AN INTEGRATED APPROACH TO FEATURE COMPENSATION COMBINING PARTICLE FILTERS AND HIDDEN MARKOV MODELS FOR ROBUST SPEECH RECOGNITION

A Dissertation
Presented to
The Academic Faculty

By

Aleem Mushtaq

In Partial Fulfillment
of the Requirements for the Degree
Doctor of Philosophy
in
Electrical and Computer Engineering

School of Electrical and Computer Engineering
Georgia Institute of Technology
August 2013

AN INTEGRATED APPROACH TO FEATURE COMPENSATION COMBINING PARTICLE FILTERS AND HIDDEN MARKOV MODELS FOR ROBUST SPEECH RECOGNITION

Approved by:

Dr. Mark A. Clements, Committee Chair
Professor, School of ECE
Georgia Institute of Technology

Dr. Fumin Zhang
Professor, School of ECE
Georgia Institute of Technology

Dr. Chin-Hui Lee, Advisor
Professor, School of ECE
Georgia Institute of Technology

Dr. David Goldsman
Professor, H. Milton Stewart School of ISyE
Georgia Institute of Technology

Dr. Ghassan Al-Regib
Professor, School of ECE
Georgia Institute of Technology

Date Approved: May 6, 2013

Dedicated to Yusaira and Filzah

ACKNOWLEDGMENTS

All praise and thank is for Almighty Allah, who is the All-Powerful and the Generous. I am extremely grateful to Dr. Chin-Hui Lee. He consented to be my PhD thesis advisor, and guided me while I was conducting research in the challenging area of speech recognition. He not only supervised me throughout my research work, but also helped me a lot in all aspects of my PhD degree.

I would also like to thank Dr. Mark A. Clements, Dr. Ghassan Al-Regib, Dr. Fumin Zhang and Dr. David Goldsman for serving on my PhD committee. Their suggestions and feedback helped me in writing and presenting my thesis. I owe a lot to my teachers at National University of Sciences and Technology, Pakistan, especially Dr. Bilal and Dr. Ejaz. Dr. Bilal was my MS thesis advisor, and his mentoring was crucial in my subsequent research work.

I am grateful to my colleagues, who helped me at various stages of my PhD. Yu Tsao helped me in the starting stages of my research. You-Chi, Ifan, Marco, and Salman Aslam, all contributed to my work in various ways. I would also like to thank Ha NGUYEN and his companions at Nanyang Technological University, Singapore, for extending the PFC algorithm to large vocabulary speech recognition systems.

Higher Education Commission of Pakistan (HEC) played a key role in facilitating my graduate studies. Ms. Saima Naurin, Project Director, (HRD), was very helpful and supported me whenever I requested for assistance. The support staff at Georgia Tech was also very helpful. I would like to extend special thanks to Ms. Brenda Thompson (Office of the Bursar), who looked after my financial assistance programs and coordinated with HEC. Her considerateness allowed me to be unworried and to concentrate on my studies.

Finally, I am extremely indebted to all my family members for their support and constant prayers. My parents, Abboo and Ammi, supported me in every way possible during the course of my studies. The companionship of my wife, Yusaira, and the arrival of our

daughter, Filzah, during my PhD studies meant a lot to me. Both were a constant source of comfort for me in my grueling schedule.

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

CHAPTER 1

SCIENTIFIC GOALS

Speech communication is a fundamental part of the human activities. With the advent of a wide variety of gadgets, machines have become an integral part of the human society. Consequently, the ability to communicate with these machines i.e. Automatic Speech Recognition (ASR) has become a highly sought after technology. Modern Hidden Markov Model (HMM) based ASR systems work well when there is a good conformation between the training conditions where the models are built and the testing conditions where the models are used. However, if there is mismatch between the training and the testing conditions, the performance of the ASR systems degrade rapidly. On the other hand, humans perform well in the speech understanding task even in adverse conditions[1][2]. Bringing the performance of ASR systems at par with a human listener is a goal that has remained elusive for many decades.

Research in the psycho-acoustics of humans indicate that the superior human performance in speech perception is due to the ability to track the speech signal of interest in the presence of multiple interfering signals corrupting the speech signal. For example, humans can maintain an intelligibility of $70\% - 80\%$ under following conditions [1]:

1. The sentence is mixed with broadband random noise where
   $Power\,of\,noise = 4\,x\,power\,of\,speech\,(SNR - 6dB)$.

2. When the noise is at $90degrees$ angle to the speech source
   $Power\,of\,noise = 40\,x\,power\,of\,speech\,(SNR - 16dB)$.

Henceforth, many approaches have been proposed that attempt to estimate the clean speech features from the features of the distorted signal. Stochastic matching [3][4][5], vector Taylor series VTS [6] and codeword dependent cepstral normalization (CDCN) [7] are some of the well known instances of this approach. Although these methods estimate

the speech features from the distorted features, they do not attempt to track the clean speech features. Consequently, they are not adaptable to a changing environment within an utterance. A method that is capable of handling the variable noise will have to track the clean speech features in the conventional sense.

A sequential Monte Carlo based tracking algorithm for feature compensation algorithm was initially proposed [8][9] in which the noise was treated as a state variable while speech was considered as the signal corrupting the observation noise. In [8], VTS approximation was used to approximate the clean speech signal by applying a minimum mean square error (MMSE) procedure. In [9] extended Kalman filters were used to model a dynamical system representing the noise which was further improved by using Polyak averaging and feedback with a switching dynamical system [10]. These were initial attempts to incorporate particle filter for speech recognition in a more indirect fashion as it was used for tracking of noise instead of the speech signal itself. Since the speech signal is treated as the corrupting signal to the noise, there is limited information readily available from the HMMs or the recognition process that can be utilized efficiently in the compensation process.

In contrast to the previous particle filter studies [8][9][10], we develop a method where we treat the speech signal as the state variable and the noise as the corrupting signal, and attempt to estimate the clean speech from the noisy speech [11]. The purpose of the proposed research is to develop algorithms that are able to directly track the clean speech features. These algorithms, which we call particle filter compensation (PFC), are based on tracking the clean speech features with the particle filters with the aim of compensating the noisy features. It is assumed that the clean speech is corrupted according to an additive noise model. The statistical information available in the acoustic models of clean speech, obtained from the HMMs trained with clean speech, is incorporated in the tracking algorithm as an altrnative to the state transition model. Noise statistics are collected from the background environment and used in the observation model, which is derived from the additive noise model.

The similarity between HMMs and particles filters can be seen from the fact that an observation probability density function corresponding to each state of an HMM describes, in statistical terms, the characteristics of the source generating a signal of interest if the source is in that particular state, whereas in particle filters we try to estimate the probability distribution of the state the system is in when it generates the observed signal of interest. Particle filters are suited for feature compensation because the probability density of the state can be updated dynamically on a sample-by-sample basis. On the other hand, state densities of the HMMs are assumed independent of each other. Although they are good for speech inference problems, HMMs do not adapt well in fast changing environments. By establishing a close interaction of the particle filters and HMMs, the potentials of both models can be harnessed in a joint framework to perform feature compensation for robust speech recognition. We improve the recognition accuracy through compensation of noisy speech, and we enhance the compensation process by utilizing information in the HMM state transition and mixture component sequences obtained in the recognition process.

Instead of using specific knowledge at the model and state levels from HMMs which is hard to estimate, we pool model states into clusters as side information [12]. Since each cluster encompasses more statistics when compared to the original HMM states, there is a higher possibility that the newly formed probability density function at the cluster level can cover the underlying speech variation to generate appropriate particle filter samples for feature compensation. Additionally, a dynamic joint tracking framework to monitor the clean speech signal and noise simultaneously is also introduced to obtain good noise statistics [13][14]. In this approach, the information available from clean speech tracking can be effectively used for noise estimation. The availability of dynamic noise information can enhance the robustness of the algorithm in case of large fluctuations in noise parameters within an utterance.

The goal of this thesis is to develop algorithms that are robust to non-stationery additive noise distortions. To achieve our aim, we will pursue the following steps:

1. To develop a particle filter based framework for tracking the clean speech features from the noisy features observed in the testing environment.

2. To overcome the non-availability of a suitable state transition model for the speech signal by integrating the HMMs in the particle filter framework.

3. To effectively reorganize the statistical information available from the HMMs for optimal utilization in the PFC algorithm

4. To develop a particle filter algorithm that works in parallel with the PFC algorithm, for tracking the noise signal corrupting the clean speech signal and use the information obtained to improve the PFC algorithm.

5. To present a Markov chain Monte Carlo algorithm for joint estimation of speech features and noise features to enhance the compensation of the noisy speech features

6. To investigate the performance of the PFC algorithm is ASR applications

   (a) Apply PFC to robust speech recognition tasks

   (b) Extend PFC to LVCSR tasks

The particle filter algorithms have gained popularity as a tracking algorithm is the past decade due to its versatility and the freedom from constraints that effect other tracking algorithms such as Kalman filter. The algorithm is a sequential importance sampling (Monte Carlo) method, which has two important components. First, it's effectiveness is dependent on the accurate placement of samples. In this thesis, we will investigate this aspect in detail and see how the placement of the samples impact the performance of the PFC algorithm. Second, the weights assigned to the samples also play an important role. These weights are computed using the observation model. In the PFC algorithm, the observation model is derived from the additive noise model, which in turn depends on the noise statistics retrieved from the background environment. Consequently, the methods to obtain the noise statistics

and update them during the utterance will also be discussed in detail. The performance of the PFC algorithm will be evaluated on the Aurora-2 connected and the Aurora-4 large vocabulary continuous speech recognition task.

The dissertation will be organized as follows. Chapter 2 provides an overview of the theory used in the development of the PFC framework. Tracking algorithms and Monte Carlo methods in general and particle filters method in particular are introduced. The chapter also discusses the relationship between the HMMs and the particle filters and the two can be integrated together. Chapter 3 details the development of the PFC algorithm and the utilization of statistics available from the HMMs for plugging in the PFC algorithm. Chapter 4 presents two methods for dynamic estimation of noise statistics used in the PFC algorithm. One is a parallel implementation of two particle filters that run simultaneously while the second is based on the MCMC approach. In chapter 5, we extend the PFC algorithm to large vocabulary continuous speech recognition tasks before presenting the concluding remarks in chapter 6.

CHAPTER 2

BACKGROUND OVERVIEW

In this chapter, the human speech perception, which provides the motivation behind PFC is briefly described. Tracking methods and Monte-Carlo (MC) methods are the building blocks of the PFC and other algorithms in this thesis and are therefore described in detail. The Monte Carlo overview leads to the development of particle filters, which are flexible tracking techniques based on a MC approach. Finally, HMMs and introduced and the relationship and similarities between HMMs and particle filters are explored. This relationship is an important topic of the chapter because it leads to the integration of HMMs and particle filters and subsequently the formulation of the PFC algorithm.

2.1 Background of Automatic Speech Recognition

The speech production mechanism goes through various stages. A thought is generated in the speaker's mind, which is then put into a sequence of words. These words are converted into a speech signal using various muscles including face muscles, chest muscles, tongue, etc. This signal is distorted by environmental factors such as background noise, reverberations, channel distortions when sent through a microphone, telephone channel etc. The aim of Automatic Speech Recognition Systems (ASR) is to reconstruct the spoken words from the speech signal [15]. From an information theoretic perspective, we can treat what is between the speaker and the machine as a distortion channel as shown in Figure 1.

Here, W represent the spoken words, and X is the speech signal. The problem of extracting W from X can be viewed as finding the words sequence that most likely resulted in the observed signal X, as given in Equation (1).

$$\begin{aligned} \hat{W} &= \arg\max_{W} p(W, X) \\ &= \arg\max_{W} p_{\hat{\Lambda}}(X|W). \end{aligned} \tag{1}$$

Like any other Machine Learning/Pattern Recognition problem, the posterior $p(X|W)$ plays

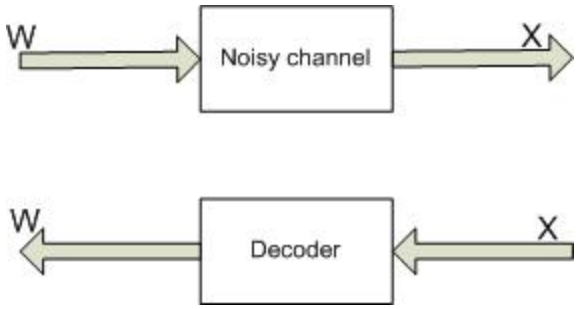

Figure 1: Information theoretic view of speech recognition

a fundamental role in the decoding process. This distribution is parametric and its parameters $\hat{\Lambda}$ are found from the available training data. Modern ASR systems do well when the environment of speech signal being tested matches well with that of the training data. This is because the parameter values correspond well to the speech signal being decoded. However, if the environments of training and testing data do not match well, the performance of the ASR systems degrade. Many schemes have been proposed to overcome this problem, but humans still outperform them, especially in adverse conditions. The distortions in a speech signal can be viewed in the signal space, the feature space and the model space [3] as shown in Figure 2. Resilience to environmental distortions can be added in the feature extraction process by modifying the distorted features, or adapting the acoustic models to better match the environment from which test signal has emanated. S_X and F_X represent the speech signal and the speech features respectively whereas M_X represents the acoustic models. $T_S(.)$, $T_F(.)$ and $T_M(.)$ represent the distortions in signal, feature and model space respectively.

2.1.1 Robustness in Model Space

The approaches to overcome the environment mismatch problem falls under two categories. One way is to adapt the parameters of acoustic models, $\hat{\Lambda}$ such that they match better with the testing environment, and are implemented at stage 3 of the scheme shown in Figure 2. Most commonly used structure for the acoustic models in ASR systems is the Hidden

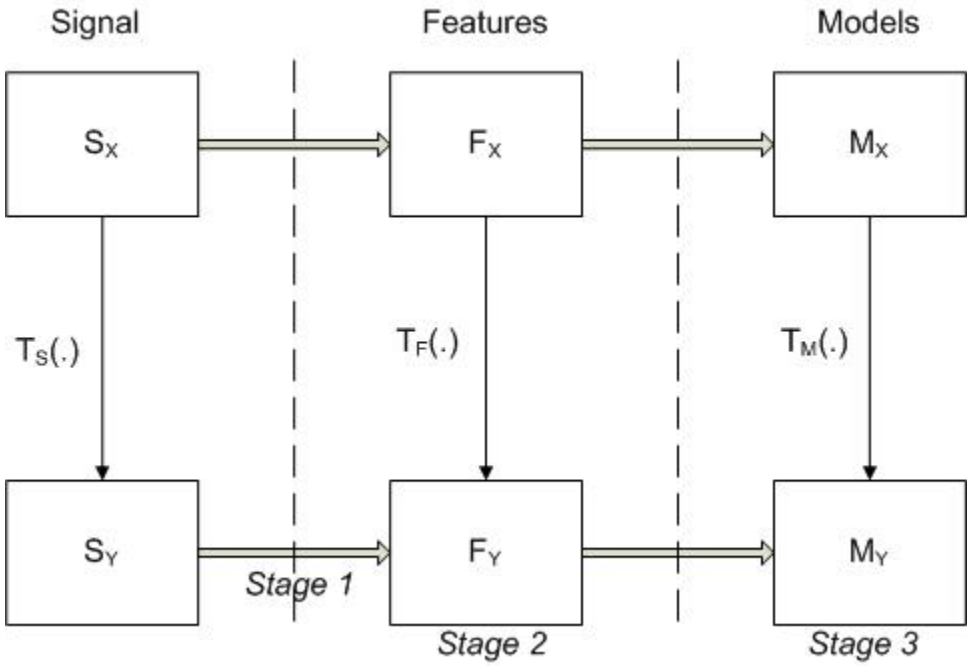

Figure 2: Stages of training and test speech data

Markov Models (HMM)[16]. We start with an initial set of HMMs and constantly update them as new information is obtained from the environment. A maximum a posteriori (MAP) estimation directly maps the acoustic models to a new set of models that are better suited for the new environment [17]. If sufficient testing data is available, the performance approaches that which is achieved by maximum likelihood training. However, it may not be very effective if training data size is limited. Structural MAP (SMAP) [18] exploits the hierarchical structure of the acoustic models and performs better when limited data is available. Joint MAP [19] also does well with small limited adaptation data. Maximum likelihood linear regression (MLLR) [20] is an indirect approach to model adaptation that computes the affine transformation from test data, which is then applied to the initial set of models to obtain the new models that match better with the testing environment. MAP and MLLR have been successfully combined [15] [21] where the affine transformation is obtained using MAP instead of maximum likelihood. For both MAP and MLLR techniques, good transcripts from the testing environment are required and may be difficult to obtain in

some scenarios.

Methods such as stochastic matching [3][4][5], parallel model combination (PMC) [22][23]and vector Taylor series VTS [6] model the environmental distortions using some nuisance parameters and capture them from the test data. Stochastic matching estimates the nuisance parameters using maximum likelihood technique and applies the correction to model parameters. VTS approximates the new set of models using vector Taylor series of the distortion model and obtains the noise parameters using expectation maximization (EM) [24] algorithm. In PMC algorithm, in addition to the clean speech HMMs (initial model set), another set of HMMs is obtained for the noise signal. This secondary model set represents the noise component of the noisy signal and is obtained from the samples before the onset of speech. It is then combined with the clean speech HMMs to get the hybrid HMMs that will be used for testing in the new environment. The advantage of Stochastic matching, PMC and VTS over MAP and MLLR techniques is that they do not require transcripts for adaptation of models. However, some type of assumption has to be made regarding the distortion model.

The above five techniques aim to improve the model parameters, for example if CDHMMs [25] are used, this implies adjusting the means and covariance matrices of the Gaussian mixtures to best fit the data from the testing environment. Another approach for improving acoustic models is to optimize them with the aim of minimizing the recognition error of ASR system instead of finding the best possible distribution for test data. Minimum classification error (MCE) [26] and soft margin estimation (SME) [27] strategies fall under this category. These are offline techniques that are known to improve the ML training of HMMs. Similarly, maximum mutual information (MMI) training [28][29] maximizes the mutual information between the observations and the corresponding HMM from an Information Theoretic perspective.

2.1.2 Robustness in Feature Space

The techniques that operate directly on features such as MFCCs are similar to approaches described for model space primarily because the distortion models are the same. Stochastic matching [3] and VTS [30] for example work well in the feature space. Feature-space maximum likelihood (fMLLR) [31] and feature-space eigen-MLLR [32] compensate noisy features to reduce mismatch between training and testing data. Cepstral mean subtraction (CMS) [33]is a simple yet effective technique where long-term mean value of the feature vectors is subtracted from each vector. It normalizes the mean of the features and reduces the variability of the data. Codeword dependent cepstral normalization (CDCN) [7] finds a correction vector from the stereo data, which is then used to correct the noisy observation and obtain the compensated feature vector. Higher order normalizations such as mean and variance normalization (MVN) [34] as well as histogram normalization [35] have also been proposed. These strategies occupy the stage 2 of Figure 2.

Robustness to noise can also be improved by a good choice of features in stage 1 of Figure 2. Mel Frequency Cepstral Coefficients (MFCC) have been widely used and are well known in the speech recognition community [36]. These are based on perceptual cues as the short term Fourier transform (STFT) of the speech is mapped to the perceptual scale (mel scale). Subsequently, the DCT transform of the log of power in each window, which is positioned precisely according to the mel scale, gives the final features. It is common to use the first and second time derivatives of MFCCs along with the features themselves in ASR systems. MFCCs allow both suppression of insignificant spectral variations in higher frequency bands and preserve acoustic information. However, these features do not discriminate well between consonants and may not be very robust in presence of additive noise [37]. Perceptual linear predictive (PLP) features are based on linear predictive (LP) analysis [38]. The power spectrum of speech signal is warped along its frequency axis into Bark frequency, which is based on the shape of auditory filters. The spectrum is then passed through an equal loudness pre-emphasis and intensity loudness converter that

reduces the spectral amplitude variation in the critical band and the resultant spectrum is approximated by an all pole model. PLP features are inferior to MFCCs in terms of capturing the acoustic information [39]. Principal component analysis (PCA) and linear discriminate analysis (LDA) [40] have also been widely used for feature extraction because of their good classification capability. These feature extraction techniques do not address the additive noise problem specifically and rely more on their capability to capture acoustic information, conciseness of feature size and their classification ability.

Recently proposed ETSI [41] is a technique that addresses additive noise more directly by removing it in a two stage Wiener filter. First stage removes the noise while the second stage removes the residual noise introduced in the first stage. Additionally, gain factorization is done to apply more aggressive noise reduction in purely noisy frames compared to ones having speech content. Finally cepstrum coefficient is obtained from the resultant signal. ETSI advanced front end achieves good performance in terms of noise robustness [42], but is computationally expansive as the noise removal is done on the waveform [42]. Spectral subtraction (SS) [43][44][45] was one of the first techniques that estimated clean speech spectrum from the noisy spectrum. SS approach specifically addresses the additive noise distortion and estimates the noise spectrum from frames available before the onset of speech. It's an effective technique and inspired many noise robustness algorithms. However, it suffers from the drawback that it introduces musical noise due to non-linear subtraction procedures. A summary of various robustness techniques is shown in Figure 3.

2.1.3 The challenge of non-stationary noise

SM and VTS are some of the best available techniques that compensate the distorted speech features under additive noise conditions for robust ASR. The noise is treated as a nuisance parameter, and is estimated from the noisy features. The information thus obtained is then used to estimate a new set of clean features. Both techniques are ,however, constricted to assume that the noise is stationery for the purpose of obtaning the noise parameter. To

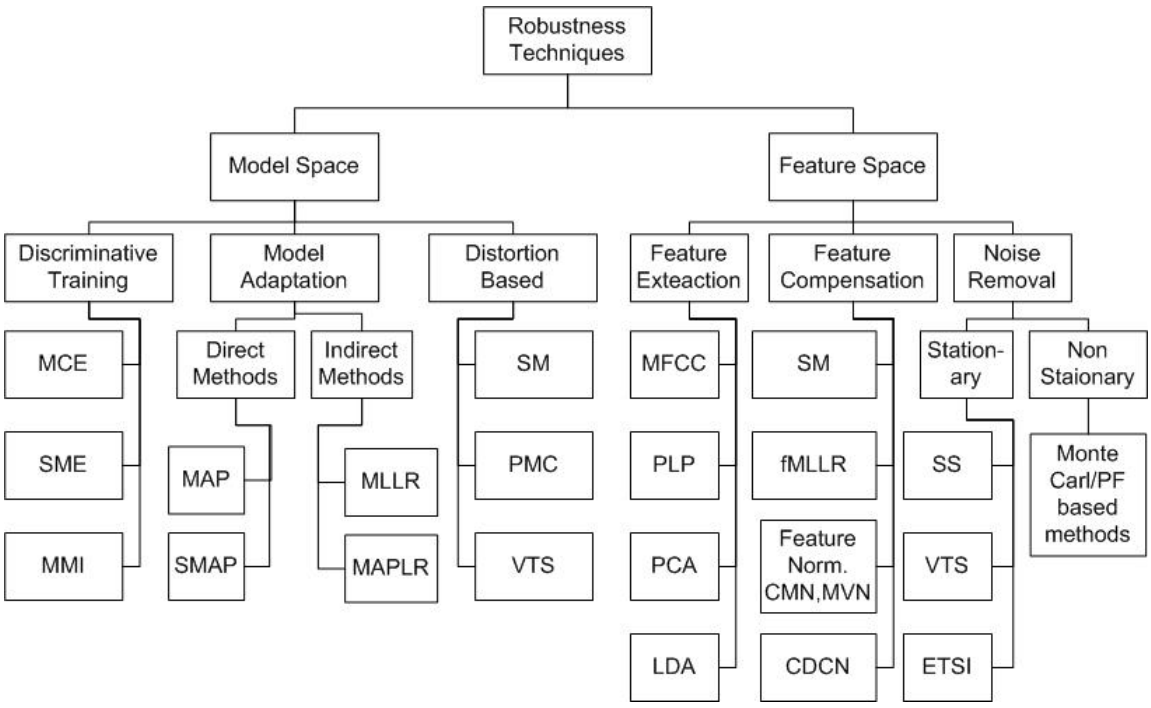

Figure 3: Summary of robustness techniques

address the contamination of varying noise within an utterance, noise tracking technniques have been proposed. These techniques track the noise using PF algorithms. However, due to a lack of structure in the noises typically encountered in ASR problems, noise tracking strategy also had a limited potential. For these reasons, the current state of the art technologies suffer from the dearth of successful strategies to address the issues of non-stationary noise within an utterance.

