G}(\mathbf{z}_0)^T \boldsymbol{\Psi}^{-1} \mathbf{G}(\mathbf{z}_0) \boldsymbol{\Phi}_s]. \tag{A-26}$$

The remaining terms are much easier to handle.

C.1.2 Second Term: Equation (A-16)

The second term is simple:

$$\sum_s \int [q(\mathbf{z}, s) \ln \pi_s] d\mathbf{z} = \sum_s \rho_s \ln \pi_s \tag{A-27}$$

C.1.3 Third Term: Equation (A-17)

The third term has the same form as the first term:

$$\sum_s \int_z q(\mathbf{z}, s) \ln N(\mathbf{z}; \boldsymbol{\mu}_s, \boldsymbol{\Sigma}_s) \tag{A-28}$$

$$= \sum_s \int_z \rho_s N(\mathbf{z}; \boldsymbol{\eta}_s, \boldsymbol{\Phi}_s) \ln |2\pi \boldsymbol{\Sigma}_s|^{-\frac{1}{2}} \tag{A-29}$$

$$- \frac{1}{2} \sum_s \int_z \rho_s N(\mathbf{z}; \boldsymbol{\eta}_s, \boldsymbol{\Phi}_s)(\mathbf{z} - \boldsymbol{\mu}_s)^T \boldsymbol{\Sigma}_s^{-1} (\mathbf{z} - \boldsymbol{\mu}_s) \tag{A-30}$$

$$= -\frac{1}{2} \sum_s \rho_s \ln |2\pi \boldsymbol{\Sigma}_s| \tag{A-31}$$

$$- \frac{1}{2} \sum_s \rho_s (\boldsymbol{\mu}_s - \boldsymbol{\eta}_s)^T \boldsymbol{\Sigma}_s^{-1} (\boldsymbol{\mu}_s - \boldsymbol{\eta}_s) + \sum_s Tr[\boldsymbol{\Sigma}_s^{-1} \boldsymbol{\Phi}_s], \tag{A-32}$$

where we have used identity A-2 in the last step.

C.1.4 Fourth Term: Equation (A-18)

The fourth term is similar to the second term.

$$\sum_s \int q(\mathbf{z}, s) \ln \rho_s d\mathbf{z} = \sum_s \rho_s \ln \rho_s \tag{A-33}$$

C.1.5 Fifth Term: Equation (A-19)

$$\sum_s \int_z q(\mathbf{z}, s) \ln N(\mathbf{z}; \boldsymbol{\eta}_s, \boldsymbol{\Phi}_s) \tag{A-34}$$

$$= \sum_s \int_z \rho_s N(\mathbf{z}; \boldsymbol{\eta}_s, \boldsymbol{\Phi}_s) \ln |2\pi \boldsymbol{\Phi}_s|^{-\frac{1}{2}} \tag{A-35}$$

$$- \frac{1}{2} \sum_s \int_z \rho_s N(\mathbf{z}; \boldsymbol{\eta}_s, \boldsymbol{\Phi}_s)(\mathbf{z} - \boldsymbol{\eta}_s)^T \boldsymbol{\Phi}_s^{-1} (\mathbf{z} - \boldsymbol{\eta}_s) \tag{A-36}$$

$$= -\frac{1}{2} \sum_s \rho_s \ln |2\pi \boldsymbol{\Phi}_s| \tag{A-37}$$

$$- \frac{1}{2} \sum_s \rho_s (\boldsymbol{\eta}_s - \boldsymbol{\eta}_s)^T \boldsymbol{\Phi}_s^{-1} (\boldsymbol{\eta}_s - \boldsymbol{\eta}_s) + \sum_s Tr[\boldsymbol{\Phi}_s^{-1} \boldsymbol{\Phi}_s] \tag{A-38}$$

$$= -\frac{1}{2} \sum_s \rho_s \ln |2\pi \boldsymbol{\Phi}_s| + 3d \tag{A-39}$$

where the term (A-38) simplifies considerably.

C.1.6 The Negative Relative Entropy F

Now we can gather the above terms to write the negative relative entropy:

$$\begin{aligned}
F =& -\frac{1}{2}\sum_s \rho_s \ln|2\pi\mathbf{\Psi}| \\
& - \sum_s \rho_s \big[(\mathbf{y}_{obs} - \mathbf{g}(\mathbf{z}_0) + \mathbf{G}(\mathbf{z}_0)(\mathbf{z}_0 - \boldsymbol{\eta}_s))^T \mathbf{\Psi}^{-1} \\
& \qquad ((\mathbf{y}_{obs} - \mathbf{g}(\mathbf{z}_0) + \mathbf{G}(\mathbf{z}_0)(\mathbf{z}_0 - \boldsymbol{\eta}_s)\big] && \text{(A-40)} \\
& + \sum_s \rho_s Tr[\mathbf{G}(\mathbf{z}_0)^T \mathbf{\Psi}^{-1} \mathbf{G}(\mathbf{z}_0) \mathbf{\Phi}_s] && \text{(A-41)} \\
& + \sum_s \rho_s \ln \pi_s && \text{(A-42)} \\
& - \frac{1}{2}\sum_s \rho_s \ln|(2\pi)\mathbf{\Sigma}_s| \\
& - \frac{1}{2}\sum_s \rho_s \left\{(\boldsymbol{\mu}_s - \boldsymbol{\eta}_s)^T \mathbf{\Sigma}_s^{-1}(\boldsymbol{\mu}_s - \boldsymbol{\eta}_s) + Tr[\mathbf{\Sigma}_s^{-1}\mathbf{\Phi}_s]\right\} && \text{(A-43)} \\
& - \sum_s \rho_s \ln \rho_s && \text{(A-44)} \\
& + \frac{1}{2}\sum_s \rho_s \ln|2\pi\mathbf{\Phi}_s| - 3d && \text{(A-45)}
\end{aligned}$$

To derive the re-estimation formulas, we differentiate w.r.t the parameters of q.

C.1.7 Derivation of $\boldsymbol{\eta}$

Define $\mathbf{G} = \mathbf{G}(\mathbf{z}_0)$ and $\mathbf{g} = \mathbf{g}(\mathbf{z}_0)$. We differentiate with respect to $\boldsymbol{\eta}_s$ and equate to zero:

$$\begin{aligned}
0 &= \frac{\delta F}{\delta \boldsymbol{\eta}_s} \\
&= \frac{\delta}{\delta \boldsymbol{\eta}_s}\Big(-\frac{1}{2}\sum_s \rho_s \left\{ (\boldsymbol{\mu}_s - \boldsymbol{\eta}_s)^T \boldsymbol{\Sigma}^{-1} (\boldsymbol{\mu}_s - \boldsymbol{\eta}_s) \right\} \\
&\quad -\frac{1}{2}\sum_s \rho_s \left[(\mathbf{y}_{obs} - \mathbf{g} + \mathbf{G}(\mathbf{z}_0 - \boldsymbol{\eta}_s))^T \boldsymbol{\Psi}^{-1} (\mathbf{y}_{obs} - \mathbf{g} + \mathbf{G}(\mathbf{z}_0 - \boldsymbol{\eta}_s)) \right] \Big) \\
&= -\rho_s \boldsymbol{\Sigma}^{-1} (\boldsymbol{\mu}_s - \boldsymbol{\eta}_s) && \text{(A-46)} \\
&\quad -\rho_s \mathbf{G}^T \boldsymbol{\Psi}^{-1} (\mathbf{y}_{obs} - \mathbf{g} + \mathbf{G}(\mathbf{z}_0 - \boldsymbol{\eta}_s)), && \text{(A-47)}
\end{aligned}$$

where we have used identity A-18 in the last step. Now we solve for $\boldsymbol{\eta}_s$ and we find

$$\boldsymbol{\eta}_s = \left(\boldsymbol{\Sigma}^{-1} + \mathbf{G}^T \boldsymbol{\Psi}^{-1} \mathbf{G} \right)^{-1} \left[\boldsymbol{\Sigma}^{-1} \boldsymbol{\mu}_s + \mathbf{G}^T \boldsymbol{\Psi}^{-1} (\mathbf{y}_{obs} - \mathbf{g} + \mathbf{G}\mathbf{z}_0) \right] \quad \text{(A-48)}$$

Alternately (add $\boldsymbol{\Sigma}^{-1}\mathbf{z}_0 - \boldsymbol{\Sigma}^{-1}\mathbf{z}_0$ term and re-arrange) we can write

$$\boldsymbol{\eta}_s = \mathbf{z}_0 + \boldsymbol{\Phi} \left[\boldsymbol{\Sigma}^{-1} (\boldsymbol{\mu}_s - \mathbf{z}_0) + \mathbf{G} \boldsymbol{\Psi}^{-1} (\mathbf{y} - \mathbf{g}) \right]. \quad \text{(A-49)}$$

C.1.8 Derivation of $\boldsymbol{\Phi}$

To find the variance, we differentiate F and set to 0:

$$\begin{aligned}
0 &= \frac{\delta F}{\delta \boldsymbol{\Phi}_s} \\
&= \frac{\delta}{\delta \boldsymbol{\Phi}_s}\Big(-\frac{1}{2}\sum_s \rho_s Tr[\boldsymbol{\Sigma}_s^{-1} \boldsymbol{\Phi}_s] - \frac{1}{2}\sum_s \rho_s Tr[\mathbf{G}^T \boldsymbol{\Psi}^{-1} \mathbf{G} \boldsymbol{\Phi}_s] && \text{(A-50)} \\
&+ \frac{1}{2}\sum_s \rho_s \ln |2\pi \boldsymbol{\Phi}_s| \Big) && \text{(A-51)}
\end{aligned}$$

And using identities for derivatives of traces A-14 and determinants A-15, we arrive at

$$0 = -\frac{1}{2}\rho_s(\mathbf{\Sigma}_s^{-1})^T - \frac{1}{2}\rho_s[\mathbf{G}^T\mathbf{\Psi}^{-1}\mathbf{G}]^T + \frac{1}{2}\rho_s|\mathbf{\Phi}_s|^{-1}|\mathbf{\Phi}_s|(\mathbf{\Phi}_s^{-1})^T. \tag{A-52}$$

And therefore:

$$\mathbf{\Phi}_s = \left(\mathbf{\Sigma}_s^{-1} + \mathbf{G}^T\mathbf{\Psi}^{-1}\mathbf{G}\right)^{-1}. \tag{A-53}$$