2.2 Human Speech Perception

The robustness of the human speech communication can be observed in everyday life. As described in the psycho-acoustic literature, following are some of the adverse scenarios in which humans of a normal hearing capability can easily maintain an intelligibility of $70\% - 80\%$[1]:

1. In additive noise conditions, the noise power can be four times the speech power, i.e., the speech to noise ratio (SNR) of $-6dB$. If the noise source and the speech are

separated by 90° and the speech source is straight ahead, then the SNR can be as low as $-16dB$ (noise power can be 40 times that of speech)

2. Both the source and the listener are in highly reverberations conditions with reverberation times exceeding 4 seconds[46].

3. The spectral lope of the communication channel can be varied between $\pm 5dB/octave$ at rates up to $2Hz$ if the noise power is equal to the speech power[47].

4. Background speech from multiple speakers is contaminating the speech of interest and the powers of both speech signals is the same[48].

The robustness in human speech intelligibility is achieved in many ways. Amongst them,the ability to track the speech signal of interest and isolate it from the interfering signals is illuminating for the ASR technology. Listeners exploit the pressure difference of the speech signal on the two ears, known as the interaural level difference (ILD), and the time of arrival difference, known as the interaural time difference (ITD) to track the speech signal[1]. In machines, this behavior can be mimicked by using microphone arrays to capture the speech signal of interest. In addition to using ILD and ITD, human listeners have the ability to give selective attention to the signal of interest [49][50][51]. Among other cues used to focus on the speech signal of interest, the fundamental frequency of the speech signal plays a critical role. For example, Sheffers [52] showed that humans can differentiate between two vowels based on the difference between their fundamental frequency.

Another experiment by Moore [53] sheds light on the human speech perception in presence of environmental noise. The experiment highlights a human speech perception feature, which is reminiscent of modern tracking algorithms that track objects in presence of occlusions. He showed that, under noisy conditions, humans are able to listen only to some glimpses of the speech signal and use these glimpses to identify the speech. The phenomena is shown in Figure 4. From left to right, the three panels show the utterance masked

by a single talker, eight talkers bable, and speech shaped noise, all at $-6dB$.The images depict the spectrotemporal excitation pattern (STEP), is a smoothed and compressed representation of the envelope of the basilar membrane response to sound. The row represents the STEP for the actual speech token, the middle row represents distorted STEP and the bottom row shows the glimpses observed by the listener. Depending on the type masking, glimpses of the speech seen by the listener varies. But it was shown in [54] that the word identification scores were almost at the level of uninterrupted speech despite a loss of 50% of the waveform due to masking.

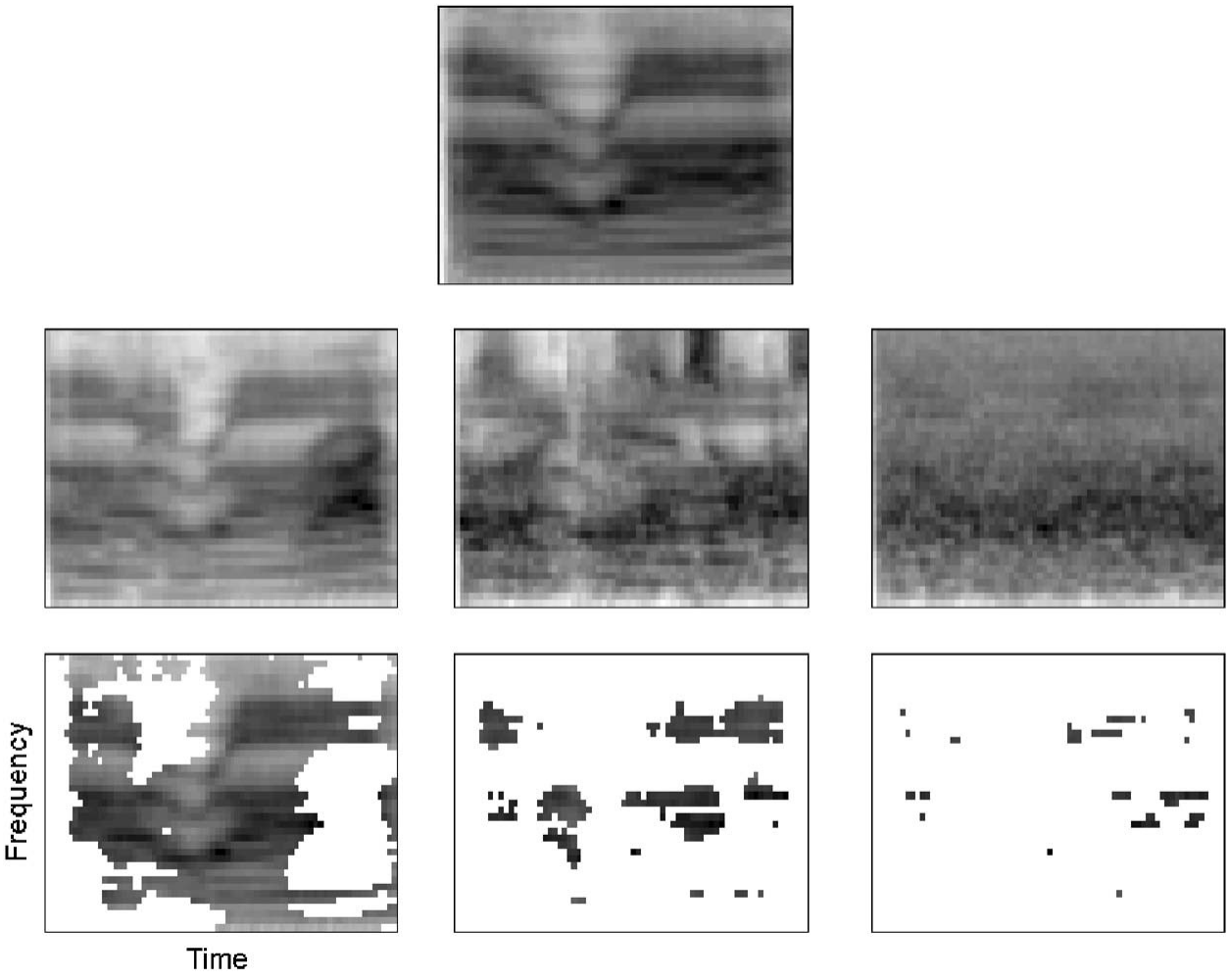

Figure 4: An illustration of potential glimpses of a short speech token

Although the understanding that how human speech perception works is not concrete, yet from the above discussion, it is evident that some form of tracking of the frequencies of interest in the speech signal is taking place. Taking a cue from human speech perception, a tracking approach for ASR systems has the potential to enhance the robustness of these systems in adverse environments. To consider the feasibility of such an approach, we first take a look at fundamentals of the modern tracking algorithms.

2.3 Tracking Algorithms

Tracking is the problem of estimating the trajectory of an object in a space as it moves through that space. The space could be an image plane captured directly from a camera or it could be synthetically generated from a radar sweep. Generally, tracking schemes can be applied to any system that can be represented by a time dynamical system which consists of a state space model and an observation:

$$\begin{aligned} x_t &= f(x_{t-1}, w_t) \\ y_t &= h(x_t, n_t) \end{aligned} \tag{2}$$

where n_t is the observation noise and w_t is called the process noise and represents the model uncertainties in the state transition function $f(.)$. What is available is an observation y_t which is function of x_t.We are interested in finding a good estimate of current state given observations till current time t i.e. $p(x_t|y_t, y_{t-1}, y_{t-2}, ..., y_0)$. The state space model $f(.)$ represents the relation between states adjacent in time. The model in Equation (2) assumes that the state sequence is a one step Markov process:

$$f(x_{t+1}|x_t, x_{t-1}, x_{t-2}, ..., x_0) = f(x_{t+1}|x_t) \tag{3}$$

It is further assumed that observations are independent of one another:

$$f(y_{t+1}|x_{t+1}, y_t, y_{t-1}, y_{t-2}, ..., y_0) = f(y_{t+1}|x_{t+1}) \tag{4}$$

Tracking is a two step process. The first step is to obtain density x_t at time $t-1$. This is called the prior density of x_t. Once it is available, we can construct a posterior density

upon availability of observation y_t. The propagation step is given in Equation (5) and the update step is obtained using Bayesian theory (Equation (6)).

$$f(x_t|y_{t-1}, y_{t-2}, ..., y_0) = \int f(x_t|x_{t-1}) f(x_{t-1}|y_{t-1}, y_{t-2}, ..., y_0) dx_{t-1} \tag{5}$$

$$f(x_t|y_t, y_{t-1}, y_{t-2}, ..., y_0) = \frac{f(y_t|x_t, y_{t-1}, y_{t-2}, ..., y_0) f(x_t|y_{t-1}, y_{t-2}, ..., y_0)}{f(y_t|y_{t-1}, y_{t-2}, ..., y_0)} \tag{6}$$

2.3.1 Kalman Filter as a Recursive Estimation Algorithm

Kalman filter is the optimal recursive estimation solution for posterior density $p(x_{t+1}|y_t, ..., y_0)$ if the time dynamical system is linear:

$$\begin{aligned} x_{t+1} &= A_t x_t + w_t \\ y_t &= C_t x_t + n_t \end{aligned} \tag{7}$$

where A_t and C_t are known as state transition matrix and observation matrix respectively. Subscript t indicates that both can vary with time. Under the assumption that both process noise w_t and observation noise n_t are Gaussian with zero mean and covariance Q_t and R_t respectively, $p(x_{t+1}|x_t)$ is Gaussian and can be readily obtained as follows:

$$\begin{aligned} mean(x_{t+1}|x_t) &= E(A_t x_t + w_t) \\ &= A_t x_t \\ covariance(x_{t+1}|x_t) &= E(w_t {w_t}^T) \\ &= Q_t \end{aligned} \tag{8}$$

Therefore

$$p(x_{t+1}|x_t) \sim \mathcal{N}(A_t x_t, Q_t) \tag{9}$$

To obtain the propagation step, we need $p(x_t|y_t, ..., y_0)$ in addition to $p(x_{t+1}|x_t)$. Since this is an iterative step, the estimate of x_t, given observations up to time t, is available at $t + 1$. Let's call it $x_{t|t}$ and let the covariance of $x_{t|t}$ be $P_{t|t}$. Then

$$p(x_t|y_t, ..., y_0) \sim \mathcal{N}(\hat{x}_{t|t}, P_{t|t}) \tag{10}$$

where $P_{t|t}$ is the covariance of $x_t|y_t, ..., y_0$ and is given by $E[(x_t - E[x_t])(x_t - E[x_t])^T|y_t, ..., y_0]$. Now, both components of the integral in Equation (5) are available in (9) and (10). Solving the integral using expanding and completing the squares [55], we get

$$p(x_{t+1}|y_t, ..., y_0) \sim \mathcal{N}(A_t\hat{x}_{t|t}, A_tP_{t|t}A_t^T + Q_t) \tag{11}$$

This is the propagation step and is sometimes also written as

$$p(x_{t+1}|y_t, ..., y_0) \sim \mathcal{N}(\hat{x}_{t+1|t}, P_{t+1|t}) \tag{12}$$

To get the update step, we note that the distributions of $x_{t+1}|y_t, ..., y_0$ and y_{t+1} are both Gaussian and consequently, $x_{t+1}|y_{t+1}, y_t, ..., y_0$ is also Gaussian.

$$\begin{aligned} \hat{x}_{t+1}|x_{t+1} &= E[x_{t+1}|y_{t+1}, y_t, ..., y_0] \\ &= \hat{x}_{t+1}|x_t + R_{xy}R_{yy}^{-1}(y_{t+1} - E[y_{t+1}|y_t, ..., y_0]) \end{aligned} \tag{13}$$

where

$$\begin{aligned} R_{xy} &= E[(x_{t+1} - E[x_{t+1}])(y_{t+1} - E[y_{t+1}])^T|y_t, ..., y_0] \\ &= E[(x_{t+1} - \hat{x}_{t+1|t})(C_{t+1}(x_{t+1} - \hat{x}_{t+1|t}) + n_{t+1})^T|y_t, ..., y_0] \\ &= P_{t+1|t}C_{t+1}^T \end{aligned} \tag{14}$$

and

$$R_{yy} = C_{t+1}P_{t+1|t}C_{t+1}^T + R_{t+1} \tag{15}$$

Back substituting Equation (14) and Equation (15) in Equation(13), we get

$$\hat{x}_{t+1}|x_{t+1} = \hat{x}_{t+1|t} + K_{t+1}(y_{t+1} - C_{t+1}\hat{x}_{t+1|t}) \tag{16}$$

where K_{t+1} is called the Kalman gain and is given by

$$K_{t+1} = P_{t+1|t}C_{t+1}^T(C_{t+1}P_{t+1|t}C_{t+1}^T + R_{t+1})^{-1} \tag{17}$$

The covariance of $\hat{x}_{t+1|t+1}$ is given by

$$\begin{aligned} P_{t+1|t+1} &= P_{t+1|t} - P_{t+1|t}C_{t+1}^T(C_{t+1}P_{t+1|t}C_{t+1}^T + R_{t+1})^{-1}C_{t+1}P_{t+1|t} \\ &= (1 - K_{t+1}C_{t+1}^T)P_{t+1|t} \end{aligned} \tag{18}$$

The block diagram in Figure 5 below shows a general recursive estimation algorithm steps starting from some initial state estimate x_0. The block labeled Kalman filter summarizes the steps specific to Kalman filter algorithm.

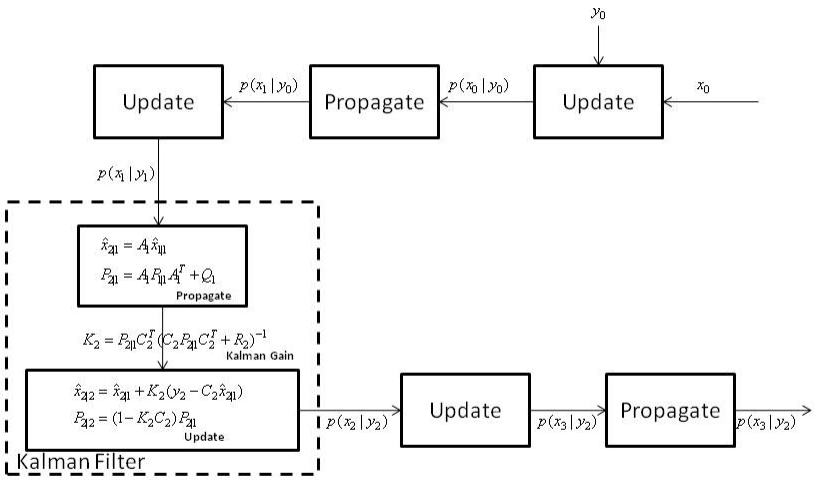

Figure 5: Recursive estimation alorithm

2.3.2 Grid Based Methods

It is hard to obtain analytical solutions to most recursive estimation algorithms. If the state space for a problem is discrete, then we can use grid based methods and can still obtain the optimal solution. Considering that state x can take one of N_s possible values, we can represent discrete density $p(y|x)$ using N_s samples[56].

$$p(x_k|y_t, y_{t-1}, ..., y_0) = \sum_{i=1}^{N_s} w_{k|k}\delta(x_k - x_k^i) \tag{19}$$

where the weights are computed as follows:

$$\begin{aligned} w_{k|k}^i &\triangleq \frac{1}{C} w_{k|k-1}^i p(y_k|x_k^i) \\ &\triangleq \sum_{j=1}^{N_s} w_{k-1|k-1}^i p(x_k^i|x_{k-1}^j) \end{aligned} \tag{20}$$

Here C is the normalizing constant to make total probability equal one. The assumption that state can be represented by finite number of points gives us the ability to sample the whole state space. The weight w_k^i represents the probability of being in state x_k^i when observation at time k is y_k. In grid based method we construct the discrete density at every time instant in two steps. First we estimate the weights at k without the current observation $w_(^i k|k-1)$ and then update them when observation is available and obtain $w_(^i k|k)$. In the propagation step we take into account probabilities (weights) for all possible state values at $k-1$ to estimate the weights at time k as shown in Figure 6. If the prior $p(x_k^i|x_k^j)$ and the

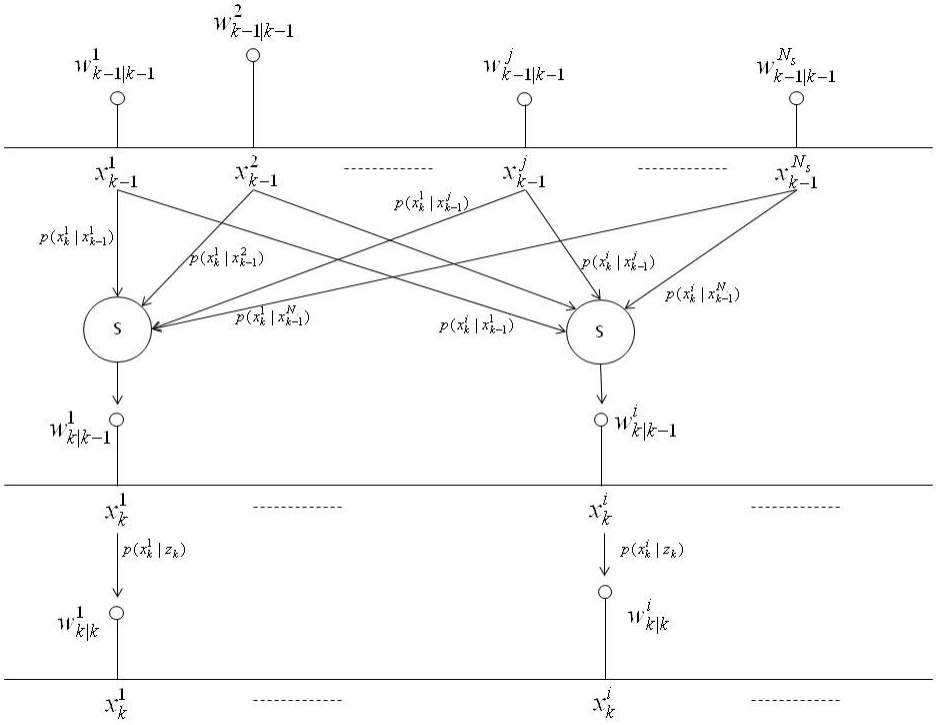

Figure 6: Grid based method

observation probability $p(z_k|x_k)$ are available, the grid based method gives us the optimal solution for tracking the state of the system. If the state of the system is not discrete, then we can obtain an approximate solution using this method. We divide the continuous space into say J cells and for each cell we compute the prior and posterior in a way that takes into

account the range of the whole cell:

$$
\begin{aligned}
p(x_k^i|x_{k-1}^j) &= \int_{x \in x_k^i} p(x|\bar{x}_{k-1}^i)dx \\
p(y_k|x_k^i) &= \int_{x \in x_k^i} p(y_k|x)dx
\end{aligned}
\tag{21}
$$

where $\bar{x}_k$ is the center of j^{th} cell at time $k-1$. The weight update in Equation (21) subsequently remains unchanged.

2.4 Monte Carlo Methods

Grid based methods are a good alternative for situations where analytical solutions are not available. However, they suffer from the limitation that the state has to be represented by a finite set of points. Monte Carlo methods provide another powerful alternative to the cases where close form solutions are intractable [57]. They can approximate both continuous states and discrete states equally well. MC refers to a broad class of techniques that approximate the distribution of a state or a parameter by a set of random samples. The requirement for implementing a successful MC scheme is the availability of a proper sampling process. In other words, a pseudorandom number generator is required that can generate numbers that have properties similar to those of random numbers. Most modern computer tools and languages have built in pseudorandom generators that can generate long sequence of numbers with properties identical to a uniform distribution $\mathcal{U}[0, 1)$, and therefore enable the implementation of MC algorithms.

2.4.1 Sampling

2.4.1.1 Inverse CDF Method

For sampling from a specific distribution, an MC algorithm starts with a pseudorandom number sequence. The inverse CDF method [58] for continuous variables begins by selecting a pseudorandom number ρ_i and then finds the desired sample value by using the relation

$$
x^i = F^{-1}(\rho_i) \tag{22}
$$

where $F(.)$ is the CDF of x, the random variable being sampled. The problem with inverse CDF method is that the inverse function is not always available such as the cases where $f(.)$ can not be integrated or $F^{-1}(.)$ can not be obtained analytically. For such cases, the Rejection method can be used.

2.4.1.2 Rejection Method

Consider a CDF that has support in the region $\mathcal{R} \in [a, b]$ and it is known that the value of $f(.)$ never exceeds a constant M. The rejection method [59] proceeds by first generating a sample from the uniform distribution $\mathcal{U}[0, 1)$, followed by the generation of another sample ρ_i, before selecting x^i based on the following: accept x^i if $\rho_i M \leq f(x^i)$ and reject otherwise.

2.4.1.3 Composition Method

Composition method [57] is used where a PDF is difficult to sample from but can be split into multiple PDFs, which are easier to sample from. For example, if we can write a PDF $f(x)$ as

$$f(x) = A_1 f_1(x) + A_2 f_2(x) + ... + A_i f_i(x) + ... + A_n f_n(x), \tag{23}$$

and all $f_i(x)$ are proper PDFs, then the composition method can be applied. First, a ρ_i is selected. Based on it's value, one of the i-component is chosen. Then, the sample is generated from the selected component PDF using the inverse CDF or the rejection method. Note that the A_i's can be considered as the weight of the PDF $f_i(x)$. Samples from the Gaussian mixture models (GMMs) are drawn using the composition method.

2.4.2 Scoring

The heart of the MC methods is to obtain the expected value

$$\langle z \rangle = \int_a^b z(x) f(x) dx, \tag{24}$$

which is approximated using

$$\bar{z} = \frac{1}{N} \sum_{i=1}^{N} z(x_i). \tag{25}$$

According to the law of large numbers

$$\lim_{N\to\infty} \bar{z} = \langle z \rangle. \tag{26}$$

The standard deviation of the sample mean is given by

$$S(\bar{z}) = \frac{1}{\sqrt{N-1}} \sqrt{\bar{z^2} - \bar{z}^2}. \tag{27}$$

To obtain an accurate value of $\bar{z}$, the standard deviation must be small. From Equation (27), it can be observed that either the sample size N should be increased or the difference between $\bar{z^2}$ and $\bar{z}^2$ should be reduced. Here, two popular methods are described to reduce the standard deviation.

2.4.2.1 Importance Sampling

The power of MC lies in the fact that the sampling can be biased to get more samples in the regions of interest [60]. Subsequently, to produce an unbiased estimate of the expected value, weights are assigned to the samples. Consider that for a PDF $f(x)$ that needs to be sampled, almost all the value of $\langle z \rangle$ comes from a small region of the support of $f(x)$. In such a case, an efficient sampling scheme would be to give preference to the regions where the value of $|z(x)f(x)|$ is large. To implement this scheme, an arbitrary PDF $f^*(x)$ is introduced with the desired properties, and consequently

$$\langle z \rangle = \int \frac{z(x)f(x)}{f^*(x)} f^*(x)d(x) \equiv \int z^*(x)f^*(x)dx = \langle z^* \rangle. \tag{28}$$

Here, the function $W(x) \equiv \frac{f(x)}{f^*(x)}$ is the weight assigned to the samples and z and z^* have the same expectation. Now, if $f^*(x)$ is chosen such that $W(x) < 1$ over regions where $z(x)$ makes a large contribution to the expected value, then

$$\langle z^{*2} \rangle = \int z^2(x)W(x)f(x)d(x) < \int z^2(x)f(x)d(x). \tag{29}$$

It follows that $\sigma^2(z^*) < \sigma^2(z)$.

2.4.2.2 *Splitting*

In the splitting method, an attempt is made to sample more frequently from attractive regions and less frequently from unattractive regions. Attractive regions are the ones where variance of $z(x)$ is large or cost of picking x^i and evaluating $z(x)$ is small. In unattractive regions, the samples are generated with a predetermined probability q_m. otherwise, no sample is taken with a probability $1 - q_m$. The more unattractive the region, the smaller is the value of q_m. On the contrary, the samples in the attractive regions are split into multiple samples. Splitting decreases the variance per selected subregion.

2.4.3 Morkov Chain Monte Carlo (MCMC)

MCMC [61] is a case of MC where samples are generated from a Markov chain. MCMC can be used to sample single parameter distributions as well as multiple parameter distributions. Consider a set of parameters, $a_1, a_2, ..., a_p$, such that we want to estimate $p(a_1, a_2, ..., a_p|y_1, y_2, ..., y_n)$, where $y_1, y_2, ..., y_n$ is the set of observations. Given a starting point $a_1^{(0)}, a_2^{(0)}, ..., a_p^{(0)}$, the MCMC generates $a_j^{(s)}$ from $a_1^{(s-1)}, a_2^{(s-1)}, ..., a_{j-1}^{(s-1)}, a_{j+1}^{(s-1)}, ..., a_p^{(s-1)}$ as follows [62]:

1. Sample $a_1^{(s)} \sim p(a_1^{(s)}|a_2^{(s-1)}, a_3^{(s-1)}, ..., a_p^{(s-1)})$
2. Sample $a_2^{(s)} \sim p(a_2^{(s)}|a_1^{(s-1)}, a_3^{(s-1)}, ..., a_p^{(s-1)})$

   $\vdots$

3. Sample $a_p^{(s)} \sim p(a_1^{(s)}|a_2^{(s-1)}, a_3^{(s-1)}, ..., a_{p-1}^{(s-1)})$

The resultant samples can be seen from two different perspectives. First, the sequence of the following parameter vectors are dependent.

$$\begin{aligned} \mathbf{a^{(1)}} &= \{a_1^{(1)}, a_2^{(1)}, ..., a_p^{(1)}\} \\ \mathbf{a^{(2)}} &= \{a_1^{(2)}, a_2^{(2)}, ..., a_p^{(2)}\} \\ &\vdots \\ \mathbf{a^{(s)}} &= \{a_1^{(s)}, a_2^{(s)}, ..., a_p^{(s)}\} \end{aligned} \tag{30}$$

Note that the parameter $\mathbf{a^{(s)}}$ is conditionally independent on $\mathbf{a^{(1)}}, \mathbf{a^{(2)}}, ..., \mathbf{a^{(s-2)}}$ given $\mathbf{a^{(s-1)}}$ and therefore called the Markov chain. The other perspective is that the marginal distribution of parameter $\mathbf{a_j}$ is given by $a_j^{(0)}, a_j^{(1)}, ..., a_j^{(s)}$. This sampling distribution approaches the target distribution as $s \to \infty$. These samples can also be used to approximate the expected value of a function g using

$$\frac{1}{S}\sum_{s=1}^{S} g(\mathbf{a_j}) \to E[g(\mathbf{a_j})] \tag{31}$$

The above sampling scheme, known as the Gibbs sampling, can only be implemented if the conditional probabilities can be directly sampled. If direct sampling from the conditional distributions is not possible, then a more general Metropolis sampling scheme [63] can be used. It proceeds as follows:

1. Sample from a proposal distribution $J(a_i^{n*}|a_i^n)$ using any sampling method described before. The proposal distribution must be symmetric, i.e., it must satisfy the condition $J(a|b) = J(b|a)$

2. Evaluate the quantity γ, called the Metropolis ratio

$$\gamma = \frac{p(a_1^{n*}|a_2^n, a_3^n, ...)}{p(a_1^n|a_2^n, a_3^n, ...)} \tag{32}$$

3. Choose a_1^{n+1} using the following relation

$$a_1^{n+1} = \begin{cases} a_1^{n*}, & \text{with probability } \min(1, \gamma) \\ a_1^n, & \text{with probability } 1 - \min(1, \gamma) \end{cases} \tag{33}$$

Choosing a_1^n means that a replica of sample a_1^n is created for a_1^{n+1} and there are now two samples at the same location. Practically, the weight of the sample is increased to offset the effect of multiple samples at the same location. The samples for $a_2^n, a_3^n, ...$ can be sequentially obtained in the same fashion. If some conditional distributions can be sampled and some not, then a combination of Gibbs and Metropolis sampling can be used.

2.4.4 Particle Filters

Consider a problem of filtering where we want to estimate the full trajectory of a state sequence given all the observations, i.e., $p(x_{1:n}|y_{1:n})$. This problem can be thought of as a tracking problem where a the track of the system's current location is being maintained, given noisy observations. To obtain an approximate distribution of $x_{1:n}$ using MC methods, one has to take into account the complexity involved. The dimension of the distribution is very high and will only increase as n becomes larger. Therefore, it is unfeasible to approximate this kind of distribution using the MC schemes described above.

2.4.4.1 Sequential Importance Sampling

The solution to obtaining an MC approximation to tracking problems is based on selecting an importance distribution that has the following structure:

$$\begin{aligned} q_n(x_{1:n}) &= q_{n-1}(x_{1:n-1})q_n(x_n|x_{1:n-1}) \\ &= q_1(x_1)\prod_{k=2}^{n} q_k(x_k|x_{1:k-1}). \end{aligned} \tag{34}$$

The weights are an important component of any IS algorithm. To obtain a distribution for $x_{1:n}$, the weight is computed as

$$w_n(x_{1:n}) = \frac{p_n(x_{1:n})}{q_n(x_{1:n})}, \tag{35}$$

where $p_n(x_{1:n})$ is the distribution at time n. The weights obtained using Equation (35) are not normalized. However, for convenience of notation, we will not differentiate between the non-normalized and normalized weights. Based on Equation (34), weight update equation can be written as:

$$\begin{aligned} w_n(x_{1:n}) &= w_{n-1}(x_{1:n-1})\alpha_n(x_{1:n}) \\ &= w_1(x_1)\prod_{k=2}^{n} \alpha_k(x_{1:k}), \end{aligned} \tag{36}$$

where

$$\alpha_n(x_{1:n}) = \frac{p_n(x_{1:n})}{p_{n-1}(x_{1:n-1})q_n(x_n|x_{1:n-1}}). \tag{37}$$

The computational effort required to sample from $q(.)$ and compute $\alpha(.)$ in this case is independent of n.

2.4.4.2 Resampling

SIS provides an efficient solution for sampling from distributions encountered in filtering tracking problems. However, it suffers from the degeneracy problem. The variance of the estimated parameter using SIS increases exponentially with n. At some stage, all but one particle will be left with weights close to zero and the computational effort required to process the samples with near zero weights is wasted. Degeneracy can be estimated from the approximate effective sample size [64] [65]

$$\hat{N}_eff = \frac{1}{\sum_{i=1}^{N_s}(W_k^i)^2}. \tag{38}$$

A small $\hat{N}_eff$ indicates severe degeneracy. The solution to the degeneracy problem is resampling, where particles that have small weights are eliminated and a new set of samples are generated.

2.4.4.3 Sequential Monte Carlo

An SIS algorithm combined with the resampling step is called a Sequential Monte Carlo (SMC) algorithm. Particle filter is a SMC method. Other names for Particle filtering are bootstrap filtering, the condensation algorithm interacting particle approximations and the survival of the fittest. Particle filtering (PF) is a popular way to model signals emanating from a dynamical system. If the underlying state transition is known and the relationship between the system state and the observed output is available, then the system state can be found using Monte Carlo simulations [66]. Consider the discrete time Markov process such that

$$\begin{aligned} X_1 &\sim \mu(x_1) \\ X_t|(X_{t-1} = x_{t-1}) &\sim p(x_t|x_{t-1}) \\ Y_t|(X_t = x_t) &\sim p(y_t|x_t) \end{aligned} \tag{39}$$

We are interested in obtaining $p(x_t|y_t, y_{t-1}, y_{t-2}, ..., y_0)$ so that we have a filtered estimate of x_t from the measurements available so far. If the state space model for the process is available, and both the state and the observation equations are linear, then Kalman filter described above can be used to determine the optimal estimate of x_t given observations $y_t, y_{t-1}, y_{t-2}, ..., y_0$. Kalman Filter can only be used under the condition that the process and observation noises are white Gaussian noise with zero mean and are mutually independent. In case the state and observation equations are nonlinear, the Extended Kalman Filter (EKF) [67], which is a modified form of the Kalman filter, can be used. Particle filter algorithm estimates the state's posterior density, $p(x_t|y_t, y_{t-1}, y_{t-2}, ..., y_0)$ represented by a finite set of support points [56]:

$$p(x_t|y_t, y_{t-1}, ..., y_0) = \sum_{i=1}^{N_s} w_t^i \delta(x_t - x_t^i) \tag{40}$$

where x_i^t for $i = 1, ..., N_s$ are the support points and w_t^i are the associated weights. We thus have a discretized and weighted approximation of the posterior density without the need of an analytical solution. In the PF algorithm, the support points are determined based on the concept of importance sampling, in which we draw points from a particular distribution $q(.)$. The algorithm proceeds as follows:

1. At $n = 1$, sample $x_1^i \sim q(x_1)$

2. Compute weight w_1 using

$$w_1^i \propto \frac{p(x_1^i)}{q(x_1^i)}, \tag{41}$$

   which is an IS approximation, so that

$$p(x_1^i) = \sum_{i=1}^{N_s} w_1^i \delta(x_1 - x_1^i). \tag{42}$$

3. Resample to eliminate the weights that have values close to zero, and split the weights with greater weights.

4. For $n \geq 2$,

(a) the importance density is chosen such that it can be factorized:

$$q(x_{0:n}) = q(x_n|x_{n-1})q(x_{0:n-1}), \tag{43}$$

and the new weights are sampled using

$$x_n^i \sim q(x_n|x_{n-1}). \tag{44}$$

(b) The weight update equation is obtained as follows:

$$w_n^i \propto \frac{p(x_{0:n}^i|z_{1:n})}{q(x_{0:n}^i)}. \tag{45}$$

Here

$$\begin{aligned} p(x_{0:n}|z_{1:n}) &= \frac{p(z_n|x_{0:n}, z_{1:n-1})p(x_{0:n}|z_{1:n-1})}{p(z_n|z_{1:n-1})} \\ &\propto p(z_n|x_{0:n}, z_{1:n-1})p(x_{0:n}|z_{1:n-1}). \end{aligned} \tag{46}$$

And since $p(x_{0:n}) = p(x_n|x_{0:n-1}).p(x_{0:n-1})$, we can write

$$p(x_{0:n}|z_{1:n}) \propto p(z_n|x_n)p(x_n|x_{n-1})p(x_{0:n-1}|z_{1:n-1}). \tag{47}$$

Combining Equations (43) and (47), the weight update Equation (45) can be rewritten as

$$\begin{aligned} w_n^i &\propto \frac{p(x_{0:n-1}^i|z_{1:n-1})}{q(x_{0:n-1})^i} \frac{p(z_n|x_n^i)p(x_n^i|x_{n-1}^i)}{q(x_n^i|x_{n-1}^i)} \\ &\propto w_{n-1}^i \frac{p(z_n|x_n^i)p(x_n^i|x_{n-1}^i)}{q(x_n^i|x_{n-1}^i)}. \end{aligned} \tag{48}$$

The sampling distribution can equally well be dependent on the observation along with the previous state. In that case, $q(x_n^i|x_{n-1}^i)$ will be replaced with $q(x_n^i|x_{n-1}^i, z_n)$

5. Resample $\{w_n^i, x_n^i\}$ to obtain N equal weight particles $\{\frac{1}{N}, \hat{x}_n^i\}$ if required.

A general implementation of a particle filter is shown in Figure 7. The samples have the flexibility to represent diverse distributions including multimodal ones. Also, note that the particles with the largest weights will produce more child particles and some of the particles with smaller weights will not be resampled at all.

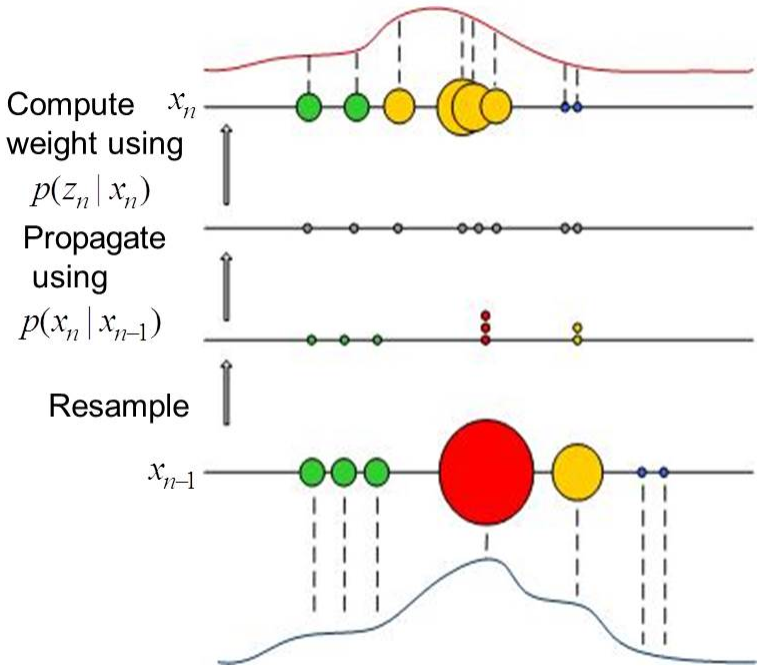

Figure 7: A General Particle Filter Implementation

2.5 HMM and PF Integration

HMMs have been widely used and studied for speech recognition [16] and other speech related technologies such as speech enhancement [68], speech synthesis [69], and statistical language modeling [70]. The wide acceptance of HMMs for modeling speech signals is due to their ability to capture diverse statistical information and accommodate warping of time axis associated with the natural variations in the speed of speech. HMMs are ideally suited for signals that go through a sequentially changing behavior, where the properties remain steady for a certain period, are then followed by a transition (rapid or gradual), before going through a steady period again, and so forth. HMMs capture the statistical properties of the signal in the short steady periods. Human speech is like the signal described, and hence, a good application for the HMMs.

From the human psychoacoustics literature, we learn that ”in selectively attending to one talker in the presence of interfering sounds, listeners exploit principles stemming from

the physical structure of sounds that indicate their origin in a single source" [71]. Inspired by this information, we will investigate whether the physical structure of the human speech, captured by the HMMs, can be used in the speech tracking framework. For this goal, the inherent flexibility of the PF can be exploited, and integration of HMMs with PF can be pursued. This section will dwell on the possibility of using HMMs within the PF algorithm.

2.5.1 Introduction to HMMs

HMMs model a process that is going through a series of states, and is generating an observation from the probability distribution of state it is visiting. The states follow a Markov chain. Following are the three elements of an HMM:

1. The first element, the state of the model, may or may not represent a real quantity. The number of states is always finite and each state represents some statistical properties of the process modeled by the HMM.

2. The second element of the HMM are the state transition probabilities, which define the probability of each state transitioning to another state at a particular time instant. The state can transit to itself or to any other state in the model.

3. The third element of the HMM is the set of observation probabilities. An observation probability distribution is defined for each state and can be discrete or continuous. Gaussian mixture models (GMMs) are commonly used as observation probabilities for speech related HMMs.

A comparison of HMMs with the Kalman Filter is given in Table 1

If we look at the properties listed in the Table 1, PF has more similarities with KF compared to an HMM. The difference between PF and Kalman filter is that the earlier does not restrict the state space and observation models to be linear. Otherwise, both are quite similar.

Table 1: A comparison of Kalman filter and HMM

	Kalman Filter	**HMM**
1.	Based on state space model	Is a stochastic signal model. Can be viewed as a state space model with discrete latent varibles
2.	The state can take any value in the continuum of its space	Can only have a limited number of discrete states
3.	The state represents some real quantity, for example, the position or speed of a target	States can be used merely as a modeling strategy and do not have to represent any real quantity
4.	Observation has a linear relation to the state. The inaccuracy of the relation is represented by the observation noise	The observation model is probabilistic and its distribution is learned from training data
5.	The state space model is linear	The state transition is probabilistic and its distribution is learned from training data

2.5.2 Description of HMMs

We start the description of HMMs with the description of an observable model because it is a precursor of the HMM.

2.5.2.1 Observable Markov Models

A three state observable model is shown in Figure 8. The state here is a real observable event. The figure depicts a one step Markov process, whose description is given by the equation:

$$p[q_t = S_j | q_{t-1} = S_i, q_{t-2} = S_k, ...] = p[q_t = S_j | q_{t-1} = S_i], \tag{49}$$

where q_t is the actual state at time t. Furthermore, the process is assumed to be independent of time, and hence, the state transition probabilities between any two states remain fixed. I this case, the complete system can be described by the state transition probabilities a_{ij},

where,

$$\begin{aligned} a_{ij} &= p[q_t = S_j | q_{t-1} = S_i], 1 \leq i, j \leq N \\ a_{ij} &\geq 0 \\ \sum_{j=1}^{N} a_{ij} &= 1. \end{aligned} \tag{50}$$

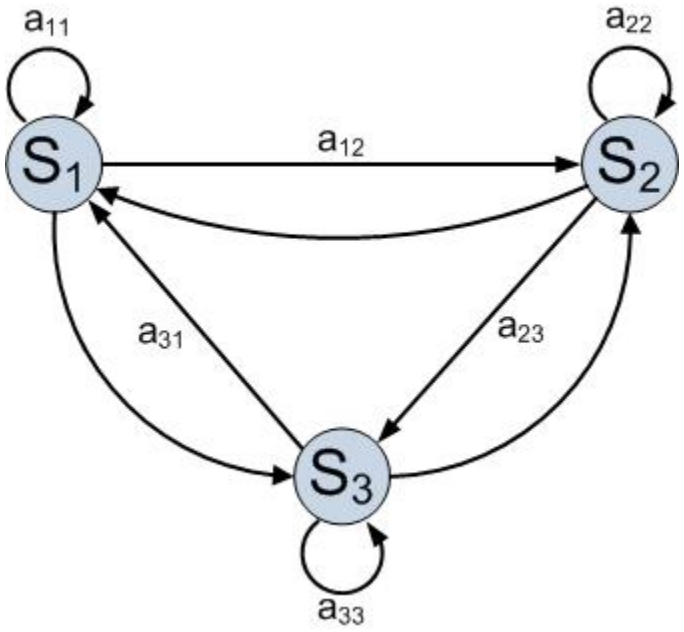

Figure 8: Observable Markov models

2.5.2.2 Hidden Markov models

Most modern speech recognition systems are based on HMMs. Each individual HMM may be used to characterize a phone, triphone or a whole word. The HMM is an extension of the observable Markov model in that the state of an HMM is not observable. Instead, each state of the HMM has a corresponding output probability distribution and the the observation is drawn from this distribution. A typical left-to-right HMM used in ASR is shown in Figure 9.

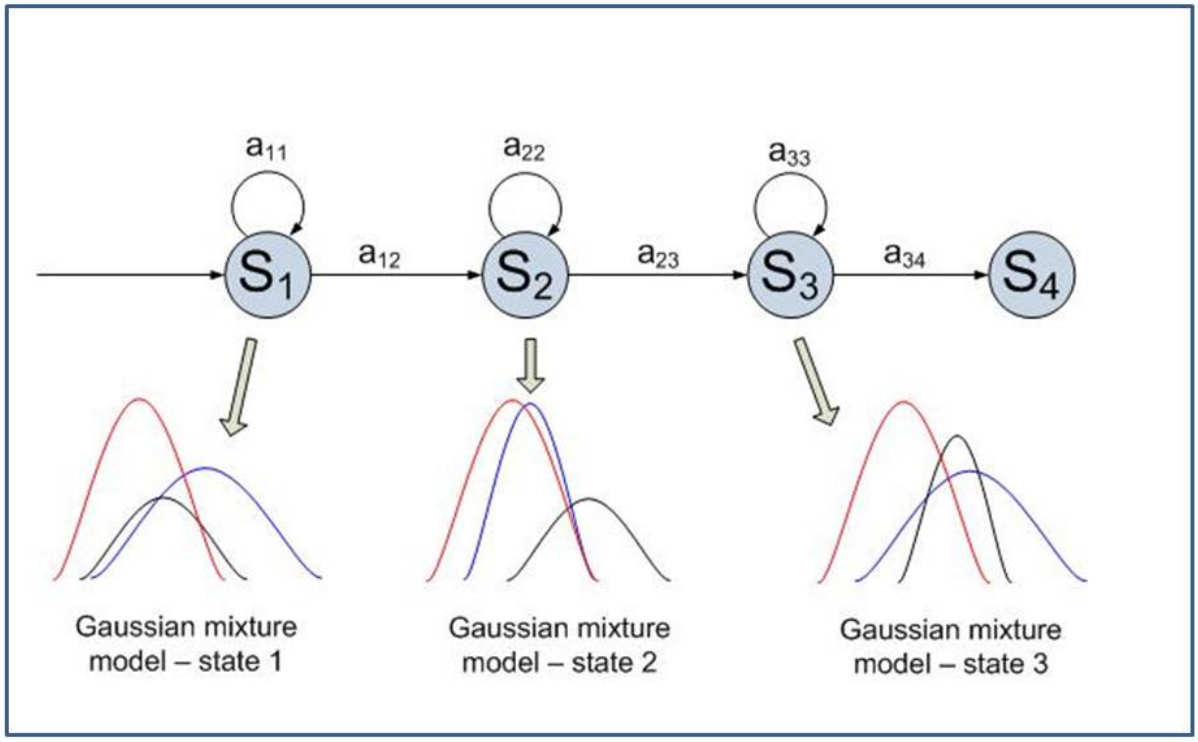

Figure 9: Hidden Markov Models

Assuming that $b_m(i, x)$ is the output probability distribution of the i^{th} state of the recognition class m, then $b_m(i, x)$ can be represented as

$$b_m(i, x) = \sum_k c_{i,k}^m \mathcal{N}(x, \mu_{i,k}^m, \Sigma_{i,k}^m), \tag{51}$$

where $\mathcal{N}(x, \mu_{i,k}^m, \Sigma_{i,k}^m)$ and $c_{i,k}^m$ are the individual Gaussian distributions and mixture coefficients of the k^{th} Gaussian mixture component, and the parameters $\mu_{i,k}^m$ and $\Sigma_{i,k}^m$ represent the mean and covariance of Gaussian component k. For reducing computational complexity, $\Sigma_{i,k}^m$ is constrained to be a diagonal matrix.

The transition probabilities of the states are the same as the ones defined in Equation (50) for the observable models. The complete representation of an HMM is given by $\lambda_m = \{A_m, B_m\}$, where

1. $B_m = \{c_{i,k}^m, \mu_{i,k}^m, \Sigma_{i,k}^m\}$ is the set of GMM observation probabilities, and $c_{i,k}^m$ is the weight associated with the k^{th} mixture of the GMM.

2. $A_m = [a_m(i, j)]$ is the set of transition probabilities.

Given the HMM state sequence $S = \{s_1, s_2, ...s_N\}$ and model parameters λ_m, the likelihood of a feature vector sequence $X = \{x_1, x_2, ..., x_N\}$ is computed using

$$P(X|S, \lambda_m) = \prod_i P(x_i|s_i, \lambda_m). \tag{52}$$

The complete likelihood of the feature vector sequence is obtained using

$$P(X|\lambda_m) = \sum_S P(X|S, \lambda_m) = \sum_S \Big[\prod_t P(x_t|s_t, \lambda_m)\Big] P(S|\lambda_m), \tag{53}$$

where $P(x_t|s_t, \lambda_m)$ is the state observation probability, $P(S|\lambda_m)$ is the state transition probability generated from the transition matrix A_m:

$$P(S|\lambda_m) = \prod_t a_s(s_t, s_{t+1}) p(s_1). \tag{54}$$

2.5.3 HMMs as generative models

Instead of considering HMMs for estimating the likelihood of an observed feature vector, we will now look at an HMM as a generator of some feature vectors. In other words, is it possible to generate feature vector samples from an HMM? Figure 10 shows the example of the digit *two*, sampled using an HMM that was trained with 45 handwritten *twos* [72]. The top row shows some of the actual handwritten digits used to train the HMM, while the bottom row shows the digits generated using the HMM. It can be noted that all the curves traced by the human hand to write a *two* have been captured by the HMM and subsequently reproduced in the artificial *two*. The models potential for sample generation is apparent. HMM as a sample generator can also be justified based on the fact that, when modeling a signal with an HMM, it is an underlying assumption that the signal is generated from such a model.

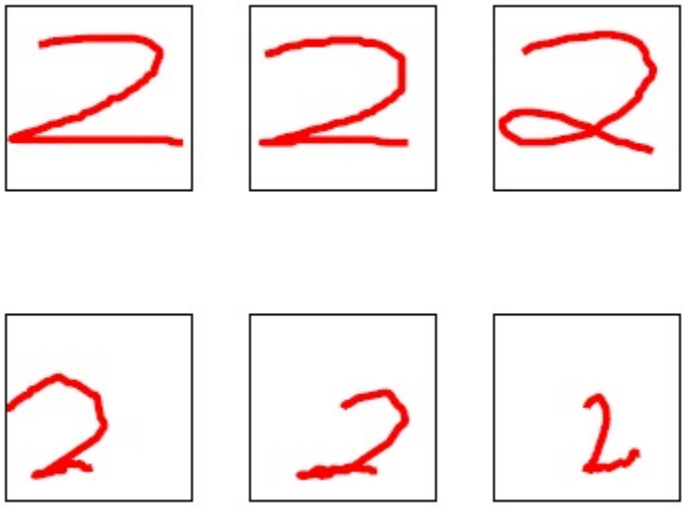

Figure 10: An example of on-line handwritten digit

The sampling process will comprise of two steps. First, a state will be picked based on the state transition probabilities. Once an appropriate state has been selected, samples will be generated from its observation probability. If an HMM is intended to be used for sample generation, its observation distributions should preferably be easy to sample from.