C.1.9 Derivation of ρ

When finding ρ_s we need to ensure that $\sum_s \rho_s = 1$ so we use a Lagrange multiplier $\lambda(\sum_s \rho_s - 1))$. Again, differentiate and set to 0:

$$\begin{aligned}
0 &= \frac{\delta(F - \lambda(\sum_s \rho_s - 1))}{\delta\rho_s} \\
&= \lambda + \ln \pi_s \\
&- \frac{1}{2}\ln|2\pi\mathbf{\Sigma}_s| \\
&- \frac{1}{2}\left\{(\boldsymbol{\mu}_s - \boldsymbol{\eta}_s)^T\mathbf{\Sigma}^{-1}(\boldsymbol{\mu}_s - \boldsymbol{\eta}_s) + Tr[\mathbf{\Sigma}_s^{-1}\mathbf{\Phi}_s]\right\} \\
&- \frac{1}{2}\ln|2\pi\mathbf{\Psi}| \\
&- \frac{1}{2}\left\{(\mathbf{y} - \mathbf{g} + \mathbf{G}\mathbf{z}_0 - \mathbf{G}\boldsymbol{\eta}_s)^T\mathbf{\Psi}^{-1}(\mathbf{y} - \mathbf{g} + \mathbf{G}\mathbf{z}_0 - \mathbf{G}\boldsymbol{\eta}_s) + Tr[\mathbf{G}^T\mathbf{\Psi}^{-1}\mathbf{G}\mathbf{\Phi}_s]\right\} \\
&- (\ln\rho_s + 1) \\
&+ \frac{1}{2}\ln|2\pi\mathbf{\Phi}_s|.
\end{aligned} \tag{A-54}$$

Notice that

$$Tr[\mathbf{\Sigma}_s^{-1}\mathbf{\Phi}_s] + Tr[\mathbf{G}^T\mathbf{\Psi}^{-1}\mathbf{G}\mathbf{\Phi}_s] = Tr[(\mathbf{\Sigma}_s^{-1} + \mathbf{G}^T\mathbf{\Psi}^{-1}\mathbf{G})\mathbf{\Phi}_s] = Tr[I]. \tag{A-55}$$

Define:

$$\begin{aligned}
L_s &= \ln \pi_s \\
&- \frac{1}{2}\ln|2\pi\boldsymbol{\Sigma}_s| + \frac{1}{2}\ln|2\pi\Phi_s| \\
&- \frac{1}{2}\left\{(\boldsymbol{\mu}_s - \boldsymbol{\eta}_s)^T\boldsymbol{\Sigma}^{-1}(\boldsymbol{\mu}_s - \boldsymbol{\eta}_s)\right\} \\
&- \frac{1}{2}\left[(\mathbf{y} - \mathbf{g} + \mathbf{G}\mathbf{z}_0 - \mathbf{G}\boldsymbol{\eta}_s)^T\boldsymbol{\Psi}^{-1}(\mathbf{y} - \mathbf{g} + \mathbf{G}\mathbf{z}_0 - \mathbf{G}\boldsymbol{\eta}_s)\right] \\
&+ \mathit{const},
\end{aligned} \tag{A-56}$$

where all terms not dependent on s are gathered into the $const$ term. The $const$ term in Equation (A-56) cancels when computing ρ_s below. Now rewrite Equation (A-54) and solve for ρ_s:

$$\rho_s = \exp(\lambda)\exp(L_s) \tag{A-57}$$

It follows that

$$\sum_s \rho_s = \exp(\lambda)\sum_s \exp(L_s). \tag{A-58}$$

And finally, solving for $\exp(\lambda)$ and plugging back into Equation (A-57) we get

$$\rho_s = \frac{\exp(L_s)}{\sum_i \exp(L_i)}. \tag{A-59}$$

The form of L_s differs from that derived in 8.17. However, when q is unfactorized these two forms are equivalent.

The above derivation can be used as a template to derive the Algonquin algorithm in the spectrum and cepstrum domains. The spectrum domain version suffers from the problem of returning negative estimates of clean speech, similarly to Spectral Subtraction. The cepstrum domain version produces superior recognition results (see Table D.2) to the log-spectrum version, but is considerably slower.

Appendix D

Results

This Appendix contains complete details of most of the results reported in the Thesis. In addition, some results have been included that were not directly pertinent to the discussion, but may be of interest. These include results for different feature sets (see Figure D.1), as well as results for using the Algonquin algorithm in the cepstrum domain (see Table D.2).

D.1 Comparison of Log-Spectrum and Cepstrum Features

Figure D.1 shows the accuracy of four different feature sets. The effectiveness of MFCC features with delta and acceleration is clear from the plot. Notice that the performance of all features for clean speech is similar. At 10dB SNR, using Delta and Acceleration features gives substantial gains, while using Filter Bank features is considerably worse.

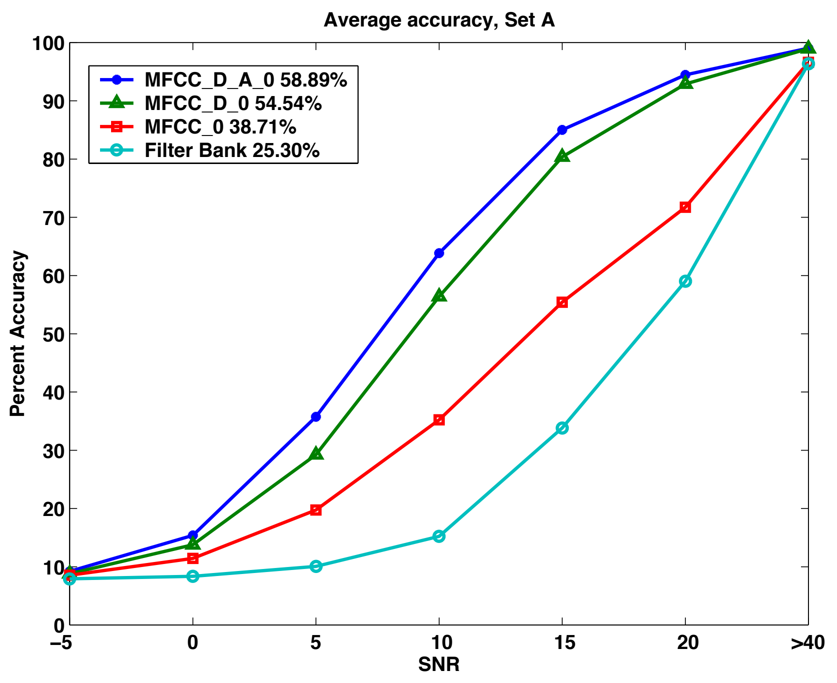


Figure D.1: The plot shows the results for using MFCC features without delta and acceleration, with delta but not acceleration and with delta and acceleration, and Filter bank features.

D.2 Multicondition Results

An effective way of dealing with noisy speech is simply to train on the noisy data. This is called *matched training*. Recently, it has become clear that one can obtain even better results by first running a cleaning algorithm on the *multicondition* training set, which consists of speech files with noise at different SNRs, and different types of noise. Then the acoustic models are re-trained on the cleaned training set. This is called *multicondition* training.

In Table D.1 we show the result of using Algonquin (256 mixture speech model, 1 mixture noise model) to clean the multicondition training set, and then use the new acoustic models for recognizing set A. Note that we are using the same cleaned files as in Table 8.1. The average accuracy increases to 86.57%. This is a reduction in Word Error Rate from the baseline (82.22%) of 24.5%.

	Subway	Car	Babble	Exhib.	Average
Clean	98.83	98.43	98.30	98.77	98.58
20 dB	97.42	97.37	98.15	97.38	97.58
15 dB	95.64	95.68	97.88	95.68	96.22
10 dB	91.07	92.41	95.62	92.44	92.89
5 dB	80.84	81.86	90.07	85.87	84.66
0 dB	55.36	52.51	68.86	69.30	61.51
-5 dB	25.24	19.26	29.14	39.46	28.27
Average	84.07	83.97	90.12	88.13	86.57

Table D.1: Accuracy for Set A. Log-Spectrum domain MMSE-Algorithm. Recognition using models trained on multicondition data. The training set was cleaned using a single mixture noise model estimated using the first 20 frames. A 256 mixture speech model was used.

D.3 Cepstrum Domain MMSE Algonquin

The Algonquin algorithm can be formulated in the cepstrum domain. The recognition results for this algorithm are shown in Table D.2. The recognition result of 85.25% is comparable to an accuracy of 82.22% in the log-spectrum domain. This is a relative reduction in WER of 17.04%.

	Subway	Car	Babble	Exhibition	Average
Clean	98.74%	98.70%	98.75%	99.01%	98.80%
20dB	97.24%	97.52%	98.54%	97.01%	97.58%
15dB	94.78%	96.43%	97.44%	95.25%	95.97%
10dB	89.41%	92.56%	93.77%	91.79%	91.88%
5dB	80.47%	80.44%	85.57%	82.66%	82.28%
0dB	60.15%	49.94%	59.02%	65.04%	58.54%
-5dB	28.83%	19.38%	20.91%	35.14%	26.06%
Average	84.41%	72.71%	86.87%	86.35%	85.25%

Table D.2: Accuracy for Set A. Cepstrum domain MMSE-Algonquin. The speech model contained 256 mixtures. Single component noise model estimated from the first 20 frames.

	Subway	Car	Babble	Exhibition	Average
Clean	98.93	99.12	98.99	99.32	99.09
20 dB	96.13	97.88	98.36	97.01	97.34
15 dB	92.54	95.65	97.08	94.57	94.96
10 dB	84.80	90.21	93.32	89.39	89.43
5 dB	68.74	75.15	84.16	80.07	77.03
0 dB	43.63	46.07	59.53	60.07	52.33
-5 dB	18.42	16.69	26.69	33.14	23.73
Average	77.17	80.99	86.49	84.22	82.22

Table D.3: Accuracy for Set A. Log-Spectrum domain MMSE-Algorithm after 2 iterations. The speech model contained 256 mixtures. A single component noise model was used which was estimated from the first 20 frames of each file.

	Subway	Car	Babble	Exhib.	Ave.
Clean	98.62	98.55	98.81	99.11	98.77
20 dB	93.77	94.98	96.72	94.35	94.96
15 dB	87.29	89.51	94.63	89.20	90.16
10 dB	76.27	78.20	87.21	79.51	80.30
5 dB	58.70	58.31	71.67	63.96	63.16
0 dB	34.33	30.29	41.37	39.49	36.37
-5 dB	16.18	13.72	16.85	20.80	16.89
Average	70.07	70.26	78.32	73.30	72.99

Table D.4: Accuracy for Set A. Using 4 mixture noise model. Log-Spectrum domain MMSE-Algorithm. The speech model contained 256 mixtures. A single component noise model was used which was estimated from the first 20 frames of each file.

	Subway	Car	Babble	Exhib.	Ave.
Clean	98.71	98.91	98.90	99.26	98.94
20 dB	95.73	97.40	98.12	96.30	96.89
15 dB	91.46	94.74	96.45	93.09	93.94
10 dB	82.62	88.85	91.98	85.78	87.31
5 dB	65.55	71.77	80.41	72.66	72.60
0 dB	39.94	39.12	51.69	50.05	45.20
-5 dB	17.19	15.54	20.52	25.21	19.62
Average	75.06	78.38	83.73	79.58	79.19

Table D.5: Accuracy for Set A. Using 8 mixture noise model. Log-Spectrum domain MMSE-Algorithm.