Any appropriate scheme from amongst the methods defined in Section 2.4.1 can be used to sample from the HMMs observation distribution.

2.5.4 HMMs within PF

As highlighted in Table 1, HMMs differ in nature from the standard tracking algorithms and by themselves, have limited capability for tracking a continuously varying signal. Both HMMs and PF have states, but these states different in nature. The state of a PF is a real quantity. On the contrary, the states of an HMM may be used only as a modeling strategy. The observation distribution of an HMM, however, is not only a real quantity, but also a valid source for sample generation. Consequently, there is a possibility of utilizing the observation distribution to generate the samples in the PF algorithm. In such a setup, the observation distribution of the HMM will correspond to the state of the PF. The structure can be viewed as a three layer scheme as shown in Figure 11.

The red line is the observed signal, the blue line is the state of the signal being estimated and S_1, S_2 and S_3 in the circle are the HMM states, whose observation distribution is used to generate the samples representing the state. Instead of obtaining the samples from the state space model as is done in a conventional PF algorithm, the samples are generated from the observation model of a particular state of an HMM. The wights of the samples can then be computed using the observed signal. The diameter of the sample in the figure indicates its weight. The idea will be actualized for tracking of speech signals contaminated by noise in the next chapter.

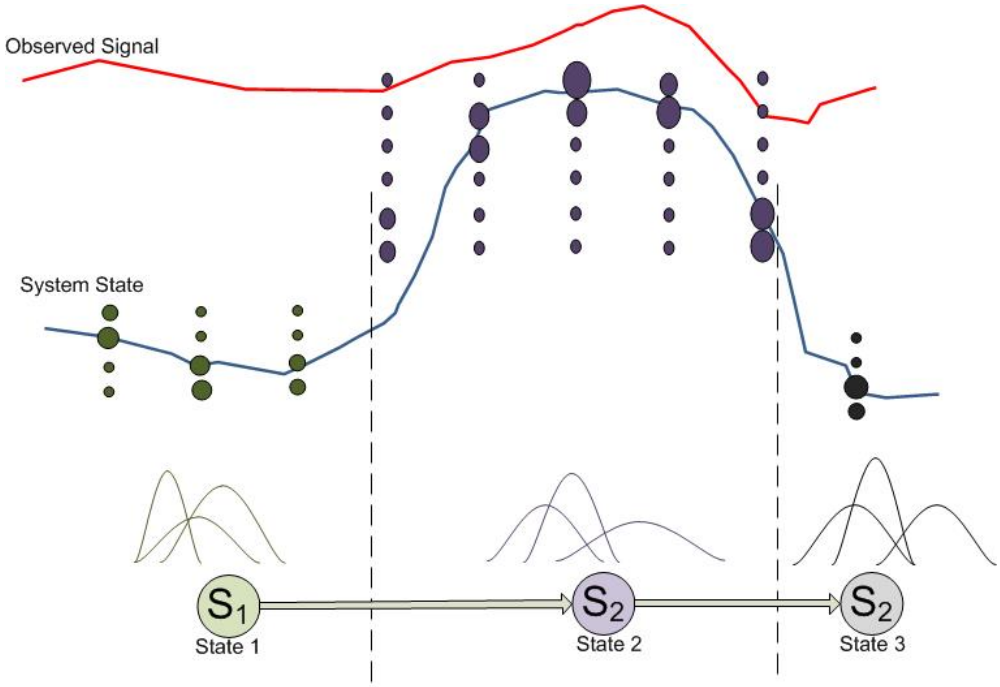

Figure 11: HMM for sample generation

CHAPTER 3

CLEAN SPEECH TRACKING USING PARTICLE FILTER COMPENSATION (PFC)

In this chapter, we develop the integration of the particle filter with HMMs to form the Particle Filter Compensation (PFC) algorithm [11]. The observation model is derived and we describe the specifics of how HMMs, used as side information, act as the replacement of the state transition model in the particle filter framework. Then, it is shown that the challenge of estimating suitable side information can be overcome by merging the HMM sates into a small number of clusters [12]. The performance of the PFC algorithm is evaluated on a connected digit recognition task.

3.1 The Compensation Scheme

If the clean speech is corrupted by an additive noise, n, and a distortion channel, h, then we can represent the noise corrupted speech with an additive noise model [30] shown in Figure 12. Assuming known statistics of the noise parameters,

$$y = x + \log(1 + e^{n-x-h}), \tag{55}$$

where $y = log(S_y(m_p))$, $x = log(S_x(m_p))$ and $h = log(|H(m_p)|^2)$ and $S(m_p)$ denotes the p^{th} mel spectrum.

$$S_y(m_p) = S_x(m_p)|H(m_p)|^2 + S_N(m_p). \tag{56}$$

The additional side information needed for feature compensation is a set of nuisance parameters, Φ. Similar to *stochastic matching* [3], we can iteratively find Φ followed by decoding as shown in Figure 13.

$$\Phi' = \arg\max_{\Phi} p(Y'|\Phi, \Lambda), \tag{57}$$

where Y' is the noisy or compensated utterance.

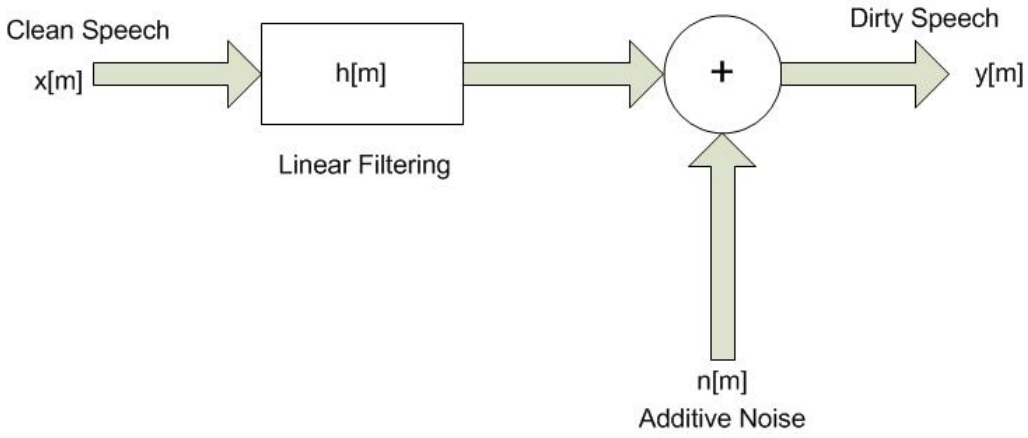

Figure 12: Distortion model

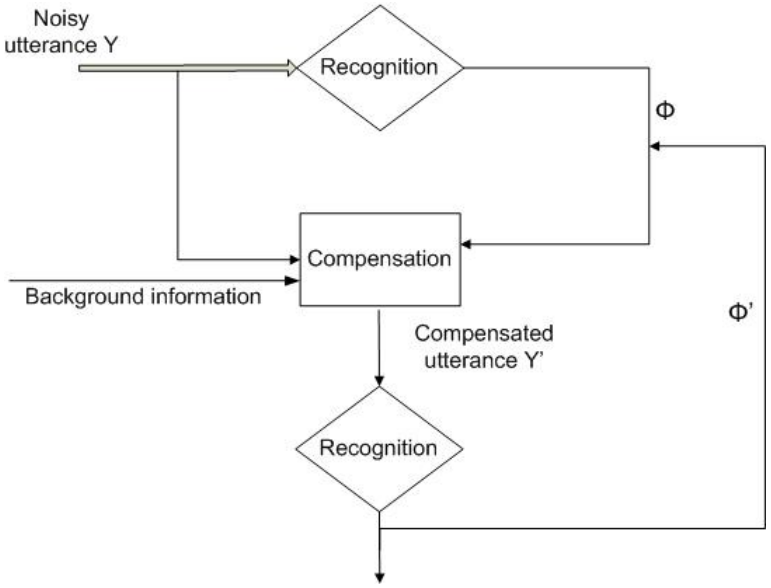

Figure 13: Compensation scheme based on Stochastic Matching

The clean HMMs and the background noise information enable us to generate appropriate samples from $q(.)$ in Equation (48). The parameters Φ in Equation (57) in our particle filter compensation (PFC) scheme, correspond to the corresponding correct HMM state sequence and mixture component sequence. These sequences provide critical information for density approximation in PFC. As shown in Figure 13 this can be done in two stages. We first perform a front-end compensation of noisy speech. Then recognition is done in the second stage to generate the side information Φ so as to improve compensation. This process can be iterated similar to what's done in maximum likelihood stochastic matching [3]. During compensation, the observed speech y is mapped to clean speech features x. For this purpose, clean speech alone cannot be represented by a finite set of points, and therefore, HMMs by themselves cannot be used directly for tracking x.

3.2 PFC

The State transition information is an integral part of the particle filter algorithm, and is used to propagate the particle samples through the time transitions of the signal being processed. Specifically, the state transition is important to be able to position the samples at the right locations. The difficulty of using particle filters for tracking the speech signal lies in obtaining a state space model for speech, as consecutive speech features are usually highly correlated. To solve this problem, statistics from HMMs can be used. Although we only have discrete states in HMMs, each state is characterized by a continuous density Gaussian mixture model (GMM), and therefore, it enables us to capture part of the variation in speech features, to generate particle samples for feature compensation.

3.2.1 The Observation Model

The observation model is used to obtain $p(y_t|x_t^i)$. It's derivation for the case of clean speech corrupted by an additive noise is described here. The distribution of the log spectra of noise for each channel is assumed Gaussian with mean μ_n and variance σ_n^2. Assuming there is

additive noise only with no channel effects

$$y = x + \log(1 + e^{n-s}). \tag{58}$$

We are interested in evaluating $p(y|x)$, where x represents the clean speech and n is the noise with density $\mathcal{N}(\mu_n, \sigma_n)$. Then,

$$\begin{aligned} p[Y < y|x] &= p[x + \log(i + e^{N-x}) < y|x] \\ p(y|x) &= F'(u) \\ &= p(u)\frac{e^{y-x}}{e^{y-x} - 1}, \end{aligned} \tag{59}$$

where $F(\mu))$ is the Gaussian cumulative density function with mean μ_n and variance σ_n^2 and $u = \log(e^{y-x} - 1) + x$. In the case of MFCC features, the nonlinear transformation is [6]:

$$y = x + D\log(1 + e^{D^{-1}(n-x)}). \tag{60}$$

Consequently,

$$p(y|x) = P_N(g^{-1}(y))J_{g^{-1}}(y), \tag{61}$$

where $P_N(.)$ is a Gaussian pdf, $J_{g^{-1}}(y)$ is the corresponding Jacobian and D is a discrete cosine transform matrix which is not square, and thus not invertible. To overcome this problem, we zero-pad the y and x vectors and extend D to be a square matrix. The variance of the noise density is obtained from the available noise samples.

3.2.2 HMMs as State Transition Information in PF

Now we describe how feature compensation can be done using the particle filter algorithm if the side information about the statistics of clean speech is available. Consider that an HMM λ_m is available that adequately represents the speech segment under consideration for compensation, along with an estimated state sequence $s_1, s_2, ..., s_T$ that correspond to T feature vectors. Then, we can generate new samples from the current i^{th} sample according to

$$q(x_t^i|x_{t-1}^i, y_t) \sim \sum_{k=1}^{K} c_{k,s_t}\mathcal{N}(\mu_{k,s_t}, \Sigma_{k,s_t}), \tag{62}$$

where $\mathcal{N}(\mu_{k,s_t}, \Sigma_{k,s_t})$ is the k^{th} Gaussian mixture for the state s_t in λ_m, and c_{k,s_t} is the corresponding weight for the mixture. The total number of particles is fixed, and the contribution from each mixture, computed at run time, depends on its weight. We have chosen the importance sampling density, $q(x_t^i|x_{t-1}^i, y_t)$ in Equation (48) to be $p(x_t|x_{t-1}^i)$ in Equation (62). This is known as the sampling importance resampling (SIR) filter [56]. It is one of the simplest implementations of particle filters, and it enables the generation of samples independently from the observation. For the SIR filter, we only need to know the state and the observation equations and should be able to sample from the prior. Also, the resampling step is applied at every stage and the weight assigned to the i^{th} support point of the distribution of the speech signal at time t is updated as follows:

$$w_t^i \propto p(y_t|x_t^i). \tag{63}$$

Once the point density of the clean speech features is available, we estimate the compensated features using the discrete approximation of the expectation as follow:

$$x_t = \sum_{i=1}^{N_s} w_t^i x_t^i. \tag{64}$$

The compensation is done on the fbank features because good observation model (distortion model) is available for them. The fbank features obtained are then converted to MFCC features, whose discriminative potential can be exploited in the recognition phase. The final recognition result for the test utterance is thus decoded from the compensated MFCC features. A block diagram of the flow of the compensation scheme is shown in Figure 14

3.2.3 Estimation of HMM Side Information

As described above, it is important to obtain $\Phi \in \{\lambda_m, S\}$, where λ_m is an HMM that faithfully represents the speech segment being compensated, and $S = s_1, s_2, ..., s_T$ is the state sequence corresponding to the utterance of length T. To obtain λ_m for the m^{th} word W_m in the utterance, we chose the $N - best$ models $\lambda_{m_1}, \lambda_{m_2}, ..., \lambda_{m_N}$ from HMMs trained using 'clean speech data'. The N models are combined together to obtain a single model λ_m.

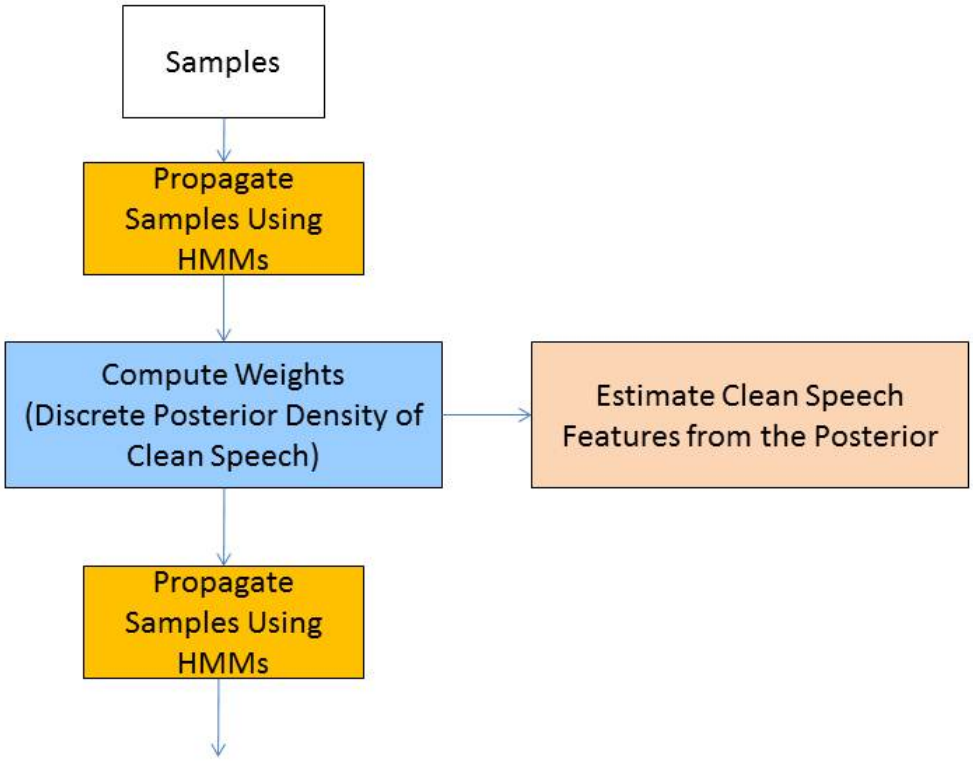

Figure 14: PF Compensation Scheme Block Diagram

3.2.3.1 Gaussian Mixture estimation

To obtain the observation model for each state j of model λ_m, we concatenate mixtures from the corresponding states of all component models as follows:

$$\hat{b}_j^{(m)}(o) = \sum_{l=1}^{L} \sum_{k=1}^{K} c_{k,j}^{ml} \mathcal{N}(\mu_{k,j}^{ml} \Sigma_{k,j}^{ml}), \tag{65}$$

where K is the number of Gaussian mixtures in each original HMM, and L is the number of different words $m_1, m_2, ..., m_L$ in the $N-best$ hypothesis. $\mu_{k,j}^{ml}$ and $\Sigma_{k,j}^{ml}$ are the mean and the covariance from the $k-th$ mixture in the $j-th$ state of model m_l respectively. The mixture weights are normalized by scaling them according to the likelihood of the occurrence of the model, from which they come from:

$$c_{k,j}^{ml} = c_{k,j}^{ml} p(W_m = \lambda_{ml}). \tag{66}$$

The mixture weight is an important parameter because it determines the number of samples that will be generated from the corresponding mixture. The state transition coefficients for λ_m are computed using the following:

$$\begin{aligned} \hat{a}_{ij}^{(m)} &= \sum_{l=1}^{L} p[s_t^{(ml)} = i, s_{t-1}^{(ml)} = j | W_m = \lambda_{ml}] p[W_m = \lambda_{ml}] \\ &= \sum_{l=1}^{L} [a^{(m)} | W_m = \lambda_{ml}] p[W_m = \lambda_{ml}]. \end{aligned} \tag{67}$$

3.2.3.2 State sequence estimation

The recognition performance can be greatly improved if a good estimate of the HMM state sequence S is available, but obtaining this sequence in a noisy operational environment in ASR is very challenging. The simplest approach is to use the decoded state sequence obtained with multi-condition trained models in an ASR recognition process as shown in Figure 13. However, these states could often correspond to incorrect models and deviate significantly from the optimal one. Alternatively, we can determine the states (to generate samples from) sequentially during compensation. For left-to-right HMMs, given the state

s_{t-1} at time $t-1$, we chose s_t as follows:

$$\begin{aligned} s_t &\sim a_{s_t, s_{t-1}} \\ s_t &= \arg\max_i (a_{ij}), \end{aligned} \tag{68}$$

where a comes from the state transition matrix for λ_m. The mixture indices are subsequently selected from amongst the mixtures corresponding to the chosen state.

3.2.4 Experiments

Before the experiment setup is laid out, and the results are analyzed, the speech corpus used for connected digit recognition task is introduced.

3.2.4.1 Aurora-2 Test Set

Speech data in the Aurora-2 corpus are based on the TIDigits [73] speech data downsampled at 8 kHz and filtered through G.712 characteristic to simulate the global system for mobile communications (GSM) channel effect. Eight different noise sources (suburban train, babble, car, exhibition hall, restaurant, street, airport, and train station) were artificially added in a controlled fashion to cover a range of SNR levels. The range includes a no noise condition, referred to as the clean condition, along with six SNR levels 20, 15, 10, 5, 0, and −5 dB.

The Aurora-2 corpus contains two training sets and three test sets. The two training sets are referred to as clean and multicondition training sets. The clean training set does not contain any additive noise. The multicondition training set is representative of four noise types (suburban train, babble, car, and exhibition hall) covering five SNR ratios 20, 15, 10, 5 dB and the clean condition. Both clean training and multicondition training sets consist of 8440 utterances selected from the training part of the TIDigits containing the recordings of 55 male and 55 female adults.

Three test sets are defined as Test Sets A, B and C. Test Set A is representative of all four noise types seen in the multicondition training set. Test Set B is representative of four noise types not represented in the multicondition training set. Test Set C is filtered

through M-IRS filtering to introduce convolutional noise. It contains suburban street and train noises. Each Test set covers all SNR levels 20, 15, 10, 5, 0, and −5 dB and the clean condition. Therefore, Aurora-2 includes a total of 70 different testing conditions (ten different noises with seven SNR levels). Each testing condition includes 1001 utterances selected from the TIDigits test set.

3.2.4.2 Setup

To investigate the properties of the proposed approach, we first assume that a decent estimate of the state is available at each frame. Moreover, we assume that speech boundaries are marked and therefore the silence and speech sections of the utterance are known. To obtain this information, we use a set of digit HMMs (18 states, 3 Gaussian mixtures) that have been trained using clean speech represented by 23 channel mel-scale log spectral features. The speech boundaries and state information for a particular noisy utterance is then captured through digit recognition performed on the corresponding clean speech utterance. The speech boundary information is critical because the noise statistics have to be estimated from the noisy section of the utterance. To get the HMM needed for particle filter compensation L models $\lambda_1, \lambda_2, ..., \lambda_L$ are selected based on the $N-best$ hypothesis list. For our experiments, we set $L = 3$. We combine these models to get ${\lambda'}_m$ for the $m-th$ word in the utterance.

3.2.4.3 Clean FBANK Estimation

To see the efficacy of the compensation process, we consider the noisy, clean and compensated filter banks (channel 8) for the whole utterances shown in Figure 15. The SNR for this particular case is $5dB$. When compared with the noisy feature (upper solid curve in Figure 15) we can see that the compensated feature (lower dash curve matches well with the clean feature (middle dash curve in Figure 15) of the shown utterance. It should be noted however that such a good restoration of the clean speech signal from the noisy signal is achievable only when a good estimate of the side information about the state and mixture component sequences is available.

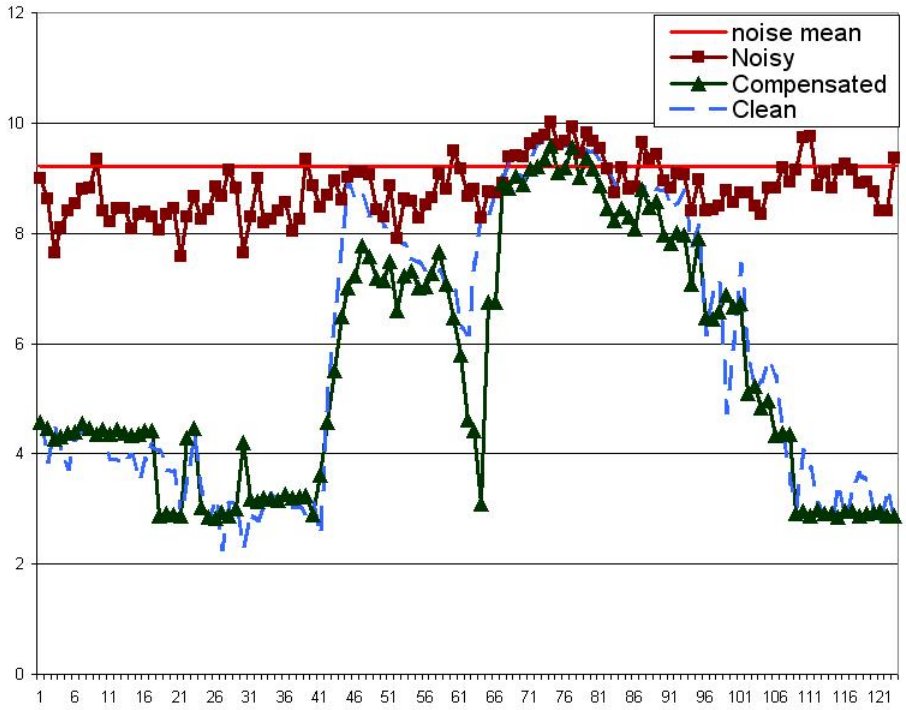

Figure 15: Actual vs. estiamted speech at 5dB noise level

Next, we show the colormap of the noisy, compensated, and the clean fbank features of two utterances at different SNRs. The clormap allows us to view the 23-fbank channels simultaneously. The left, center, and the right panes of Figures fig:Colormap1 and fig:Colormap2 show the noisy, compensated, and the clean fbank features respectively. Figue fig:Colormap1 shows the spoken utterance "*six eight eight*" at SNR 15 while Figure fig:Colormap2 shows the spoken utterance "*eight zero zero nine eight five*" at SNR 5 The efficacy of PFC can be seen from the transformation of the noisy features to a new set of features that are more similar to the reference clean speech features.