	Subway	Car	Babble	Exhib.	Ave.
Clean	98.74	99.00	98.90	99.29	98.98
20 dB	96.22	98.04	98.48	96.76	97.38
15 dB	92.42	95.62	96.93	94.05	94.76
10 dB	84.34	90.81	93.05	88.34	89.13
5 dB	68.10	74.43	82.79	77.82	75.78
0 dB	41.11	42.38	55.59	56.16	48.81
-5 dB	16.76	16.23	22.76	29.22	21.24
Average	76.44	80.26	85.37	82.63	81.17

Table D.6: Accuracy for Set A. Using 16 mixture noise model. Log-Spectrum domain MMSE-Algorithm.

	Subway	Car	Babble	Exhib.	Ave.
Clean	98.89	99.06	99.11	99.29	99.09
20 dB	96.25	97.91	98.48	97.19	97.46
15 dB	92.32	96.34	97.14	94.11	94.98
10 dB	84.07	91.11	93.05	88.52	89.19
5 dB	67.39	75.18	83.27	78.43	76.07
0 dB	41.85	44.41	56.84	58.07	50.29
-5 dB	16.76	16.44	24.46	31.63	22.32
Average	76.38	80.99	85.76	83.26	81.60

Table D.7: Accuracy for Set A. Using 32 mixture noise model. Log-Spectrum domain MMSE-Algorithm.

	Subway	Car	Babble	Exhib.	Ave.
Clean	98.96	99.09	98.96	99.32	99.08
20 dB	96.28	97.85	98.48	97.04	97.41
15 dB	92.69	95.86	97.05	94.45	95.01
10 dB	84.46	90.51	92.96	89.26	89.30
5 dB	68.07	75.94	83.54	79.33	76.72
0 dB	43.32	45.28	58.87	59.67	51.78
-5 dB	17.90	16.26	25.98	32.15	23.07
Average	76.96	81.09	86.18	83.95	82.05

Table D.8: Accuracy for Set A. Using 64 mixture noise model. Log-Spectrum domain MMSE-Algorithm.

	Subway	Car	Babble	Exhib.	Ave.
Clean	98.96	99.15	99.02	99.26	99.10
20 dB	96.19	97.85	98.45	96.79	97.32
15 dB	92.69	95.89	96.90	94.66	95.03
10 dB	84.89	89.96	93.05	88.71	89.15
5 dB	69.05	74.97	83.75	79.94	76.93
0 dB	43.60	46.16	58.87	60.10	52.18
-5 dB	18.39	16.99	26.42	32.49	23.57
Average	77.28	80.97	86.20	84.04	82.12

Table D.9: Accuracy for Set A. Using 128 mixture noise model. Log-Spectrum domain MMSE-Algorithm.

	Subway	Car	Babble	Exhib.	Ave.
Clean	98.93	99.09	99.02	99.32	99.09
20 dB	96.71	97.91	98.45	97.35	97.60
15 dB	93.98	95.65	97.52	95.50	95.66
10 dB	87.26	90.81	93.86	90.31	90.56
5 dB	74.12	77.60	86.04	81.58	79.83
0 dB	51.83	46.80	62.72	61.46	55.70
-5 dB	25.08	16.90	28.93	35.33	26.56
Average	80.78	81.75	87.72	85.24	83.87

Table D.10: Accuracy for Set A. 2 Gaussian mixture used to approximate single Gaussian. Log-Spectrum domain MMSE-Algorithm.

	Subway	Car	Babble	Exhib.	Ave.
Clean	98.93	99.09	99.02	99.32	99.09
20 dB	96.65	97.85	98.48	97.25	97.56
15 dB	94.29	95.62	97.46	95.62	95.75
10 dB	87.84	90.54	94.01	90.62	90.75
5 dB	75.53	76.96	86.52	81.27	80.07
0 dB	54.13	46.25	62.42	61.46	56.07
-5 dB	27.51	16.93	28.30	35.30	27.01
Average	81.69	81.44	87.78	85.24	84.04

Table D.11: Accuracy for Set A. 4 Gaussian mixture used to approximate single Gaussian. Log-Spectrum domain MMSE-Algorithm.

	Subway	Car	Babble	Exhib.	Ave.
Clean	96.50	96.22	96.93	96.95	96.65
20 dB	97.02	98.34	98.45	97.78	97.90
15 dB	94.50	97.01	97.79	95.68	96.25
10 dB	87.53	94.14	94.48	92.41	92.14
5 dB	75.90	84.04	86.76	84.97	82.92
0 dB	53.45	56.98	64.27	67.54	60.56
-5 dB	25.70	21.95	26.96	38.82	28.36
Average	81.68	86.10	88.35	87.68	85.95

Table D.12: Accuracy for Set A. Upper bound for 1 mixture noise model, noise estimated from true noise. Noise was found by subtracting the samples of each clean speech file from the samples of the corresponding noisy file. Log-Spectrum domain MMSE-Algorithm. The speech model contained 256 mixtures.

	Subway	Car	Babble	Exhibition	Average
Clean	4.33%	4.93%	5.04%	4.44%	4.69%
20dB	20.11%	9.82%	12.17%	16.08%	14.54%
15dB	29.75%	15.84%	16.43%	23.17%	21.30%
10dB	43.63%	27.06%	24.40%	30.82%	31.48%
5dB	56.65%	45.83%	33.31%	43.35%	44.78%
0dB	73.84%	68.29%	51.63%	57.98%	62.93%
-5dB	87.23%	88.21%	75.90%	78.06%	82.35%
Average	44.80%	42.51%	27.59%	34.28%	35.01%

Table D.13: WER for Set A. MMSE-Algonquin. Single component noise model estimated from the first 20 frames. Log-spectrum acoustic models were used in the recognizer, which accounts for the high WER.

	Subway	Car	Babble	Exhibition	Average
clean	5.10%	5.35%	5.34%	4.94%	5.18%
20dB	10.35%	6.44%	6.77%	8.49%	8.01%
15dB	16.24%	9.92%	8.41%	12.28%	11.71%
10dB	28.06%	16.72%	13.84%	18.33%	19.24%
5dB	46.02%	33.37%	22.70%	31.78%	33.47%
0dB	68.10%	59.16%	37.88%	47.08%	53.05%
-5dB	84.53%	86.09%	63.82%	69.15%	75.90%
Average	33.75%	35.28%	17.92%	23.59%	25.10%

Table D.14: WER for Set A, SOFT-Algonquin. Single component noise model estimated from the first 20 frames. Log-spectrum acoustic models were used in the recognizer, which accounts for the high WER.

Bibliography

[1] A. Acero, *Acoustical and Environmental Robustness in Automatic Speech Reconition*, Klewer Academic Publishers, 1992.

[2] A. Acero, S. Altschuler, and L. Wu, *Speech/Noise Separation Using Two Microphones and a VQ Model of Speech Signals*, Proc. Int. Conf. on Spoken Language Processing (2000).

[3] A. Acero, L. Deng, T. Kristjansson, and J. Zhang, *HMM Adaptation Using Vector Taylor Series for Noisy Speech Recognition*, Procedings of ICSLP (2000).

[4] M. Afify, H. Jiang, , F. Korkmazskiy, C.-H. Lee, Q. Li, O. Siohan, F.K. Soong, and A. Surendran, *Evaluating the Aurora Connected Digit Recognition Task - A Bell Labs Approach*, Eurospeech (2001).

[5] M. Afify and O. Siohan, *Sequential Noise Estimation with Optimal Forgetting for Robust Speech Recognition*, IEEE International Conference on Speech and Audio Processing (2001).

[6] H. Attias, J. Platt, A. Acero, and L. Deng, *Speech Denoising and Dereveration using Probabilistic Models*, Advances in Neural Information Processing Systems (2000).

[7] S. Boll, *Suppression of Acoustic Noise in Speech Using Spectral Subtraction*, IEEE Trans. Acoustics, Speech, and Signal Processing **27** (1979), 114–120.

[8] S. Boll, J. Porter, and L. Bahler, *Robust Syntax Free Speech Recognition*, IEEE Trans. Acoustics, Speech, and Signal Processing (1988), 179–182.

[9] D. M. Chickering and D. Heckerman, *Efficient Approximation for the Marginal Likelihood of Baysian Networks with Hidden Variables*, Tech. report, Microsoft Research, One Microsoft Way, April 1997.

[10] Cover and Thomas, *Elements of Information Theory*, 2 ed., Wiley-Interscience, 1991.

[11] J.R. Deller, J.H.L. Hansen, and L.G. Proakis, *Discrete-Time Processing of Speech Signals*, IEEE, 2000.

[12] L. Deng, A. Acero, J. Droppo L. Jiang, and X. Huang, *High-performance robust speech recognition using stereo training data*, Proceedings of the IEEE International Conference on Acoustics, Speech, and Signal Processing (2001).

[13] L. Deng, A. Acero, M. Plumpe, and X. Huang, *Large-Vocabulary Speech Recognition under Adverse Acoustic Environments*, Proceedings of the International Conference on Spoken Language Processing **Vol. 3** (2000), 806–809.

[14] J. Droppo and A. Acero, *Maximum A Posteriori Pitch Tracking*, ICSLP (1998).

[15] R. Duda, P. Hart, and D. Stork, *Pattern Classification*, 2 ed., Wiley-Interscience, New York., 2001.

[16] Y. Ephraim, *A Minimum Mean Square Error Approach for Speech Enhancement*, Proceedings of the IEEE International Conference on Acoustics, Speech, and Signal Processing (1990), pp. 829–832.

[17] ______, *Statistical-Model-Based Speech Enhancement Systems*, Proceedings of the IEEE **80** (1992), no. 2, 1526 – 1554.

[18] Y. Ephraim, D. Halah, and B.H. Juang, *On the Application of Hidden Markov Models for Enhancing Noisy Speech*, IEEE Transaction on Acoustics, Speech and Signal Processing **37** (1998), no. 12, 1846–1856.

[19] Y. Ephraim and M. Rahim, *On Second-Order Statistics and Linear Estimation of Cepstral Coefficients*, IEEE Transactions on Speech and Audio Processing **7** (1999), no. 2, 162–176.

[20] A. Erell and M. Weintraub, *Estimation Using Log-Spectral-Distance Criterion For Noise-Robust Speech Recognition*, Proceedings IEEE International Conference on Acoustics, Speech, and Signal Processing (1990), pp. 853–856.

[21] A. Erell and Mi. Weintraub, *Estimation of Noise-Corrupted Speech DFT-Spectrum Using the Pitch Period*, IEEE Transaction on Speech and Audio Processing **2** (1994), no. 1, pp. 1–8.