3.2.4.4 Oracle Experiment

To analyze the efficacy of the PFC algorithm and evaluate it's performance, three types of experiments are performed; oracle, artificial word boundary and state alignment and Viterbi state estimation. The experiments vary in the state estimation part only, other components of the PFC scheme are constant throughout. Aligning the word boundaries means that the

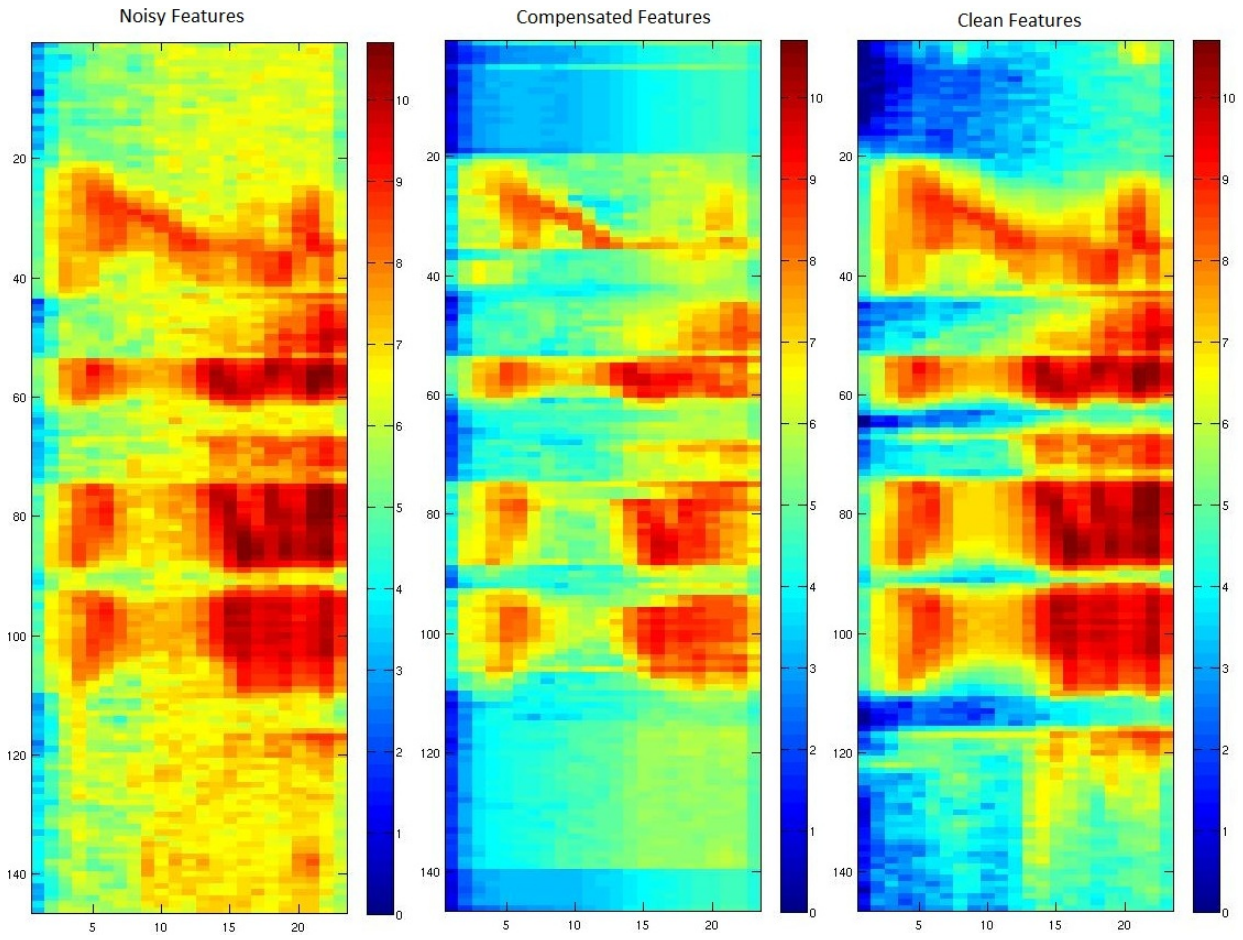

Figure 16: Compensated vs. clean features at 15dB noise level

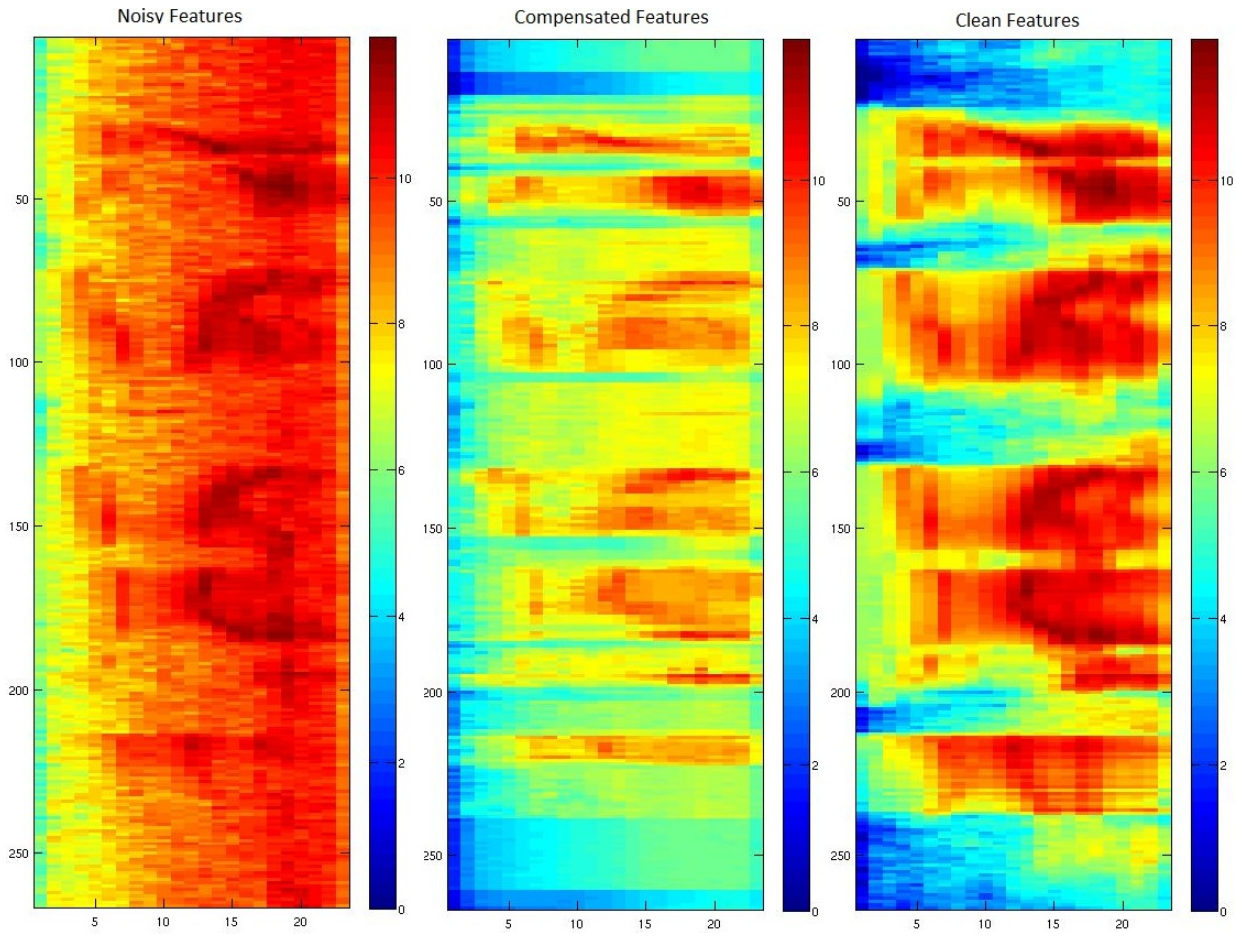

Figure 17: Compensated vs. clean features at 5dB noise level

best possible word boundary information is available and used in the compensation process. To elaborate further, when compensating a noisy utterance, a recognition is performed on the clean version of the utterance (in Auror-2, a clean version of each and every noisy utterance is available) with clean HMMs. Testing a clean utterance with clean HMMs will give the best possible information about the utterance, even though the information may not be perfect.

After obtaining the best possible word boundaries, the state sequence within the word boundaries is obtained using the standard Viterbi algorithm on the noisy utterance. The states at this stage are not well aligned but are still better compared to the case where the word boundaries are not available. Next, we gradually align the state information within the word boundaries (best possible state information is obtained in a similar fashion to the way the word boundaries were obtained). Starting from *one* state, the number of states aligned in this way is increased gradually to the maximum of 16, at which stage, the best possible state alignment (HMM side information) is available for use in the PFC. Figure 18 shows the recognition performance for four conditions (2 known and 2 unknown additive noise conditions) as more and more states are aligned.

The results point to the fact that if the correct state is used in PFC, i.e., the samples are placed at the appropriate locations, then very good recognition performance can be achieved. Also, if the state used for the generation of samples contains statistical information similar to the correct state, then a good performance can still be achieved.

3.2.4.5 Oracle vs Real Experiment

In the case of the actual operational scenarios, when no side information is available, models were chosen from the $N-Best$ list while the states were computed using Viterbi decoding. Of course, the states would correspond to only one model which might not be correct, and there might be a significant mismatch between actual and computed states. Moreover the misalignment of words also exacerbated the problem. The results for this case (Adapted Model III as shown in Table 2 Column 4) were only marginally better than those obtained

with the multi-condition trained models.

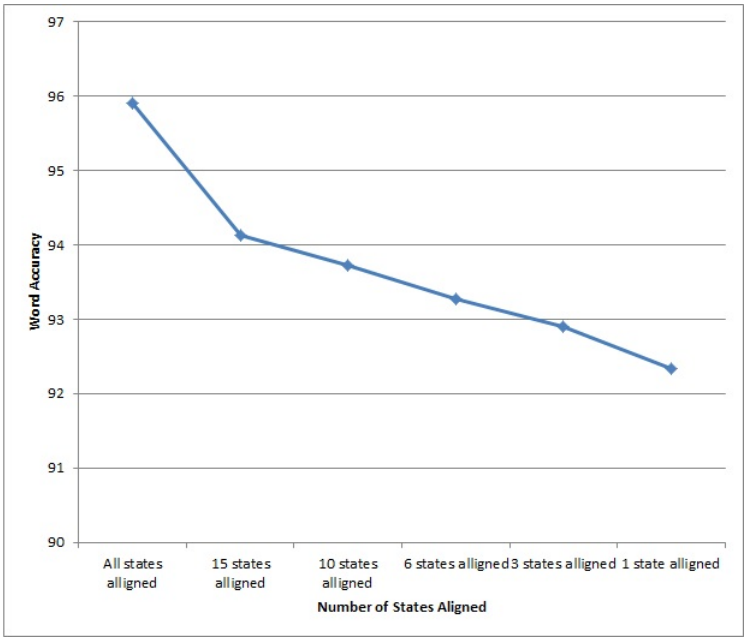

Figure 18: Performance with various state alignments

Table 2: ASR accuracy comparisons for Aurora-2

Word Accuracy	**Adapted Models I**	**Adapted Models II**	**Adapted Models III**	**Multi Cond.**	**Clean Cond.**
clean	99.1	99.1	99.1	98.5	99.11
20dB	97.75	96.46	97.38	97.66	97.21
15dB	97.61	95.98	96.47	96.95	92.36
10dB	96.66	94	94.4	95.16	75.14
5dB	95.2	90.64	88.02	89.14	42.42
0dB	92.13	82.62	68.28	64.75	22.57
-5dB	89.28	72.13	32.92	27.47	NA
0-20dB	95.86	90.23	88.91	88.73	65.94

In collumn three of Table 2, the boundaries of words were extracted from the N-Best list using exhaustive search and the states for the words between these boundaries were assigned by splitting the digits into equal-sized segments and assigning one state to each

segment. This limited the damage done by state misalignment, and it can be seen that a 13% digit error reduction from MC training was observed.

3.3 A Clustering Approach to Obtaining Correct HMM Information

HMM states are used to spread the particles at the right locations for subsequent estimation of the underlying clean speech density. If the state is incorrect, the location of particles will be wrong and the density estimate will be erroneous. One solution is to merge the states into clusters. Since the total number of clusters can be much less than the number of states, the problem of choosing the correct information block for sample generation is simplified. A tree structure to group the Gaussian mixtures from clean speech HMMs into clusters can be built with the following distance measure [18]:

$$\begin{aligned} d(m,n) &= \int g_m(x) \log \frac{g_m(x)}{g_n(x)} dx + \int g_n(x) \log \frac{g_n(x)}{g_m(x)} dx \\ &= \sum_i [\frac{\sigma_m^2(i) - \sigma_n^2(i) + (\mu_n(i) - \mu_m(i))}{\sigma_n^2(i)} \\ &\quad + \frac{\sigma_n^2(i) - \sigma_m^2(i) + (\mu_n(i) - \mu_m(i))}{\sigma_m^2(i)}], \end{aligned} \tag{69}$$

where $\mu_m(i)$ is the $i-th$ element of the mean vector μ_m, and $\sigma_m^2(i)$ is the i-th diagonal element of the covariance matrix Σ_m. The parameters of the single Gaussian representing the cluster, $g_c^k(X) = \mathcal{N}(X; \mu_k, \sigma_k^2)$, is computed as follows:

$$\begin{aligned} \mu_k(i) &= \frac{1}{M_k} \sum_{m=1}^{M_k} E(x_m^{(k)}(i)) \\ &= \frac{1}{M_k} \sum_{m=1}^{M_k} \mu_m^{(k)}(i) \\ \sigma_k^2(i) &= \frac{1}{M_k} \sum_{m=1}^{M_k} E(x_m^{(k)}(i) - \mu_k(i))^2 \\ &= \frac{1}{M_k} \sum_{m=1}^{M_k} \sigma_m^{2(k)}(i) + \sum_{m=1}^{M_k} \mu_m^{2(k)}(i) - M_k \mu_k^2(i). \end{aligned} \tag{70}$$

Alternatively, we can group the components at the state level using the following distance measure [74]:

$$d(n,m) = -\frac{1}{S}\sum_{s=1}^{S}\frac{1}{P}\sum_{p=1}^{P}\log[b_{ms}(\mu_{nsp}) + \log[b_{ns}(\mu_{msp})], \tag{71}$$

where S is the total number of states in the cluster, P is the number of mixtures per state and $b(.)$ is the observation probability. The clustering algorithm proceeds as follows:

1. Create one cluster for each mixture up to k clusters.

2. While $k > M_k$, find n and m for which $d(n,m)$ is the minimum and merge them.

Once clustering is complete, it is important to pick the most suitable cluster for feature compensation at each frame. The particle samples are then generated from the representative density of the chosen cluster. Two methods can be explored. The first is to decide the cluster based on the $N-best$ transcripts obtained from recognition using multi-condition trained models. Denote the states obtained from the $N-best$ transcripts for noisy speech feature vectors at time t as $s_{t_1}, s_{t_2}, ..., s_{t_N}$. If state s_{t_i} is a member of cluster c_k, we increment $M(c_k)$ by one, where $M(c_k)$ is a count of how many states from the $N-best$ list belong to cluster c_k. We choose the cluster based on $\arg\max_k M(c_k)$ and generate samples from it. If more than one cluster satisfies this criterion, we merge their probability density functions. In the second method, we chose the cluster that maximizes the likelihood of the MFCC vector at time t, O_t, belonging to that cluster as follows:

$$C \sim \arg\max_k g_{mc}(O_t|C_k), \tag{72}$$

where $g_{mc}(.)$ represents the probability that O_t corresponds to the cluster C_k.

It is important to emphasize here that $g_{mc}(.)$ is derived from multi-condition speech models and has a different distribution from the one used to generate the samples. The relationship between clean clusters and multi-condition clusters is shown in Figure 19.

Clean clusters are obtained using methods described above. The composition information of these clusters is then used to build a corresponding multi-condition cluster set from multi-condition HMMs. A cluster C_j in clean clusters represents statistical information of a particular section of the clean speech. The multi-condition counterpart C_j represents statistics of the noisy version of the same speech section.

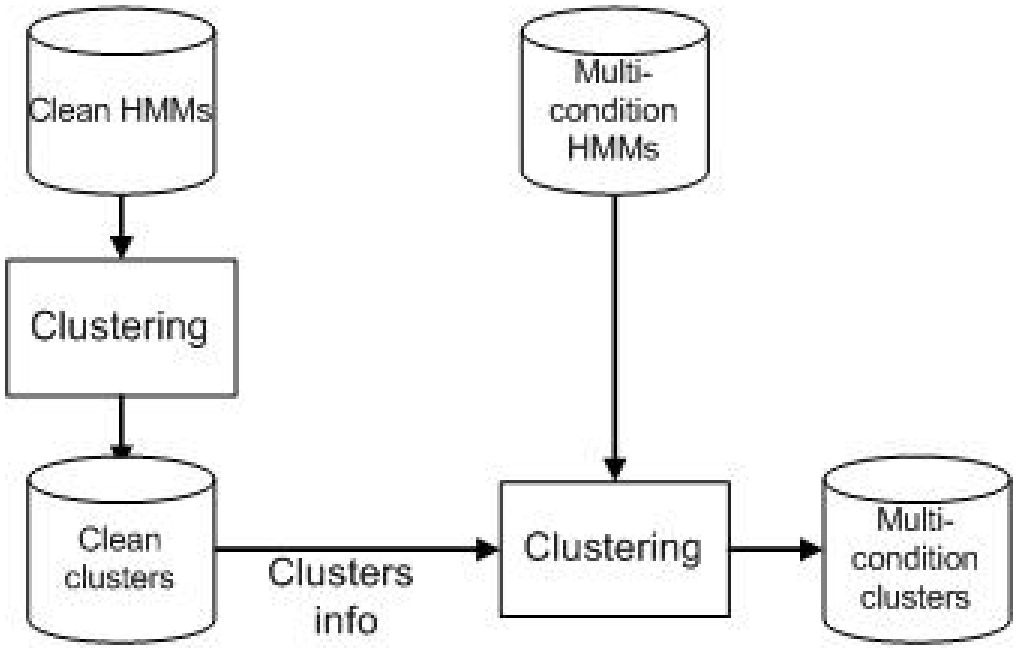

Figure 19: Clustering of multi-condition trained HMMs

Clean clusters are necessary to track clean speech because we need to generate samples from the clean speech distributions. However, they are not the best choice for estimating Equation (72) because the observation is noisy and has a different distribution. The best candidate for computing Equation (72) is the multi-condition cluster set. It is constructed from multi-condition HMMs that match more closely with noisy speech. A block diagram of the overall compensation and recognition process is shown in Figure 20. We make inference about the cluster to be used for observation vector O_t using both the $N-best$ transcripts and Equation (72) combined together. Samples at frame t are then generated using the pdf of the chosen cluster. The weights of the samples are computed using Equation (59) and compensated features are obtained using Equation (64). Once the compensated features are

available for the whole utterance, recognition is performed again using retrained HMMs with compensated features.

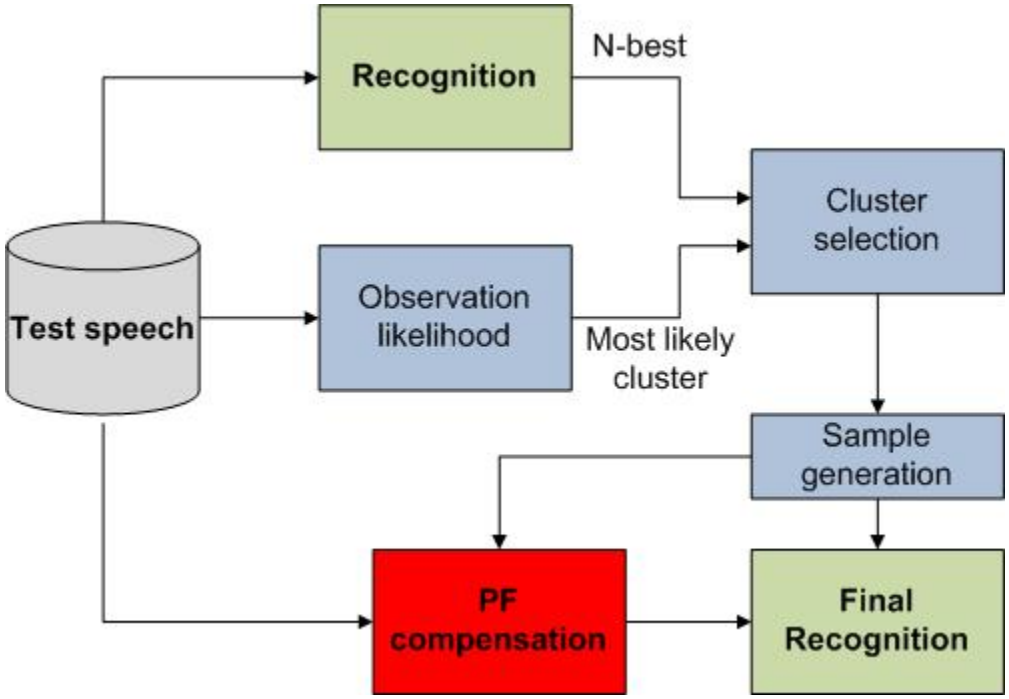

Figure 20: Complete recognition process

3.3.1 Simple Vs Complex Models

In the PFC algorithm, the compensation is done at the front end of the ASR system. Consequently, the HMM set used for ompensation (box 1 of Figure 21), and the one used for recognition (box 2 of Figure 21) can be different and independent of one another. This relaxation can be exploited in the overall compensation and recognition processes. For the compensation phase, simpler models are better since the states are ultimately merged into clusters that represent diverse statistics.

Starting from complex HMMs does not give a significant advantage in the clustering phase and because the statistical information related to a specific speech segment will be lost at some stage. On the contrary, complex models are much more useful for the recognition phase. Here, the objective is to obtain precise information about the speech segment

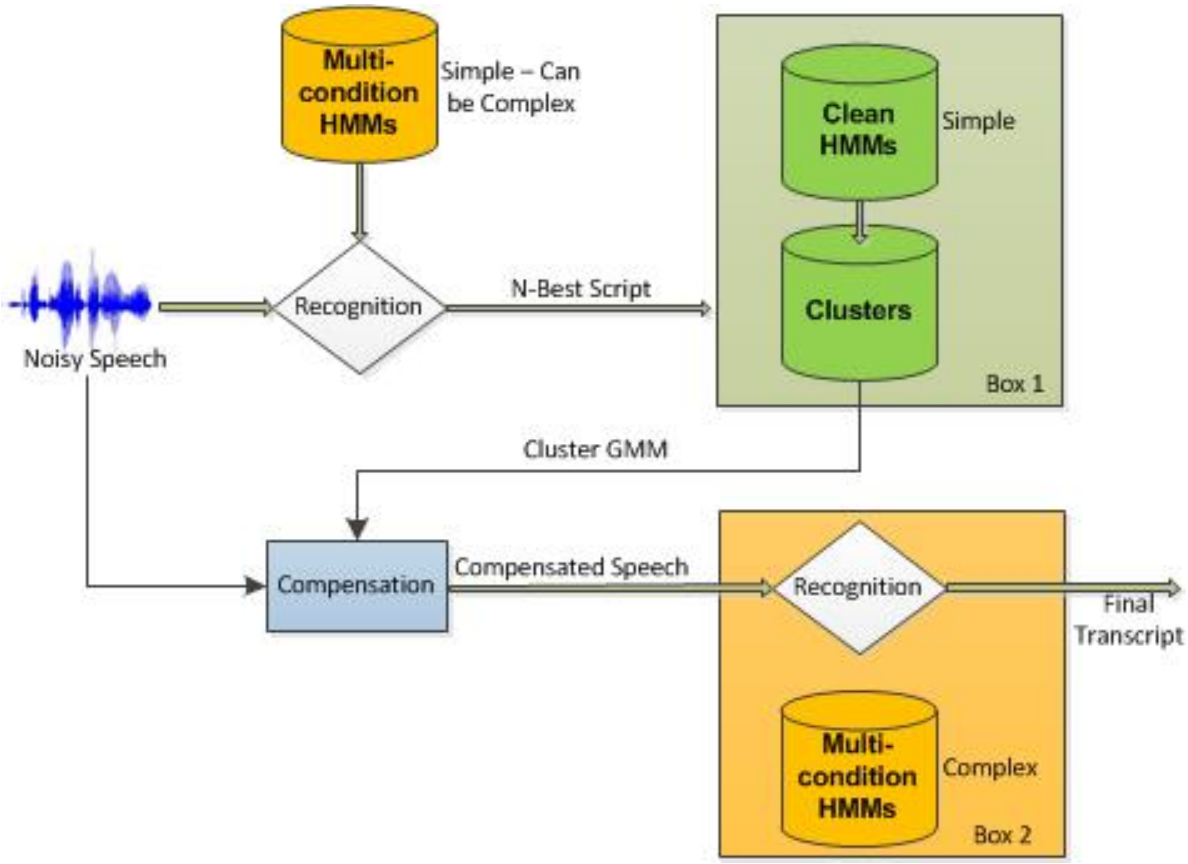

Figure 21: Simple vs Complex models

being evaluated. Complex models capture specific speech segments statistical information better.

It must be noted that if precise information about the speech segment being compensated is available, the compensation will improve. However, there will be a greater risk of selecting wrong statistics, i.e., the state might not represent the speech being compensated. It has been observed that the penalty incurred by wrong choice of cluster/state overwhelms the advantage gained from using complex and specific models and therefore, the simpler models work better in the compensation phase.