[22] B.J. Frey, *Graphical Models for Machine Learning and Digital Communication*, MIT Press, 1998.

[23] B.J. Frey, L. Deng, A. Acero, and T. Kristjansson, *ALGONQUIN: Iterating Laplace's Method to Remove Multiple Types of Noise and Channel Distortion from Log-Spectra Used in Robust Speech Recognition*, Eurospeech (2001).

[24] B.J. Frey, T. Kristjansson, L. Deng, and A. Acero, *Learning Dynamic Noise Models from Noisy Speech for Robust Speech Recognition*, NIPS (2001).

[25] S.F. Furui, *Cepstral analysis Technique for Automatic speaker Verification*, IEEE Trans. Acoustics, speech and Signal Processing **ASSP-29** (1981), 254–272.

[26] M.J.F. Gales, *Model-Based Techniques for Noise Robust Speech Recognition*, Ph.D. thesis, University of Cambridge, September 1995.

[27] M.J.F. Gales and S.J. Young, *An Improved Approach to the Hidden Markov Model Decomposition of Speech and Noise*, In Proc. of ICASSP (1992), 233– 236.

[28] ______, *Parallel Model Combination for Speech Recognition in Noise*, Tech. report, Cambridge University, Trumpington Street, June 1993.

[29] ______, *PMC for Speech Recognition in Additive and Convolutional Noise*, Tech. report, Cambridge University, Trumpington Street, December 1993.

[30] ______, *A fast and flexible implementation of parallel model combination*, In Proc. of ICASSP (1995), 133–136.

[31] ______, *Variance Compensation within the MLLR Framework*, Tech. report, Cambridge University, Trumpington Street, December 1996.

[32] H. Glotin, D. Vergyri, C. Neti, G. Potamianos, and J. Luettin, *Weighting Schemes for Audio-visual Fusion in Speech Recognition*, ICASSP (2001).

[33] Y. Gong, *Speech Recognition in Noisy Environments: A Survey*, Speech Communication **16** (1995), 261–291.

[34] A. K. Halberstadt, *Heterogeneous Acoustic Measurements and Multiple Classifiers for Speech Recognition*, Ph.D. thesis, Massachusetts Institute of Technology, November 1998.

[35] H. Hermansky, *Perceptual Linear Predictive (PLP) analysis of speech*, Journal of the Acoustical Society of America (1990), 1738–1752.

[36] H. Hermansky, N. Morgan, A. Bayya, and P. Kohn, *RASTA-PLP Speech Analysis*, Tech. report, US West, December 1991.

[37] H. Hirsch and D. Pearce, *The Aurora Experimental Framework for the Performance Evaluation of Speech Recognition Systems Under Noisy Conditions*, Proc. of ISCA ITRW Workshop on Automatic Speech Recognition (2000).

[38] X.D. Huang, A. Acero, and H. Hon, *Spoken Language Processing*, Prentice Hall, 2001.

[39] M. Ikram and D.R. Morgan, *A Multiresolution Approach to Blind Separation of Speech Signals in a Reverberant Environment*, ICASSP (2001).

[40] G.J Jang, T.W. Lee, and Y.H Oh, *Learning Statistically Efficient Features for Speaker Recognition*, In Proc. of ICASSP (2001).

[41] F. Jensen, *Introduction to Bayesian networks*, Springer, 1996.

[42] H. Jiang, F. Soong, and C.-H. Lee, *Heirarchical Stochastic Feature Matching for Robust Speech Recognition*, Proceedings of the IEEE International Conference on Acoustics, Speech, and Signal Processing (2001).

[43] M. I. Jordan, Z. Grahamani, T.S. Jaakkola, and L.K. Saul, *An Introduction to Variational Methods for Graphical Models*, Learning in Graphical Models (M. I. Jordan, ed.), MIT Press, 1999.

[44] M.I. Jordan, *Learning in Graphical Models*, MIT Press, 1999.

[45] B.H. Juang, *Speech Recognition in Adverse Environments*, Computer Speech and Language (1991), 275 – 294.

[46] J.C. Junqua and J.P. Haton, *Robustness in Automatic Speech Recognition*, Klewer Academic Publishers, 1996.

[47] N.S. Kim, *Nonstationary Environment Compensation Based on Sequential Estimation*, IEEE Signal Processing Letters **5** (1998), no. 3, pp. 57–59.

[48] ______, *Statistical Linear Approximation for Environment Compensation*, IEEE Signal Processing Letters **5** (1998), no. 1, pp. 8–10.

[49] D.H. Klatt, *A Digital Filterbank for Spectral Matching*, In Proc. of ICASSP (1989), 573 – 576.

[50] T. Kristjansson, L. Deng, A. Acero, and B.J. Frey, *Towards Non-stationary Noise Adaptation for Large Vocabulary Speech Recognition*, In Proc. of ICASSP (2001).

[51] T. Kristjansson and B.J. Frey, *Accounting for Uncertainty in Observations: A New Paradigm for Robust Automatic Speech Recognition*, In Proc. of ICASSP (2002).

[52] T. Kristjansson, B.J. Frey, L. Deng, and A. Acero, *Joint Estimation of Noise and Channel Distortion in a Generalized EM Framework*, In Proc. of Automatic Speech Recognition and Understanding Workshop (2001).

[53] R. Kuhn, F. perronnin, P. Nguyen, J.C. Junqua, and L. Rigazio, *Very Fast Adaptation with a Compact Context-Dependent Eigenvoice Model*, ICASSP (2001).

[54] S. Lauritzen and D. Spiegelhalter, *Local Computations with Probabilities on Graphical Structures and their Application to Expert Systems (with Discussion)*, Journal of the Royal Statistical Society (1988), 157–224.

[55] K.F. Lee, *Automatic Speech Recognition*, Klewer Acatemic Publishers, 1989.

[56] T.W. Lee and G.J. Jang, *The Statistical Structures of Male and Female Speech Signals*, In Proc. of ICASSP (2001).

[57] C.J. Leggetter and P.C. Woodland, *Flexible Speaker Adaptation using Maximum Likelihood Linear Regression*, Eurospeech'95 (1995), pp. 1155–1158.

[58] P. Lockwood and J. Boudy, *Experiments with a Nonlinear Spectral Subractor (NSS)*, Speech Communication **11** (1992), 215–228.

[59] D. Macho and Y.M. Cheng, *SNR-Dependent Waveform Proessing for Improving the Robustness of ASR Front-End*, ICASSP (2001).

[60] D.J.C. MacKay, *Choice of Basis for Laplace Approximation*, 1996.

[61] D. mansour and B.H. Juang, *A Family of Distortion Measures Based Upon Projection Operation for Robust Speech Recognition*, IEEE Transactions on Acoustics, Speech and Signal Processing **37** (1989), no. 11, pp. 1659–1679.

[62] P.J. Moreno, *Speech Recognition in Noisy Environments*, Ph.D. thesis, Carnegie Mellon University, Pittsburgh, Pennsylvania 15213, April 1996.

[63] P.J. Moreno, B. Raj, and R.M. Stern, *A Vector Taylor Series Approach for Environment-Independent Speech Recognition*, Proceedings of ICASSP (1994), 733–736.

[64] R. Neal, *Probabilistic Inference using Markov Chain Monte Carlo Methods*, Tech. report, University of Toronto, 1993.

[65] R. M. Neal and G. E. Hinton, *A View of the EM Algorithm that Justifies Incremental, Sparse, and other Variants*, Learning in Graphical Models (M. I. Jordan, ed.), MIT Press, 1999, pp. 355–368.

[66] T. Paek and E. Horowitz, *Uncertainty, Utility, and Misunderstanding: A Decision-Theoretic Perspective on Grounding in Conversational Systems*, In AAAI Fall Symposium on Psychological Models of Communication in Collaborative Systems (1999).

[67] C. Pal, T. Kristjansson, and B.J. Frey, *Noise Robust Speech Recognition Using Gaussian Basis Functions For Non-Linear Likelihood Function Approximation*, In Proc. of ICASSP (2002).

[68] L.R. Rabiner and B.H. Juang, *Fundamentals of Speech Recognition*, Prentice-Hall, 1993.

[69] B. Raj, E.B. Gouvea, P.J. Moreno, and R.M. Stern, *Cepstral Compensation by Polynomial Approximation for Environment-Independent Speech Recognition*, In Proc. International Conference on Spoken Language Processing (1996), 2340–2343.

[70] S. Roweis, *Gaussian identities*.

[71] ______, *One Microphone Source Separation*, Neural Information Processing Systems 13 (2000).

[72] S. Young and D. Kershaw and J. Odell and D. Ollason and V. Valtchev and P. Woodland, *The HTK book*, 1999.

[73] G. Saon, M. Padmanabhan, R. Gopinath, and S. Chen, *Maximum Likelihood Discriminant Feature Spaces*, ICASSP (2000).

[74] R. Schluter and H. Ney, *Using Phase Spectrum information for Improved Speech Recognition Performance*, ICASSP (2001).

[75] L.J. Siegel, *A Procedure for Using Pattern Classification Techniques to Obtain a Voiced/Unvoiced Classifier*, IEEE Transactions on Acoustics, Speech, and Signal Processing **27** (1979), no. 1, 83–88.

[76] M. Slaney, *A critique of pure audition*, Computational Auditory Scene Analysis (Dave Rosenthal and Hiroshi Okuno, eds.), Lawrence Erlbaum Associates, 1997.

[77] R. M. Stern, A. Acero, F.H. Liu, and Y. Ohshima, *Signal Processing For Robust Speech Recognition*, Automatic Speech Recognition (C.H. Lee and F. Soong, eds.), Klewer Academic Publishers, 1998, pp. 351–378.

[78] A.P. Varga and R.K. Moore, *Hidden Markov Model Decompostion of Speech and Noise*, In Proc. of ICASSP (1990), 845–848.

[79] M.Q. Wang and S.J. Young, *Speech Recognition using Hidden Markov Model Decompostion and a General Background Speech Model*, In Proc. of ICASSP **vol. 1** (1992), pp. 253–256.

[80] S.J. Young, N.H. Russel, and J.H.S. Thornton, *Token Passing: a Simple conceptual Model for Connected Speech Recognition Systems*, Tech. report, Cambridge University, Trumpington Street, December 1989.

[81] G. Zweig, *Speech Recognition with Dynamic Bayesian Networks*, Ph.D. thesis, University of California, Berkeley, 1998.

[82] G. Zweig and S. Russell, *Speech Recognition with Dynamic Baysian Networks*, Fifteenth National Conference on Artificial Intelligence,AAAI'98 (1998), 173–180.

[83] E. Zwicker, H. Fastl, and H. Frater, *Psychoacoustics: Facts and Models*, Springer, 1999.