3.3.2 Experiments

To evaluate the proposed framework we experimented on the Aurora-2 connected digit task. We extracted features (39-elements with 13 MFCCs and their first and second time derivatives) from test speech as well as 23-channel filter-bank features thereby forming two streams. $One-best$ transcript was obtained from the MFCC stream using the multi-condition trained HMMs. PFC is then applied to the filter-bank stream (stream two). We chose two clusters, one based on $One-best$ and the other selected with Equation (72). The multi-condition clusters used in Equation (72) were from 23 channel fbank features so that the test features from stream two can be directly used to evaluate the likelihood of the observations. For results in these experiments, clusters were formed using method two, i.e., tracking the state-wise composition of each cluster. The number of clusters and particles were varied to evaluate the performance of the algorithm under different settings. From the compensated filter-bank features of stream two, we extracted 39-element MFCC features. Final recognition on these models was done using the retrained HMMs, i.e., multi-condition training data compensated in a similar fashion as described above.

3.3.2.1 Variable Number of Clusters

The results for a fixed number of particles (100) are shown in Table 3. The number of clusters was 20, 25 or 30. To set the specific number of clusters, HMM states were combined

Table 3: ASR accuracy - Variable number of clusters (100 particles)

Word Accuracy	**20 Clust.**	**25 Clust.**	**30 Clust.**	**Multi Cond.**	**Clean Cond.**
clean	99.11	99.11	99.11	98.5	99.11
20dB	97.76	98	97.93	97.66	97.21
15dB	97	97.14	96.69	96.8	92.36
10dB	95.21	95.41	93.88	95.32	75.14
5dB	89.48	89.59	87.08	89.14	42.42
0dB	70.16	70.38	68.84	64.75	22.57
-5dB	36.3	36.63	36.94	27.47	NA
0-20dB	89.92	90.1	88.88	88.73	65.94

and clustering was stopped when the specified number was reached. HMM sets for all purposes were 18 states, with each state represented by 3 Gaussian mixtures. For the 11-digit vocabulary, we had a total of approximately 180 states. In case of 20 clusters, we have a 9 to 1 reduction of information blocks to choose from for plugging in the PF scheme.

It is interesting to note that best results were obtained for 25 clusters. Increasing the number of clusters beyond 25 did not improve the accuracy. The larger the number of clusters, the more specific speech statistics each cluster contains. If the number of clusters is large, then each cluster encompasses more specific section of the speech statistics. Having more specific information in each cluster is good for better compensation and recognition because the particles can be placed more accurately. However, due to the large number of clusters to choose from, it is difficult to pick the correct cluster for generation of particles. More errors were made in the cluster selection process resulting in degradation in the overall performance. This is further illustrated in Figure 22. If the correct cluster is known, having large number of clusters and consequently more specific information per cluster will only improve the performance. The results are for 20, 25 and 30 clusters. In the known cluster case, one cluster is obtained using Equation (72) and the second cluster is the correct one. Correct cluster means the one that contains the state (obtained by doing

recognition on the clean version of the noisy utterance using clean HMMs) to which the observation actually belongs to. For the unknown cluster case, the clusters are obtained using Equation (72) and $One - best$. It can readily be observed from the known cluster case that if the choice of cluster is always correct, the recognition performance improves drastically. Error rate was reduced by 54%, 59% and 61.4% for 20, 25 and 30 clusters, respectively. Moreover, improvement faithfully follows the number of clusters used. This was also corroborated by the fact that if the cluster is specific down to the HMM state level, i.e., the exact HMM state sequence was assumed known and each state is a separate cluster (total of approximately 180 clusters), the error rate was reduced by as much as 67%.

3.3.2.2 Variable Number of Particles

For the results in Table 4, we fixed the number of clusters and varied the number of particles. As we increased the number of particles, the accuracy of the algorithm improves for set A and B combined i.e. for additive noise. The error reduction is 17% over the MC trained models. Using a large number of particles implies more samples were utilized to construct the predicted densities of the underlying clean speech features, which is now denser and thus better approximated. Thus, a gradual improvement in the recognition results was observed as the particles increased. In case of Set C, however, the performance was worse when more particles were used. This is so because the underlying distribution is different due to the distortions other than additive noise.

Table 4: ASR accuracy - Variable number of particles (25 clusters)

	Set A	**Set B**	**Set C**	**Average**
100 particles	90.02	91.03	89.26	90.1
500 particles	90.03	91.1	89.07	90.07
1000 particles	90.02	91.13	89.07	90.07
MC Trained	88.41	88.82	88.97	88.73
Clean Trained	64	67.46	65.39	65.73

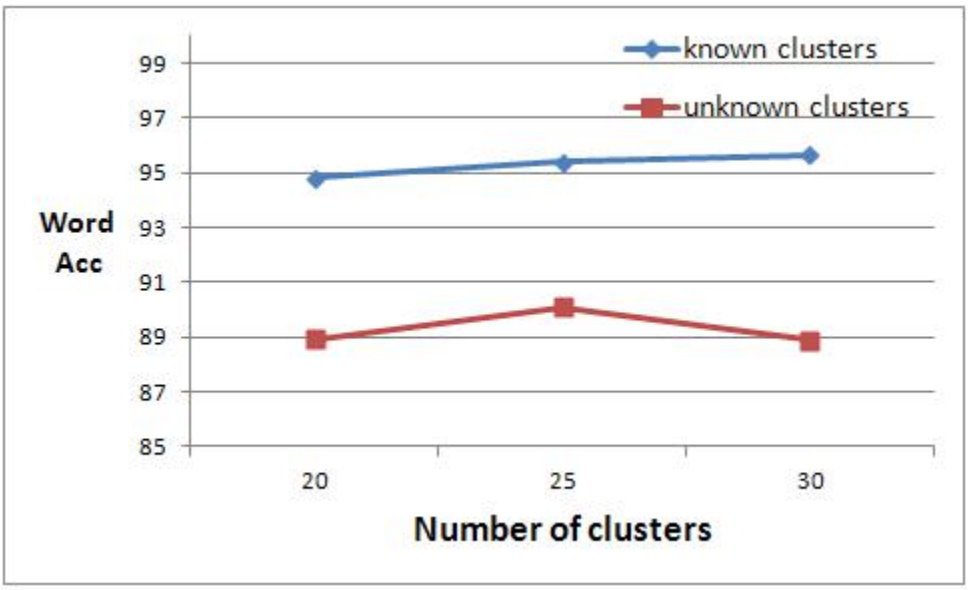

Figure 22: Word accuracy - Correct cluster known vs unknown

Table 5: Simple Vs complex backend performances for Sets A and B

Word Accuracy	**Simple Models**	**Complex Models**
clean	99.10	99.14
20dB	98.36	98.55
15dB	97.65	97.98
10dB	96	96.6
5dB	90.36	91.62
0dB	70.26	73.32
-5dB	36.01	38.96
0-20dB	90.53	91.61

3.3.2.3 Simple vs Complex Backend

Additionally, to explore the enhancement obtainable with higher complexity models at the backend, another experiment was carried out. Keeping the compensation settings the same, the number of GMMs for the backend HMMs was increased from 3 to 20. A clear improvement for all SNRs can be observed in Table 5

3.3.3 Summary

In this chapter, We propose a particle filter compensation approach to robust speech recognition, and show that a tight coupling and sharing of information between HMMs and particle filters has a strong potential to improve recognition performance in adverse environments. An integrated PF-HMM approach is presented, where we incorporate statistical information available from the HMMs to make up for the lack of suitable state transition model. This enables us to use the PF framework to compensate noisy speech signals. We further developed a scheme to merge statistically similar information in HMM states to enable us to find the right section of HMMs to dynamically plug in the particle filter algorithm. Results show that if we use information from HMMs that match specifically well with the section of the speech being compensated, significant error reduction is possible compared to multi-condition HMMs.

It is noted that we need an accurate alignment of the state and mixture sequences used for compensation with particle filters and the actual HMM state sequences that describes the underlying clean speech features. Although we have observed an improved performance in the current particle filter compensation implementation, there is still a considerable performance gap between the oracle setup with correct side information and whats achievable in this study with the missing side information estimated from the noisy speech. We anticipate that the current performance gap can be narrowed when more advanced algorithms are explored to obtain better estimates of the missing side information needed to fully utilize the power of particle filter compensation.

CHAPTER 4

JOINT ESTIMATION OF SPEECH AND NOISE FEATURES

Noise parameters play a crucial role in the computation of the particle weights. While a good selection of the cluster allows for a better placement of the samples, the availability of precise noise information improves the weight estimates of the samples. The result is a more faithful probabilistic representation of the underlying clean speech, and consequently, an improved estimation of the clean-speech signal. In this chapter, two methods for the estimation of the noise parameters are proposed. The first is based on a particle filter that runs in parallel to the PFC [13]. The second method is based on a MCMC approach [14].

4.1 Noise Tracking Using PF

Noise parameters are difficult to track because the state-transition model for the noise signal is difficult to obtain. Noise could be emanating from a wide variety of sources, and in most cases, does not have a well-defined structure. To overcome this problem, an approach to track the noise statistics is proposed that uses simplistic models for state transition. The advantage of using simple state-transition models is that little or no assumptions are made about the structure of the signal being evaluated. In the proposed tracking strategy, the clean-speech-statistical information, obtained from the PFC algorithm, will also be put to use.

4.1.1 Noise Modeling

For the purpose of noise tracking, the noise signal is treated as the hidden state of the system, and the speech signal is treated as the corrupting signal. The observed noisy signal is the observation. Two models are used to represent the state-transition process of the noise. First, a random-walk process, which can be described by the following two equations, is used:

$$N_t = N_{t-1} + W_t \tag{73}$$

$$W_t \sim \mathcal{N}(0, \Sigma_W), \tag{74}$$

where N_t is the log-spectral vector of noise at time t, and W_t is the driving noise for the state-transition process. The random-walk process translates into a simple propagation scheme:

$$p(n_t|n_{t-1}) \sim \mathcal{N}(n_t; n_{t-1}, \Sigma_W). \tag{75}$$

The propagation steps are summarized in Figure 23. A particle that has a greater weight will split into child particles in proportion to its weight. These particles are further displaced by adding a random vector w that is sampled from the distribution in Equation (74).

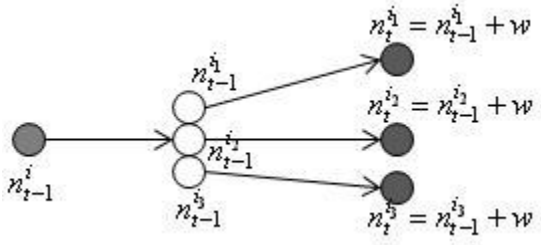

Figure 23: Particle propagation for the random-walk process

The second choice for the state-transition model is the autoregressive (AR) model. In this case, the particle for the next stage is regressed on the corresponding sample in the current stage:

$$p(n_t|n_{t-1}) \sim \mathcal{N}(n_t; A_t n_{t-1}, \phi_\varepsilon), \tag{76}$$

which is derived from the first order AR model:

$$n_t = A_t n_{t-1} + \varepsilon_t. \tag{77}$$

Here, ε_t represents the Gaussian-excitation process. Its mean is *zero*, and its variance is ϕ_ε, which can be learned from the available noise samples. The first order AR model in Equation (77) does a fair job in capturing the spectral dynamics of various noises typically encountered in ASR problems. Increasing the order does not increase the accuracy of the

model [8]. The weights of the particles propagated using the random-walk process or the AR model are computed in the same manner. The observation model is the same as the one described in Equation (55). Instead of $p(y_t|x_t)$ required for the speech particles, the posterior $p(y_t|n_t)$ is needed for computing the weights for the noise-particle samples. The posterior is derived from

$$p(Y_t < y_t|n_t) = p(X_t + \log(1 + \exp(n_t - X_t)) < y_t|n_t), \tag{78}$$

where X_t represents the clean speech random vector. It is considered to be a GMM, which is imported from the PFC scheme. It is important to emphasize here that the posterior for the noise particles cannot be obtained without some information about the clean-speech statistics. Consequently, help from the speech tracking part of the PFC algorithm is sought, and the best cluster at time t, C_t, is used to obtain the clean-speech statistics.

Looking at Equation (78), it is often difficult to resolve $\log(1 + \exp(n_t - X_t))$ in terms of X_t, a step required to obtain the posterior. To overcome this problem, a Taylor series approximation of the term is taken around $\mu_{x,t}$, which is the mean of the clean-speech distribution. Thereupon, the term in consideration can be rewritten as $\log(1+\exp(n_t-\mu_{x,t}))$, and the the posterior can be written

$$\begin{aligned} p(y_t|n_t) = \frac{1}{\sqrt{2\pi\sigma_{x,t}^2}} \exp[-\frac{1}{2\sigma_{x,t}^2}(y_t \\ - \log(1 + \exp(n_t - \mu_{x,t})) - \mu_{x,t})^2]. \end{aligned} \tag{79}$$

The clean-speech statistics are represented by a cluster, distribution of which is a Gaussian mixture. Posterior can accordingly be simplified to

$$p(y_t|n_t) = \sum_{k=1}^{K} m_k \mathcal{N}(u_t; \mu_{x,k,t}, \sigma_{x,k,t}^2), \tag{80}$$

where $u_t = y_t - \log(1 + \exp(n_t - \mu_{x,k,t})) - \mu_{x,t}$.

4.1.2 Implementation

The overall implementation scheme is shown in Figure 24. The compensation and the recognition steps are carried out separately. Since the compensation is being done at the

front end only, a different set of models can be used for each task. For the compensation phase, it is better to have simple models. Since the mixtures are merged into clusters, using complex models do not provide any advantage over simple models. For the recognition phase, however, complex models perform better as they have a greater ability to capture subtle variations of the speech signal to be decoded.

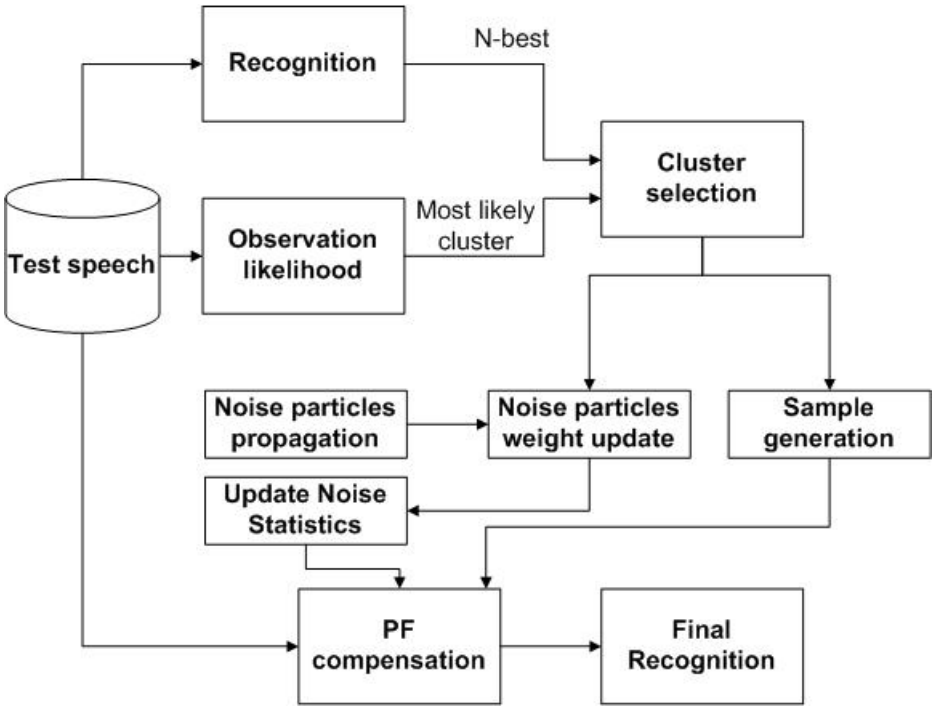

Figure 24: Complete recognition process with noise tracking

4.1.3 Experiments

To evaluate the proposed framework, experiments were conducted on the Aurora-2 (subway noise) connected-digit task. MFCC features (39 elements with 13 MFCCs and their first and second time derivatives) as well as 23-channel-filter-bank features are extracted from the test speech, thereby forming two streams. One-best transcript was obtained from the MFCC stream using the multi-condition-trained HMMs. PFC is then applied to the filter-bank stream. The multi-condition clusters, used for choosing the best cluster, were obtained from the 23-channel-fbank features so that the test features from stream two could

be directly used to evaluate the likelihood of the observations. The noise was tracked using both the random-walk process and the AR model. Approximately 20 particles were used for the noise tracking, whereas 100 particles were used for the clean-speech tracking. From the compensated filter-bank features of stream two, a new set of 39-element-MFCC features were extracted. Final recognition on these features was done using the retrained HMMs that are trained with the multi-condition-training data, compensated in a similar fashion to the compensation process described above.

If the samples are generated using the best possible state and mixture sequence (artificially obtained here), a very significant 67% digit-error reduction was attained over multi-conditioned trained models as shown in column *two* (assuming all state sequences are known in this oracle case) of Table 6. When the cluster information is estimated from the testing environment in actual operating conditions (column *three*) as opposed to the oracle case, some improvement over the state-of-the-art multi-conditioned-trained models (column $five$) was observed. However, this performance was inferior to the oracle case.

Table 6: ASR accuracy - Performance comparison with noise tracking

Word Accuracy	**Speech Tracking I**	**Speech Tracking II**	**Noise Tracking**	**Multi Condition**
clean	99.14	99.14	99.14	98.46
20dB	97.94	97.91	97.91	97.79
15dB	97.67	97.21	97.18	97.11
10dB	97.24	95.36	95.06	95.52
5dB	95.46	90.08	90.82	90.3
0dB	93.28	75.16	74.95	69.85
-5dB	92.26	41.51	39.36	28.98
0-20dB	96.37	91.14	91.18	90.11

The additional tracking of the noise signal (column $four$) does show a slight improvement in accuracy over the tracking of the speech signal only (column *three*). The performance of noise tracking can also be observed in Figure 25. For better visualization, only

one of the 23-fbank channels, under 5dB noise level, is shown. The actual noise is shown as a black-solid line and the estimated noise obtained using PF is shown as a red-dashed line. Although the tracker is able to capture the overall contour of the noise curvature at most parts, it fails to follow the smaller variations. This anomaly is because of the simplicity of the models being used for the noise-particle propagation. As a result, the particle samples may not be in the optimal locations, leading to an unfaithful construction of the noise signal. Better models that can incorporate information from the observed noisy signal could be used to improve the propagation and the overall performance.

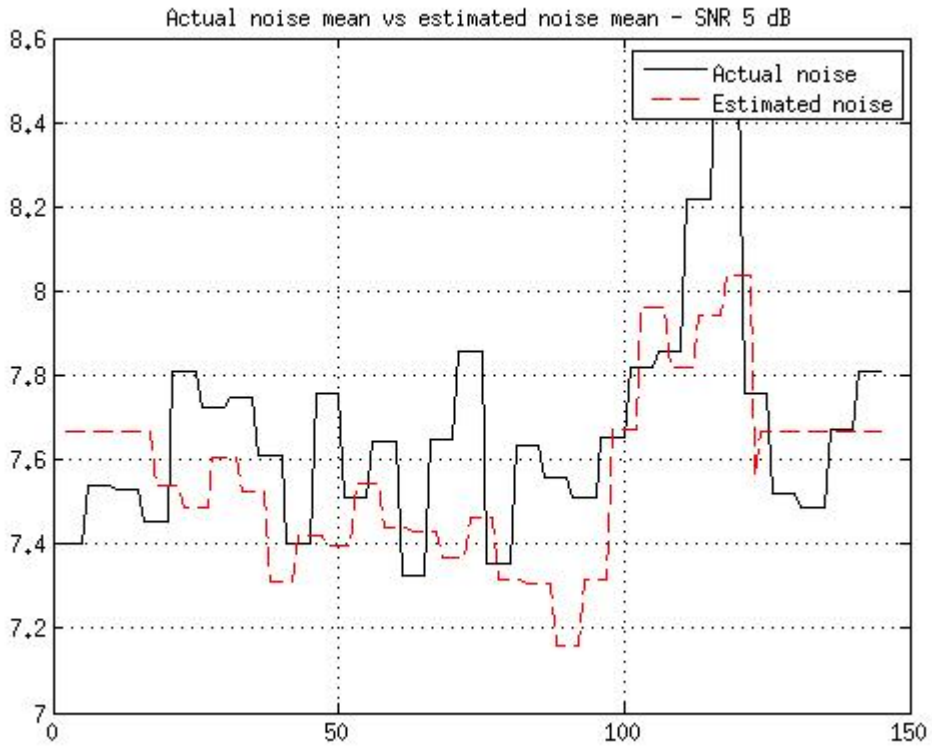

Figure 25: Actual vs. estiamted noise at 5dB noise level

4.2 An MCMC Approach to Noise Estimation

The MCMC approach allows for the estimation of the joint distribution $p(n_t, x_t|y_t)$, from which the marginal distribution $p(x_t|y_t)$ can be extracted. The straightforward implementation of MCMC described in Section 2.4.3 requires that the samples must be directly obtainable from

$$\begin{aligned} &p(n_t|x_t, y_t) \\ &p(x_t|n_t, y_t). \end{aligned} \tag{81}$$

However, it is not possible to generate samples from these distributions. To overcome this problem, the more general Metropolis algorithm can be exploited. Specifically, a combination of the importance sampling and the Metropolis-sampling algorithm is put to use. Compared to the jumping distribution used in the Metropolis-Hastings algorithm [75], where samples are rejected with probability r, the importance sampling scheme is more efficient. Another motivation for using the importance sampling is the availability of the required framework.

Now, $p(x_t^{(s)}|n_t^{(s-1)}, y_t)$ can be evaluated using

$$p(y_t|x_t^{(s)}) = p(u_t)\frac{e^{y_t - x_t^{(s)}}}{e^{y_t - x_t^{(s)}} - 1}, \tag{82}$$

where $x_t^{(s)}$ represents the s^{th}clean speech sample at time t, the noise density is given by $\mathcal{N}(n_t^{(s-1)}, \sigma_n)$, and $u_t = \log(e^{y_t - x_t^{(s)}} - 1) + x_t$ with $F(u_t)$ being the Gaussian cumulative function with mean $n_t^{(s-1)}$ and variance σ_n^2. Similarly, to evaluate $p(n_t^{(s)}|x_t^{(s-1)}, y_t)$, the following relation [13] is used

$$\begin{aligned} p(y_t|n_t) = \frac{1}{\sqrt{2\pi\sigma_{x,t}^2}} \exp[-\frac{1}{2\sigma_{x,t}^2}(y_t \\ - \log(1 + \exp(n_t^{(s)} - x_t^{(s-1)})) - x_t^{(s-1)})^2]. \end{aligned} \tag{83}$$

The conditional distributions cover only a part of the distribution approximation by providing a mechanism to evaluate the weights of the samples. Prior to evaluating the weights, the samples have to be available at the right locations. To generate the speech

samples, the statistics available from the HMMs are used. It is important to emphasize here that the sample generation for x_t is not dependent on the n_t samples. Instead, it is the computation of the weights for x_t that is conditioned on the noise samples. On the contrary, the location of the samples and the weights for the noise samples are both conditioned on the clean-speech samples.

For each frame t, the algorithm proceeds as follows:

1. Generate sample x_t^s using HMMs.
2. Compute the weight for x_t^s using n_t^{s-1} in Equation (82).
3. Generate the sample n_t^s from x_t^{s-1} using Equation (55).
4. Compute the weight for n_t^s using x_t^{s-1} Equation in (83).
5. Repeat if $s < N_s$,

where N_s is the desired number of samples. Once the point density of the clean-speech features is available, the compensated features can be obtained using:

$$x_t = \sum_{i=1}^{N_s} w_t^i x_t^i. \tag{84}$$

4.2.1 Comparison of PFC and MCMC Approaches

The comparison of the PFC approach for speech compensation and the MCMC approach is laid out in Figure 26. The dashed arrows indicate the dependencies in the sample generation for MCMC, whereas the thick arrows indicate the dependencies for PFC. The cluster selection mechanism is the same for the PFC and the MCMC methods. The main difference lies in the distribution of the speech and the noise signals. While the samples for the noise are propagated using the AR model or the random-walk process in PFC, the noise samples in MCMC are generated directly from the speech samples. The statistics from the cluster, directly and indirectly, influence the location of the speech and the noise samples respectively. The observation then plays a critical role and adds the information to compute the

weights for both samples. Moreover, unlike the PFC method, the noise samples are directly used in the weight computation of the speech samples.

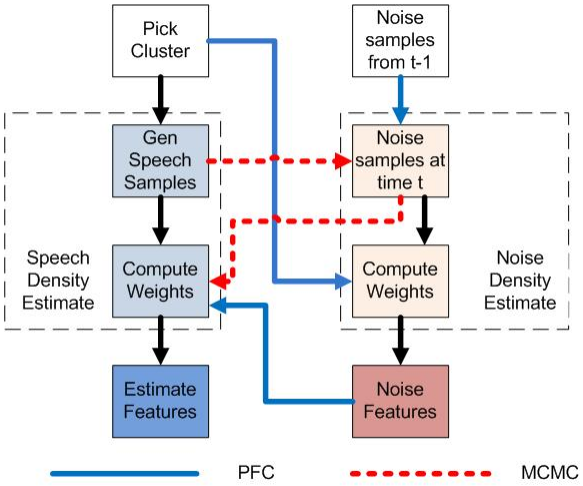

Figure 26: Comparison of PFC and MCMC

4.2.2 Experiments

To evaluate the proposed framework, experiments were conducted on the Aurora-2 connected-digit task. The setup is similar to the one used for the PF based noise-tracking experiments. Compensation is carried out in the 23-channel-fbank-feature domain. From the test speech, 39-element MFCC features (13 MFCCs and their first and second time derivatives) as well as 23-channel-filter-bank features are extracted to form the two streams described previously. The $One - best$ transcript, used for cluster selection, was determined by evaluating the MFCC stream with the MC models. The speech samples are generated using the selected clusters derived. Using the cluster instead of the noise samples for generating speech samples helps in preventing the sticking problem, which is a vulnerability of the MCMC algorithms. If the speech sample is generated based on the noise sample and vice versa, then from Equation (55), the samples would be concentrated in a very small region. This agglomeration will happen because after y_t is observed, knowing the value of n_t or x_t gives us precise information about the other. The sequence of samples generated for the speech

and the noise signals is shown in Figure 27. The dependence of the noise samples is observable. The noise samples are related to the speech samples through the observation. Note that for higher value of the speech sample, the value for noise sample is smaller and vice versa.

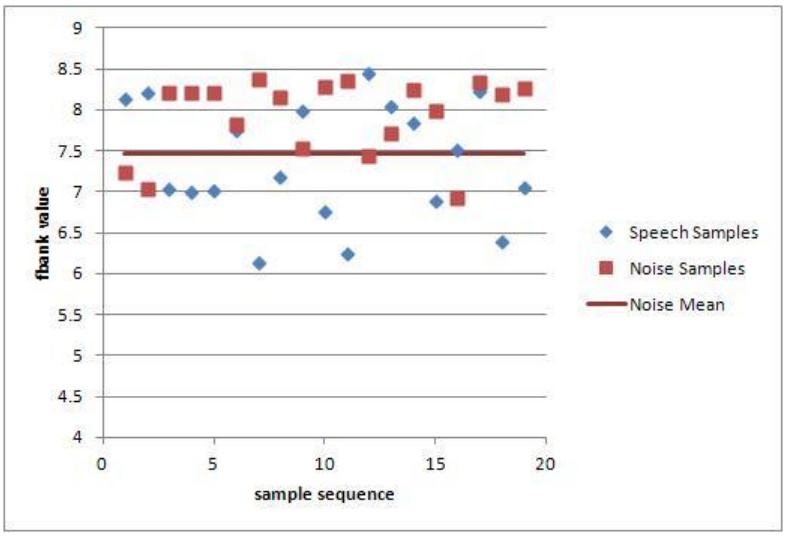

Figure 27: Sample sequence for MCMC

The improvement obtained with MCMC in terms of the error reduction in word accuracy over multi-condition training and PFC is given in Table 7. The recognition performance for the MCMC algorithm improves for all noise levels except for lower SNRs of 0dB and -5dB. Since the noise signal is dominant compared to the speech signal at these SNRs, the assumption that the noise samples can be well placed by indirectly using the speech statistics does not seem to hold, resulting in poor noise estimates. Consequently, no improvement in the recognition performance is observed. Overall, an error reduction of 12.87% is achieved over multi-condition training compared to the 12.16% obtained with PFC.

To improve the stability of the noise parameters estimate, the noise information from multiple frames, $t - N_p/2$ to $t + N_p/2$ (N_p frames), is combined together and used to recompute the weight of the x_t samples. The weight for the speech samples is recomputed

Table 7: Error reduction over MC

ER	20dB	15dB	10dB	5dB	0-20dB
MCMC	20.1%	20.3%	8.9%	8.1%	12.87%
PFC	14.5%	10.6%	1.9%	4.1%	12.16%

after getting the noise estimate from N_p frames. Combining the noise information from multiple frames improves the performance of the recognizer. While the *Word Accuracy* for $N_p = 30$ is 90.2%, the corresponding performance for $N_p = 60$ and $N_p = 15$ is 90.1% and 90.06% respectively. The performance is inferior if N_p is either smaller or larger than the value 30. If N_p is large, the smaller variations in the noise estimate are averaged out and the performance is comparable to the case where the noise statistics are considered to be non-varying. On the other hand, the reduction in performance for N_p smaller than 30 is because of the erroneous estimate of the noise in the smaller intervals.

The noise estimation for a particular fbank channel is depicted in Figure 28. The errors in the noise estimate cause corresponding fluctuations in the compensated speech estimate. These fluctuations are undesirable from the machine learning perspective. For a better recognition performance, it is desirable that the behavior is consistent, not only within training data, but also between the training and the testing data. The random variations caused by the wrong noise estimate introduces discrepancies in both training and testing data. Increasing the number of frames, however, averages out these variations and improves the recognition performance.

The performance achieved with MCMC is also superior to the improvement obtained from tracking the speech signal using a particle filter that is running parallel to the PFC algorithm. The comparison for subway noise is given in Table 8. Both approaches still fall short of the case where the exact noise information (average over 30 frames) is available and therefore, the margin for improvement in noise estimation is still present.

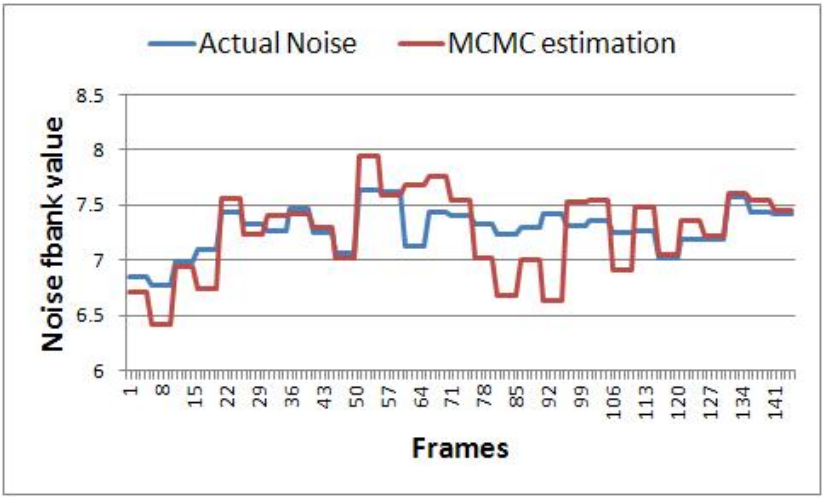

Figure 28: Actual vs. estiamted noise for MCMC at 10dB noise level

Table 8: Noise estimation comparison

Word Accuracy	**Noise Tracking MCMC**	**Noise Tracking PF**	**Noise Known**
clean	99.14	99.14	99.14
20dB	98.25	97.91	98.19
15dB	97.51	97.18	97.14
10dB	95.03	95.06	95.3
5dB	90.76	90.82	90.39
0dB	74.7	74.95	76.02
0-20dB	91.25	91.18	91.41

4.3 Summary

In this chapter, a framework was introduced to obtain improved estimates of the noise parameters, which play an important role in the PFC algorithm. A particle-filter algorithm is developed to track the noise signal. The AR model and the random-walk process were used for the state transitions of the noise signal. Both are simple models and are used as a compromise because of a lack of structure in the noise signals typically encountered in ASR. The impact of improved statistics on the PFC algorithm are also presented. The estimation of the noise signal is good enough, but the improvement in the recognition

performance of the overall scheme is only marginal.

An MCMC based approach for the joint estimation of speech and noise is also proposed. Results show that the MCMC based approach provides better performance in the overall recognition process when compared with the PF approach. MCMC performs better because it utilizes more information from the available speech statistics. Instead of relying on a point estimate of the clean-speech approximation like the PF algorithm, MCMC exploits the full information in the clean-speech distribution generated by the PFC algorithm.

The performance of both the PF algorithm and the MCMC algorithm are compared with the case where an accurate noise-parameter information is available. In general, a better noise estimate improves the recognition performance for all SNRs except the very low ones.

CHAPTER 5

PFC FOR LVCSR

In this chapter, we extend the PFC algorithm to Aurora-4 corpus, which is a large vocabulary continuous speech recognition task. Obtaining an accurately aligned state and mixture sequence of HMMs (side information) that describe the underlying clean speech features being estimated is a challenging task for sub-word based LVCSR because the total number of triphone models involved can be very large. By using separate sets of HMMs for recognition and compensation, we can simplify the models used for PFC to a great extent and thus facilitate the estimation of the side information offered in the state and mixture sequences.

5.1 Overview of PFC

Before we get into the details of the implementation of PFC for LVCSR, the important equations of PFC are reiterated here:

1. Posterior density of speech, based on the current observation is represented by a finite number of set points,

$$p(x_t|y_{0:\,t}) = \sum_{i=1}^{N_s} w_t^i \delta(x_t - x_t^i), \tag{85}$$

   where x_i^t for $i = 1, ..., N_s$ are the support points of PF.

2. The weight vector, w_t^i,associated with the support points, approximates the posterior density and are determined based on the concept of *importance sampling* computed with:

$$w_i^t = w_{t-1}^i \frac{p(y_t|x_t^i)p(x_t^i|x_{t-1}^i)}{q(x_t^i|x_{t-1}^i, y_t)}. \tag{86}$$

3. PFC is done in the spectral domain. Given additive noise with no channel effects we can evaluate $p(y|x)$ using

$$p(y|x) = p(u)\frac{e^{y-x}}{e^{y-x} - 1}, \tag{87}$$

where x represents clean speech and n represents the noise with density $\mathcal{N}(\mu_n, \sigma_n)$ and $u = \log(e^{y-x} - 1) + x$ with $F(u)$ being the Gaussian cumulative function with mean μ_n and variance σ_n^2.

4. The density $q(x_t^i | x_{t-1}^i, y_t)$ is used to generate the particle samples. The distribution is obtained by clustering HMMs:

$$q(x_t^i | x_{t-1}^i, y_t) \sim \sum_{k=1}^{K} m_{k,C_t} \mathcal{N}(\mu_{k,C_t}, \Sigma_{k,C_t}), \tag{88}$$

where m_{k,C_t}, μ_{k,C_t} and Σ_{k,C_t} are the weight, mean and variance of the mixture k in cluster C_t.

5. After generating samples, we estimate the compensated features using

$$x_t = \sum_{i=1}^{N_s} w_t^i x_t^i. \tag{89}$$

5.2 Implementation of PFC

In the PFC algorithm, four HMM sets are used in various roles. The roles of these models are explained in this section. The most important aspect of the PFC algorithm, aside from the observation model, is the placement of the samples. Clean fbank HMM set (hereafter known as set 1) is used to generate the samples because clean speech is being estimated from these samples, and clean HMMs provide the distributions that best represent the clean speech statistics. These models are derived from the fbank features because compensation is done in the fbank domain.

It is critical that the correct model from the HMM set is chosen for the treatment of a particular frame so that the samples can be generated from a distribution that precisely represents the underlying speech for that frame. The structure of the set 1 HMMs should therefore be such that it is easy to pick the most suitable model at each frame. As is described in section 3.3.1, a large number of models makes this selection harder. For LVCSR systems, subword acoustic models are a popular choice and triphone representation

achieves the best recognition performance. However, in the case of PFC, the large number of models required in the triphone representation make the model selection problem even harder.

Monophone models provide a convenient solution to the problem. Although the accuracy of the statistical representation is compromised for the case of monophone models compared to the triphone model, yet the number of statistical units is drastically reduced by a ratio of approximately $1 : 20$. By further clustering the monophone models into 10 or less statistical units, the composition of the set is reduced to about $1 : 250$ when compared to the triphone models. This procedure simplifies the cluster selection process to a great extent, but the task of estimating the appropriate cluster from the observerd noisy speech is another complication. The set 1 is unsuitable for the task because:

1. It is built from fbank features, which have inferior discrimination capability compared to MFCC features.

2. Clean models peform poorly in the recognition task when applied to noisy speech.

3. Monophone models cannot compete with the triphone models in the recognition task.

To overcome this complication, a second set of HMMs (set 2) is deployed to obtain speech information from the noisy signal. This set is derived with the aim of getting optimum recognition performance. Hence, the HMMs in set 2 are triphone models built using the multi-condition MFCC features.

5.2.1 Alignment of Set 1 and Set 2

The HMMs in set 2 are used to select the appropriate cluster from HMMs in set 1, therefore, a good alignment between the two sets is essential to obtain good performance with PFC algorithm. The two sets, however, use different features, structures (one is made up of monophone while the other of triphone models) and data (one uses clean and the other uses noisy speech). Consequently, the two sets can be severely misaligned. To overcome

this problem, the clean MFCC HMMs (set 3) are used as the source and both set 1 and set 2 are derived from it. The technique for this alignment procedure is explained in Figure 29. Training HMMs in set 1 has 2 steps. Step 1 computes forward and backward probabilities using clean MFCC monophone HMMs on clean MFCC features. Step 2 estimates parameters of fbank monophone HMMs using the statistics information from Step 1, together with clean fbank features. This is known as single-pass retraining.

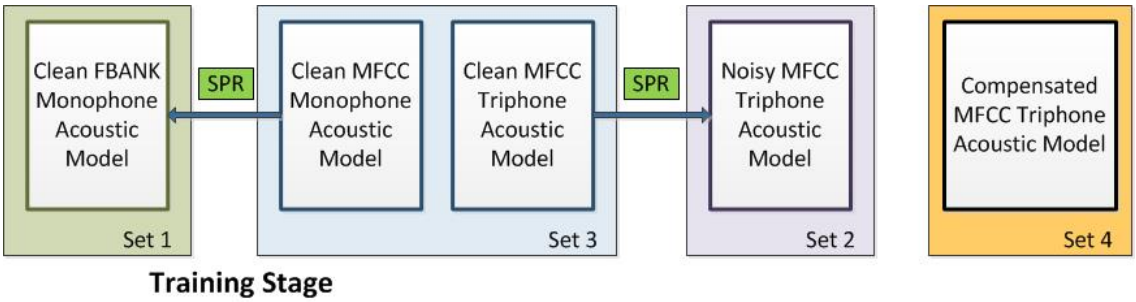

Figure 29: A block diagram illustrates training process using the single-pass retraining

In this way, the state/phone alignment (i.e., the posterior component probabilities) used to estimate parameters of monophone fbank HMMs is the same as one generated by using the monophone MFCC HMMs. Therefore, same component label of two states will model the same sound but in two different feature domains.

Training HMMs in set 2 is similar. Step 1 computes forward and backward probabilities using triphone HMMs on clean MFCC features. Step 2 estimates HMM parameters using the statistics from Step 1 along with noisy MFCC features.

Since all HMM parameters of sets 1 and 2 are estimated based on state alignment computed from clean MFCC HMMs, a state mapping between the two sets can be obtained by just using the same component labels.

5.2.2 Models for Compensated Features

As described in Section 3.3.1, the HMMs (set 4) used in the final recognition of the compensated data is isolated from the compensation process. Therefore, set 4 is independent of sets 1,2 and 3. Set 4 is trained using MC training data that has been compensated like

we would process the test data in actual scenario. Since there are no constraints on these models, their complexity can be increased to the optimum level needed to obtain the best possible recognition performance.

5.3 LVCSR Speech Corpus for the Experiments

The Aurora-4 corpus is an artificially noise-corrupted speech data. The vocabulary size is around 5000 words. The corpus involves 6 real life noises (car, babble, restaurant, airport, street, train) and clean utterances recorded with gender balance. The noisy data are generated by artificially adding the recorded noises to the clean utterances. The details of the corpus are given in Table 9

Table 9: The Aurora-4 speech corpus

Nature of task	Large vocabulary continuous English speech
Sampling frequency	16000Hz
Average utterance length	7.6 seconds
Training data	83 speakers and gender-balanced. Clean training data: 7138 clean utterances, Multi-condition data: 7138 multi-condition utterances (different microphones and 6 different noise types)
Test data	14 test cases or environments Case 1: Clean test. Case 2-7 (6 noises): car, babble, restaurant, street, airport, train. Case 8: Microphone mismatch. Case 9-14: case 2-7 + microphone mismatch.
Recognizer	HMM Toolkit
Language model	Bigram model
Acoustic model	3 states triphone model, 1700 tied states and 16 Gaussian mixtures per state
Test parameters	0 word insertion penalty,16 language model weight and 250 pruning threshold

5.4 Experiments

This section presents the experiments of PFC on Aurora-4 corpus, and analysis of the experimental results. We focus on training simple and complex acoustic models used in PFC, an oracle experiment to estimate upper bound of the method and an actual experiment to evaluate performance of the system.

5.4.1 General Configurations

HMM tool kit (HTK) is used to extract speech features and train acoustic model. Log mel filter bank (fbank) coefficients (23 coefficients) are extracted from 16K sample rate speech signal and enhanced by PFC method. Mel-frequency cepstral coefficients (13 coefficients) and their first and second differential features are then extracted from compensated fbank and used as speech features for speech recognizer. Mean normalization is also applied to reduce the channel mismatch. A bigram language model is used with scale factor set to 15.

The four acoustic models are trained as described in the previous section. Clean fbank HMMs (set 1) has 120 states. In this work, the complexity of set 2, 3 and 4 are the same and have 1594 tied-states with 16 Gaussian mixtures per state.

In test phase, we are interested in additive background noises. Six noisy test sets (car, babble, restaurant, street, airport and train noises) without channel mismatch are used to evaluate the PFC performance.

5.4.2 Oracle Experiments

To estimate the potential of PFC, we first build an oracle experiment with good mapping accuracy. In this experiment, we utilize the stereo data in Aurora-4 to generate oracle state sequence which is clean state sequence and used as noisy state sequence and thus the mapping is exact. In this way, we can focus on optimizing particle sampling and evaluate the upper bound of the PFC method.

We generate mono-phone state sequence using clean fbank monophone HMMs (set 1) on clean version of noisy data. The state sequence obtained is considered as the oracle

state sequence because, albeit not perfect, it is the best estimate used for generation of clean feature samples.

The prior information used by oracle experiment is illustrated in Figure 30. The oracle state sequence is used together with the clean fbank monophone HMMs. Un-clustered clean fbank monophone HMMs have 120 states and are denoted by "set 1-120". We group 120 states into 10 (or 5, 3, 2, 1) clusters as discussed in the previous sections and denote as "set 1-10" (or 1-5, 1-3, 1-2, 1-1 respectively).

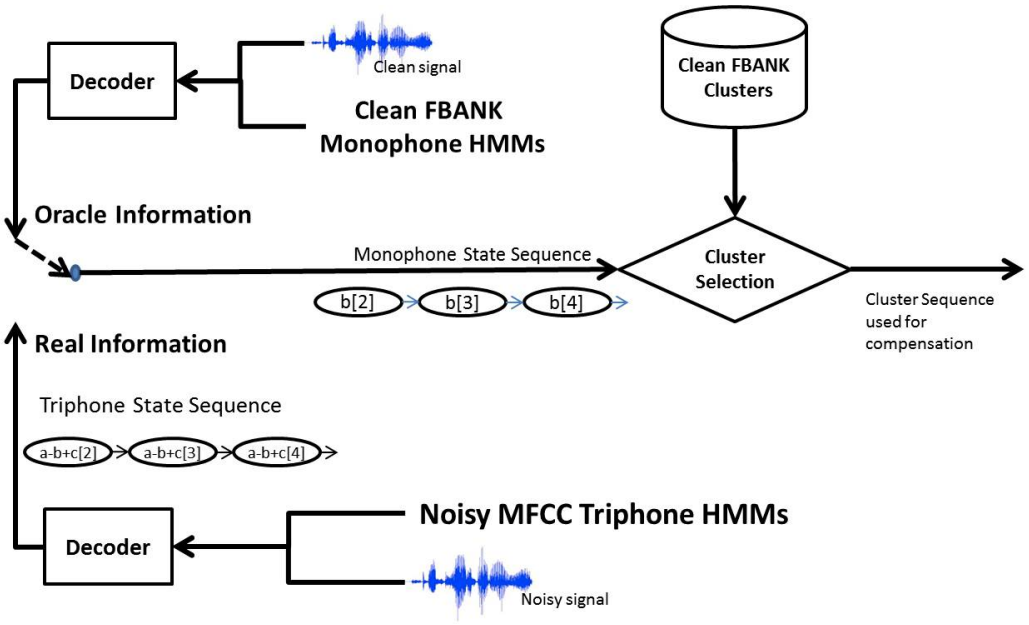

Figure 30: A block diagram illustrating the actual vs the oracle experiment setup for LVCSR

The word accuracies of these versions of set 1 are showed in table 10. In the study, 120 is the largest cluster count used. Although, the count can be increased to 1594, which is the starting number of states if clustering directly from triphone acoustic model, and it will most likely improve the performance beyond the best figure of 85.6% because the statistical information is more precise. However, it hasn't been explored due to the fact that obtaining good side information in case of such a large number of clusters will be nearly impossible in real scenarios.

On the other side, 1-cluster is the smallest cluster count that can be used. Apart from the fact that the performance for this case improves over the baseline multi-condition training, the setup has it's own advantages. First, the estimation of side information is not required, making the compensation process very efficient. Secondly, with 1-cluster, no errors can be made in the estimation of side information and therefore, the actual performance and the oracle performances are the same.

Table 10: ASR accuracy comparisons for Aurora-4

No. Clus.	2	3	4	5	6	7	Avg
-	87.4	81.5	75.6	78.4	80.9	75.4	79.9
1	86.6	82.6	76.2	79.3	80.7	76.2	80.3
2	87.2	83.9	78.2	80.4	82.1	77.1	81.5
3	87.3	84.5	78.9	81.3	82.3	79.0	82.2
5	88.1	84.9	81.2	83.0	84.0	82.1	83.9
10	88.2	85.8	81.3	83.5	83.7	81.7	84.0
120	88.8	86.3	83.4	84.4	87.1	83.8	85.6

5.4.3 Actual Experiments

In this work, we examine a straightforward cluster selection approach. The left and right contexts of triphone state label are discarded to obtain a monophone state sequence. The rest of the compensation process is the same as in the oracle experiment. A comparison of the oracle and the real performance is given in Figure 31.

When we use more clusters, each cluster will have more specific statistical information about the speech frame being compensated. Due to this, oracle 5-cluster is better than 1-cluster and oracle 120-cluster is the best performance we have. When statistical information is better, the samples generated from it are lying at locations that represent the current underlying clean speech distribution more precisely and consequently, the estimate of clean speech is better. Therefore, the greater the number of clusters, the better the performance. However, in the actual scenario, we make mistakes in picking/choosing the cluster for generating samples. Whenever an error is made, a penalty is incurred because

compensation could be wrong and subsequent recognition could be incorrect. Obviously, when we are using more clusters, there is a greater chance of picking a wrong cluster. For example when we are using 1-cluster, error cannot be made at all, but when we are using 5 clusters, there are 4 wrong clusters that can be selected by mistake. Similarly, when using 3-clusters, the mistakes are less than the 5-cluster because the choices are less. To improve the performance, we need to reduce the errors in cluster selection.

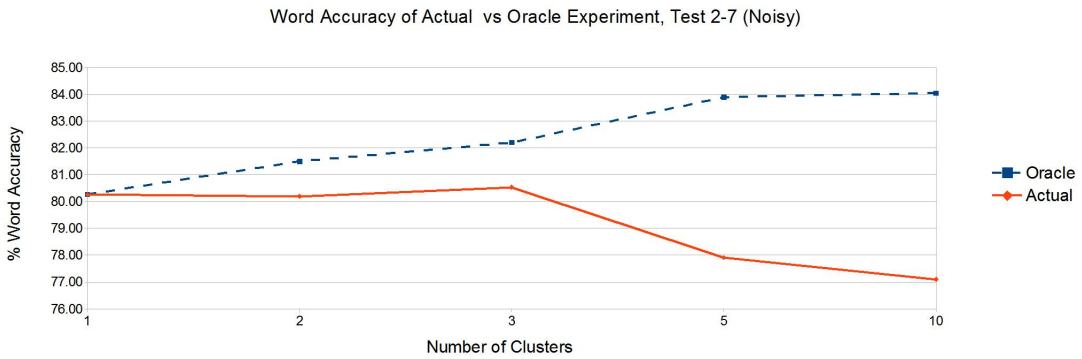

Figure 31: Real vs Oracle PFC performance for LVCSR

Table 11: ASR accuracy comparisons for Aurora-4 with PFC and MVN combined

Feat.	2	3	4	5	6	7
CMN	87.4	81.5	75.6	78.4	80.9	75.4
PFC3+.	87.3	82.8	76.6	79.3	79.5	77.7
%Rel	-0.3%	**7.0%**	3.9%	4.3%	-7.7%	**9.4%**
MVN	88.2	81.9	79.9	81.4	82.6	79.2
PFC3+.	89.0	82.9	79.6	82.4	82.2	79.5
%Rel	**6.6%**	**5.5%**	-1.5%	**5.7%**	-2.4%	1.4%

To further understand PFC, we investigated effects of variance normalization on the compensated features. In particular, we evaluated enhanced features by first processing 3-cluster PFC and then normalizing mean and variance of cepstral features. Details of word accuracies are shown in Table 11. Clearly PFC gained improvements on babble, street and train noises for both CMN and MVN features, but it was otherwise for car, restaurant and airport noises. PFC could have large fluctuations in the compensated fbank features when

the SNR is low and there are un-smoothed fluctuations in the observed fbank features. It is our conjecture that MVN compensated for that effect. In cases where large fluctuations are more frequent, we see improvements with MVN, but in other cases MVN is not as beneficial. A more thorough investigation is required to quantify the effects of MVN on PFC.

5.5 Summary

We have extended PFC to LVCSR. An incorrect state selection issue caused by a big triphone set in LVCSR can be lessened with a clustering approach. However, there is a trade-off in choosing the number of clusters. With less clusters, there is a smaller risk of incorrect selection; but with more clusters, a more precise side information will be provided to the PFC. The performance gap between the oracle and actual experiments is still rather large. Hence, there are still plenty of studies required in the future to narrow the gap. A better strategy in cluster selection is a critical direction. Creating a collection of consistent model sets as discussed in Section 5.2.1 is also another promising direction.

CHAPTER 6

CONCLUSION AND SUGGESTIONS FOR FUTURE WORK

In this dissertation, we proposed a framework for the joint tracking of the speech and the noise signals for robust speech recognition. Although, many approaches have been proposed for robust speech recognition under noisy conditions in the past, few can adapt to the challenging non-stationary noise conditions. PFC is the first algorithm that tracks the clean speech signal directly for robust ASR. By doing so, PFC can dynamically adapt to the changing noise conditions within an utterance. Another innovation of the PFC algorithm is the integration of HMMs with particle filters, and the exploitation of the statistical structure of a signal, stored by the HMMs, in a tracking scheme. Separate HMM sets are used for compensation at the front end and recognition at the back end. Consequently, the HMM set that is integrated within the particle filter is simpler and allows for easier selection of the side information. The resultant approach is a unified framework for estimation of the clean speech and the noise component of the observed noisy signal, with the ultimate aim of obtaining compensated features for ASR.

6.1 Summary of research

The PFC algorithm is motivated by the evidence of speech tracking performed by human beings, who utilize the time and direction of arrival of the speech signal on each earlobe, as well as the physical structure of the speech signal from the source of interest for better speech perception. We cannot claim to have achieved the level of performance attainable by human listeners, however, we have showed that the idea of speech tracking is both powerful and accomplishable for robust ASR. When correct side information for PFC is available, a major improvement is observed in the recognition accuracy for speech signals captured in noisy environments. The improvement is seen in the simple connected digit tasks as well as the more complex LVCSR tasks. With accurate side information, a digit error reduction

of 67% from multicondition training is attainable for the connected digit task and 28.46% for the LVCSR task.

Obtaining accurate side information in real scenarios is a challenging task. To simplify the search for the side information, the HMMs used to generate particle samples are modified. The number of statistical unit blocks, the states of HMMs, is reduced by a ratio of $1:20$ by clustering. For the LVCSR tasks, where thousands of statistical units are required to represent the triphone level information, the reduction ratio was elevated to as high as approximately $1:500$, and even more. This simplification of the HMMs used for PFC is possible because the compensation is done exclusively at the front end. Consequently, the compensation phase can be decoupled from the recognition phase. When side information was estimated from the noisy observation with simplified models, a digit error reduction of 13% from multicondition training is attainable for the connected digit task and 3.3% for the LVCSR task over multicondition training.

Apart from simplifying the search for side information, the decoupling of the compensation phase and the recognition phase provides us the leverage to use complex model for the recognition phase. For the recognition phase, the complex models (complex backend) perform better when compared with the simple models (simple backend). Accordingly, while we reduce the complexity of the HMMs used in the compensation phase, the complexity of the HMMs used in the recognition phase was increased by boosting the number of GMMs for each state from three to twenty. The word accuracy improved to 91.61% compared to 90.53% achieved with the simple backend setup.

The noise parameters are an important component of the PFC algorithm, and are obtained from the environment detected before the onset of the speech signal. In this thesis, two methods are proposed to dynamically update the parameters during an utterance. Very few existing methods for robust ASR have the capability to handle non-stationary noises within an utterance. The tracking of the noise parameters allows us to lift this constraint.

The speech tracking part of PFC provides information regarding the speech signal and facilitates the tracking and estimation of the noise signal during an utterance.

In the first method, we use a particle filter algorithm, which runs in parallel to PFC, for tracking of the noise parameters. Simulations show that the algorithm successfully captures the variations of the noise signal. The reduction in WER, however, is marginal. There are two reasons for the limited improvement in the recognition performance. One reason is that PFC is itself robust to noise variations within an utterance and therefore the margin of improvement is small to begin with. Secondly, the data sets used for evaluation of the algorithm has limited variations in the noise parameters within an utterance. Therefore, the efficacy of the noise tracking part is not fully revealed.

In the second approach for updating of noise statistics, an MCMC based sampling scheme is proposed in this thesis. The underlying distributions of the speech and noise component of the noisy signal are estimated at each frame and inferences are made from these distributions. Compared to the PF approach, MCMC utilizes more information from the estimated underlying clean speech distribution obtained with PFC. For this reason, it produces a better recognition performance compared to the one obtained with the PF based algorithm. Word accuracy improves from 91.18% for PF to 91.26% for MCMC when tested on the additive noise condition.

To investigate the potential of the PFC algorithm in more challenging large vocabulary systems, we implemented a PFC scheme for the recognition of the Aurora-4 LVCSR test corpus. Results for this test set show that PFC is equally effective for the LVCSR tasks. When accurate side information is known, the word accuracy improves from 79.87% to 85.6%. Even when the complexity of the models used for sample generation is reduced by a ratio of $1:150$, word accuracy remains as high as 84.03%

When side information is obtained from the noisy observations, best performance is observed for the $3-\textit{cluster}$ case (Simplification of $1:500$). The WER in this case reduces by 3.3% and the word accuracy is 80.53%. However, when MVN is combined with PFC,

word accuracy improves to 82.6%. There is still a large margin for improvement for the LVCSR case when PFC is used alone. The improvement can be attained by having a better alignment between the models used to compute the side information and the models used to generate the samples. The models used for extracting the side information are multicondition MFCC HMMs representing triphone models, whereas the models used to generate the samples are clean fbank HMMs representing monophones. As a result, there can be a severe mismatch between the two. With better alignment, we hope to improve the ASR performance of PFC for the LVCSR task.

6.2 Suggestions for future work

PFC is a novel concept both in terms of its technique and its application. Experimental results have shown its enormous potential. Here, we present suggestions for other applications of PFC as well as recommendations to lead the PFC towards its true potential.

6.2.1 Improving the side information estimation

Availability of accurate side information has the most significant effect on enhancing the recognition performance of the PFC algorithm. Following steps are proposed to improve the estimation of the side information:

1. Better side information can be obtained if it is computed iteratively. The transcript generated during the final recognition recognition of the algorithm can be used to obtain the side information for the next iteration.

2. Storing and exploiting the tree structure of the clustering algorithm can help us in making better cluster choice in a large cluster set. If the validity of a particular cluster selection is verified, it will be used for sample generation. Otherwise, we move one step up the tree structure where there are less clusters, and consequently, finding the correct cluster is easier.

3. Using different clustering strategies that reduce the confusions between the cluster

during the cluster selection process.

4. Improve alignment between different set of models used in the PFC algorithm. MAP is a good candidate for this task because it only updates the observation densities and does not alter the state transition matrices.

5. Use the 6 manners of speech attribute (e.g. vowel, nasal, fricative, etc) as the class and use Deep Neural Network to estimate it (Suggested by Dr Xiao Xiong).

6.2.2 Improvements for the PFC algorithm

Following two suggestions are proposed for the overall improvement of the PFC algorithm:

1. When compensating a speech utterance, the complete noisy utterance is available in most cases. Under such circumstances, particle smoothing can be used for estimation of the speech signal instead of the particle filtering. However, the implementation of the particle smoothing is more complex compared to that of the particle filtering.

2. The MCMC sampling algorithm can generate samples from densities of multiple parameters sequentially. In the future, we can add the channel parameter to the speech and the noise parameter inferred using MCMC in the current algorithm.

6.2.3 Theoretical aspects of the PFC algorithm

In this thesis, the efficacy of the PFC algorithm was proved via experimental results. In the future, it is essential that the convergence results for PFC are also obtained. This will bring forth various properties of PFC and provide the insights needed to further improve the algorithm.

6.2.4 Implementation of PFC in other areas

HMMs have been successfully used in many areas outside speech processing such as (e.g. image recognition, cryptanalysis, bioinformatics, etc.). PFC can potentially be used in other cases where the structure of the signal is preserved using HMMs, and an observation model for the distorted version of the signal being estimated is available.

REFERENCES

[1] P. Assmann and Q. Summerfield, "The perception of speech under adverse conditions," *Speech processing in the auditory system*, pp. 231–308, 2004.

[2] S. Greenberg and W. Ainsworth, *Listening to speech: an auditory perspective*. Lawrence Erlbaum, 2006.

[3] A. Sankar and C.-H. Lee, "A maximum-likelihood approach to stochastic matching for robust speech recognition," *Speech and Audio Processing, IEEE Transactions on*, vol. 4, no. 3, pp. 190–202, 1996.

[4] A. Surendran, C.-H. Lee, and M. Rahim, "Nonlinear compensation for stochastic matching," *Speech and Audio Processing, IEEE Transactions on*, vol. 7, no. 6, pp. 643–655, 1999.

[5] C.-H. Lee, "On stochastic feature and model compensation approaches to robust speech recognition," *Speech Communication*, vol. 25, no. 1, pp. 29–47, 1998.

[6] A. Acero, L. Deng, T. Kristjansson, and J. Zhang, "Hmm adaptation using vector taylor series for noisy speech recognition," in *Sixth International Conference on Spoken Language Processing*, 2000.

[7] A. Acero and R. Stern, "Environmental robustness in automatic speech recognition," in *Acoustics, Speech, and Signal Processing, 1990. ICASSP-90., 1990 International Conference on*, pp. 849–852, IEEE, 1990.

[8] B. Raj, R. Singh, and R. Stern, "On tracking noise with linear dynamical system models," in *Acoustics, Speech, and Signal Processing, 2004. Proceedings.(ICASSP'04). IEEE International Conference on*, vol. 1, pp. I–965, IEEE, 2004.

[9] M. Fujimoto and S. Nakamura, "Particle filter based non-stationary noise tracking for robust speech recognition," in *Proceedings of International Conference on Acoustics, Speech, and Signal Processing (ICASSP*, pp. 257–260, 2005.

[10] M. Fujimoto and S. Nakamura, "Sequential non-stationary noise tracking using particle filtering with switching dynamical system," in *Acoustics, Speech and Signal Processing, 2006. ICASSP 2006 Proceedings. 2006 IEEE International Conference on*, vol. 1, pp. I–I, IEEE, 2006.

[11] A. Mushtaq, Y. Tsao, and C.-H. Lee, "A particle filter feature compensation approach to robust speech recognition," in *Eleventh Annual Conference of the International Speech Communication Association*, 2010.

[12] A. Mushtaq and C.-H. Lee, "An integrated approach to feature compensation combining particle filters and hidden markov model for robust speech recognition," in *Acoustics, Speech and Signal Processing, 2012. ICASSP 2012 Proceedings. 2012 IEEE International Conference on*, vol. 1, pp. I–I, IEEE, 2012.

[13] A. Mushtaq and C.-H. Lee, "Joint tracking of clean speech and noise using hmms and particle filters for robust speech recognition," *Asilomar Conference on Signals, Systems and Computers*, 2012.

[14] A. Mushtaq and C.-H. Lee, "An mcmc approach to joint estimation of clean speech and noise for robust speech recognition," in *Acoustics, Speech and Signal Processing, 2013. ICASSP 2013 Proceedings. 2012 IEEE International Conference on*, vol. 1, pp. I–I, IEEE, 2013.

[15] C.-H. Lee and Q. Huo, "On adaptive decision rules and decision parameter adaptation for automatic speech recognition," *Proceedings of the IEEE*, vol. 88, no. 8, pp. 1241–1269, 2000.

[16] L. Rabiner and B. Juang, "An introduction to hidden markov models," *ASSP Magazine, IEEE*, vol. 3, no. 1, pp. 4–16, 1986.

[17] J. Gauvain and C.-H. Lee, "Maximum a posteriori estimation for multivariate gaussian mixture observations of markov chains," *Speech and Audio Processing, IEEE Transactions on*, vol. 2, no. 2, pp. 291–298, 1994.

[18] K. Shinoda and C.-H. Lee, "A structural bayes approach to speaker adaptation," *Speech and Audio Processing, IEEE Transactions on*, vol. 9, no. 3, pp. 276–287, 2001.

[19] O. Siohan, C. Chesta, and C.-H. Lee, "Joint maximum a posteriori adaptation of transformation and hmm parameters," *Speech and Audio Processing, IEEE Transactions on*, vol. 9, no. 4, pp. 417–428, 2001.

[20] C. Leggetter and P. Woodland, "Maximum likelihood linear regression for speaker adaptation of continuous density hidden markov models," *Computer speech and language*, vol. 9, no. 2, p. 171, 1995.

[21] C. Chesta, O. Siohan, and C.-H. Lee, "Maximum a posteriori linear regression for hidden markov model adaptation," in *Sixth European Conference on Speech Communication and Technology*, 1999.

[22] M. Gales, "Model-ased techniques for noise ro ust speech recognition," 1995.

[23] M. Gales and S. Young, "Robust continuous speech recognition using parallel model combination," *Speech and Audio Processing, IEEE Transactions on*, vol. 4, no. 5, pp. 352–359, 1996.

[24] T. Moon, "The expectation-maximization algorithm," *Signal Processing Magazine, IEEE*, vol. 13, no. 6, pp. 47–60, 1996.

[25] L. Lamel and J. Gauvain, "High performance speaker-independent phone recognition using cdhmm," in *Third European Conference on Speech Communication and Technology*, 1993.

[26] B. Juang, W. Hou, and C.-H. Lee, "Minimum classification error rate methods for speech recognition," *Speech and Audio Processing, IEEE Transactions on*, vol. 5, no. 3, pp. 257–265, 1997.

[27] J. Li, M. Yuan, and C.-H. Lee, "Approximate test risk bound minimization through soft margin estimation," *Audio, Speech, and Language Processing, IEEE Transactions on*, vol. 15, no. 8, pp. 2393–2404, 2007.

[28] L. Bahl, P. Brown, P. De Souza, and R. Mercer, "Maximum mutual information estimation of hidden markov model parameters for speech recognition," in *Acoustics, Speech, and Signal Processing, IEEE International Conference on ICASSP'86.*, vol. 11, pp. 49–52, IEEE, 1986.

[29] D. Povey, D. Kanevsky, B. Kingsbury, B. Ramabhadran, G. Saon, and K. Visweswariah, "Boosted mmi for model and feature-space discriminative training," in *Acoustics, Speech and Signal Processing, 2008. ICASSP 2008. IEEE International Conference on*, pp. 4057–4060, IEEE, 2008.

[30] P. Moreno, B. Raj, and R. Stern, "A vector taylor series approach for environment-independent speech recognition," in *Acoustics, Speech, and Signal Processing, 1996. ICASSP-96. Conference Proceedings., 1996 IEEE International Conference on*, vol. 2, pp. 733–736, IEEE, 1996.

[31] M. Gales and U. of Cambridge. Engineering Dept, "Maximum likelihood linear transformations for hmm-based speech recognition," *Computer speech and language*, vol. 12, no. 2, 1998.

[32] K. Visweswariah, V. Goel, and R. Gopinath, "Structuring linear transforms for adaptation using training time information," in *Acoustics, Speech, and Signal Processing, 1993. ICASSP-93., 1993 IEEE International Conference on*, vol. 1, pp. I–I, IEEE, 1993.

[33] B. Atal, "Effectiveness of linear prediction characteristics of the speech wave for automatic speaker identification and verification," *the Journal of the Acoustical Society of America*, vol. 55, p. 1304, 1974.

[34] O. Viikki and K. Laurila, "Cepstral domain segmental feature vector normalization for noise robust speech recognition," *Speech Communication*, vol. 25, no. 1-3, pp. 133–147, 1998.

[35] A. De La Torre, A. Peinado, J. Segura, J. Pérez-Córdoba, M. Benítez, and A. Rubio, "Histogram equalization of speech representation for robust speech recognition," *Speech and Audio Processing, IEEE Transactions on*, vol. 13, no. 3, pp. 355–366, 2005.

[36] F. Zheng, G. Zhang, and Z. Song, "Comparison of different implementations of mfcc," *Journal of Computer Science and Technology*, vol. 16, no. 6, pp. 582–589, 2001.

[37] V. Tyagi and C. Wellekens, "On desensitizing the mel-cepstrum to spurious spectral components for robust speech recognition," in *Proc. ICASSP*, vol. 1, pp. 529–532, 2005.

[38] H. Hermansky, "Perceptual linear predictive (plp) analysis of speech," *The Journal of the Acoustical Society of America*, vol. 87, p. 1738, 1990.

[39] S. Davis and P. Mermelstein, "Comparison of parametric representations for monosyllabic word recognition in continuously spoken sentences," *Acoustics, Speech and Signal Processing, IEEE Transactions on*, vol. 28, no. 4, pp. 357–366, 1980.

[40] O. Richard, E. Peter, and G. David, "Pattern classification," *A Wiley-Interscience Publication*, vol. 373378, 2001.

[41] E. Standard, "Speech processing, transmission and quality aspects (stq); distributed speech recognition; advanced front-end feature extraction algorithm; compression algorithms," *ETSI ES*, vol. 202, no. 050, p. V1.

[42] D. Macho, L. Mauuary, B. Noé, Y. Cheng, D. Ealey, D. Jouvet, H. Kelleher, D. Pearce, and F. Saadoun, "Evaluation of a noise-robust dsr front-end on aurora databases," in *Seventh International Conference on Spoken Language Processing*, 2002.

[43] D. Van Compernolle, "Noise adaptation in a hidden markov model speech recognition system," *Computer Speech & Language*, vol. 3, no. 2, pp. 151–167, 1989.

[44] S. Boll, "Suppression of acoustic noise in speech using spectral subtraction," *Acoustics, Speech and Signal Processing, IEEE Transactions on*, vol. 27, no. 2, pp. 113–120, 1979.

[45] L. Arslan, A. McCree, and V. Viswanathan, "New methods for adaptive noise suppression," in *Acoustics, Speech, and Signal Processing, 1995. ICASSP-95., 1995 International Conference on*, vol. 1, pp. 812–815, IEEE, 1995.

[46] T. Houtgast, "Predicting speech intelligibility in rooms from the modulation transfer function. i. general room acoustics," *Acustica*, vol. 46, pp. 60–72, 1980.

[47] J. Van Dijkhuizen, P. Anema, and R. Plomp, "The effect of varying the slope of the amplitude-frequency response on the masked speech-reception threshold of sentences," *The Journal of the Acoustical Society of America*, vol. 81, p. 465, 1987.

[48] R. J. Stubbs and Q. Summerfield, "Effects of signal-to-noise ratio, signal periodicity, and degree of hearing impairment on the performance of voice-separation algorithms," *The Journal of the Acoustical Society of America*, vol. 89, p. 1383, 1991.

[49] E. C. Cherry, "Some experiments on the recognition of speech, with one and with two ears," *The Journal of the acoustical society of America*, vol. 25, no. 5, pp. 975–979, 1953.

[50] C. J. Darwin, "Perceptual grouping of speech components differing in fundamental frequency and onset-time," *The Quarterly Journal of Experimental Psychology*, vol. 33, no. 2, pp. 185–207, 1981.

[51] W. M. Hartmann, "Pitch perception and the segregation and integration of auditory entities," *Auditory Function, Neurobiological Bases of Hearing*, pp. 623–645, 1988.

[52] M. T. M. Scheffers, *Sifting vowels: auditory pitch analysis and sound segregation*. PhD thesis, Rijksuniversiteit te Groningen, 1983.

[53] B. C. Moore, "Temporal integration and context effects in hearing," *Journal of Phonetics*, vol. 31, no. 3, pp. 563–574, 2003.

[54] G. A. Miller and J. Licklider, "The intelligibility of interrupted speech," *The Journal of the Acoustical Society of America*, vol. 22, no. 2, pp. 167–173, 1950.

[55] T. Moon and W. Stirling, *Mathematical methods and algorithms for signal processing*, vol. 204. Prentice hall, 2000.

[56] M. Arulampalam, S. Maskell, N. Gordon, and T. Clapp, "A tutorial on particle filters for online nonlinear/non-gaussian bayesian tracking," *Signal Processing, IEEE Transactions on*, vol. 50, no. 2, pp. 174–188, 2002.

[57] W. L. Dunn and J. K. Shultis, *Exploring Monte Carlo Methods*. Elsevier Science, 2011.

[58] L. Devroye, "Sample-based non-uniform random variate generation," in *Proceedings of the 18th conference on Winter simulation*, pp. 260–265, ACM, 1986.

[59] R. M. Neal, "Slice sampling," *Annals of statistics*, pp. 705–741, 2003.

[60] P. W. Glynn and D. L. Iglehart, "Importance sampling for stochastic simulations," *Management Science*, vol. 35, no. 11, pp. 1367–1392, 1989.

[61] W. R. Gilks, S. Richardson, and D. J. Spiegelhalter, *Markov chain Monte Carlo in practice*, vol. 2. Chapman & Hall/CRC, 1996.

[62] P. D. Hoff, *A first course in Bayesian statistical methods*. Springer, 2009.

[63] W. R. Gilks, N. Best, and K. Tan, "Adaptive rejection metropolis sampling within gibbs sampling," *Applied Statistics*, pp. 455–472, 1995.

[64] N. Bergman, "Recursive bayesian estimation," *Department of Electrical Engineering, Linköping University, Linköping Studies in Science and Technology. Doctoral dissertation*, no. 579, 1999.

[65] J. S. Liu and R. Chen, "Sequential monte carlo methods for dynamic systems," *Journal of the American statistical association*, vol. 93, no. 443, pp. 1032–1044, 1998.

[66] A. Doucet and A. Johansen, "A tutorial on particle filtering and smoothing: Fifteen years later," *Handbook of Nonlinear Filtering*, pp. 656–704, 2009.

[67] S. Haykin, "Adaptive filter theory, 2002."

[68] Y. Ephraim, "A bayesian estimation approach for speech enhancement using hidden markov models," *Signal Processing, IEEE Transactions on*, vol. 40, no. 4, pp. 725–735, 1992.

[69] T. Masuko, K. Tokuda, T. Kobayashi, and S. Imai, "Speech synthesis using hmms with dynamic features," in *Acoustics, Speech, and Signal Processing, 1996. ICASSP-96. Conference Proceedings., 1996 IEEE International Conference on*, vol. 1, pp. 389–392, IEEE, 1996.

[70] K. W. Church, "A stochastic parts program and noun phrase parser for unrestricted text," in *Proceedings of the second conference on Applied natural language processing*, pp. 136–143, Association for Computational Linguistics, 1988.

[71] C. Darwin and R. P. Carlyon, "Auditory grouping," *The handbook of perception and cognition*, vol. 6, pp. 387–424, 1995.

[72] C. M. Bishop, "Pattern recognition and machine learning (information science and statistics)," 2007.

[73] R. Leonard, "A database for speaker-independent digit recognition," in *Acoustics, Speech, and Signal Processing, IEEE International Conference on ICASSP'84.*, vol. 9, pp. 328–331, IEEE, 1984.

[74] S. Young, J. Odell, and P. Woodland, "Tree-based state tying for high accuracy acoustic modelling," in *Proceedings of the workshop on Human Language Technology*, pp. 307–312, Association for Computational Linguistics, 1994.

[75] S. Chib and E. Greenberg, "Understanding the metropolis-hastings algorithm," *The American Statistician*, vol. 49, no. 4, pp. 327–335, 1995.