Signal Modeling Techniques In Speech Recognition

by,

Joseph Picone

Texas Instruments
Systems and Information Sciences Laboratory
Tsukuba Research and Development Center
Tsukuba, Japan

ABSTRACT

We have seen three important trends develop in the last five years in speech recognition. First, heterogeneous parameter sets that mix absolute spectral information with dynamic, or time-derivative, spectral information, have become common. Second, similarity transform techniques, often used to normalize and decorrelate parameters in some computationally inexpensive way, have become popular. Third, the signal parameter estimation problem has merged with the speech recognition process so that more sophisticated statistical models of the signal's spectrum can be estimated in a closed-loop manner. In this paper, we review the signal processing components of these algorithms. These algorithms are presented as part of a unified view of the signal parameterization problem in which there are three major tasks: measurement, transformation, and statistical modeling.

This paper is by no means a comprehensive survey of all possible techniques of signal modeling in speech recognition. There are far too many algorithms in use today to make an exhaustive survey feasible (and cohesive). Instead, this paper is meant to serve as a tutorial on signal processing in state-of-the-art speech recognition systems and to review those techniques most commonly used. In keeping with this goal, a complete mathematical description of each algorithm has been included in the paper.

I. INTRODUCTION

Parameterization of an analog speech signal is the first step in the speech recognition process. Several popular signal analysis techniques have emerged as *de facto* standards in the literature. These algorithms are intended to produce a "perceptually meaningful" parametric representation of the speech signal: parameters that emulate some of the behavior observed in the human auditory and perceptual systems. Of course, and perhaps more importantly, these algorithms are also designed to maximize recognition performance.

The roots of many of these techniques can be traced to early speech recognition research on speaker dependent technology. Today, though significant portions of speech recognition research are now focused on the speaker independent recognition problem, many of these parameterizations continue to be useful. In speaker independent speech recognition, a premium is placed on developing descriptions that are somewhat invariant to changes in the speaker. Parameters that represent salient spectral energies of the sound, rather than details of the particular speaker's voice, are desired.

In this paper, we will adopt a view that a syntactic pattern recognition approach to speech recognition consists of two fundamental operations: signal modeling and network searching. **Signal modeling** represents the process of converting sequences of speech samples to observation vectors representing events in a probability space. **Network searching** is the task of finding the most probable sequence of these events given some syntactic constraints. In this tutorial, we present an overview of popular approaches to signal modeling in speech recognition.

1.1 The Signal Model Paradigm

Signal modeling can be subdivided into four basic operations: spectral shaping, spectral analysis, parametric transformation, and statistical modeling. The complete sequence of steps is summarized in Fig. 1. The first three operations are straightforward problems in digital signal processing. The last task, however, is often divided between the signal modeling system and the speech recognition system.

There are three main driving forces in designing signal modeling systems. First, parameterizations are sought that represent salient aspects of the speech signal, preferably parameters that are analogous to those used by the human auditory system. This is often referred to as **perceptually-meaningful** parameters. Second, parameterizations are desired that are robust to variations in channel, speaker, and transducer. We refer to this as the robustness, or **invariance**, problem. Finally, most recently, parameters that capture spectral dynamics, or changes of the spectrum with time, are desired. We refer to this as the **temporal correlation** problem. With the introduction of Markov modeling techniques that are capable of statistically modeling the time course of the signal, parameters that incorporate both absolute and differential measurements of the signal spectrum have become increasingly common.

Signal modeling now requires less than 10% of the total processing time required in a typical large vocabulary speech recognition application. The difference in processing time between various signal modeling approaches is now a small percentage of the total processing time. The focus today has shifted towards maintaining high performance and minimizing the number of degrees of freedom. Parameterizations that concisely describe the

signal, can be easily computed in fixed point hardware, and can be compressed through straightforward quantization techniques are often preferred over more exotic approaches. Memory considerations often outweigh any small gains that may be achieved in speech recognition performance by a new signal model.

Historically, robustness to background acoustic noise has been a major driving force in the design of signal models. In fact, many of the signal models in use today were the outgrowth of research into applications involving noisy environments: voice control of military instrumentation (speech recognition in the cockpit) [1-2] and voice control of the telephone (automatic telephone transactions) [3-6]. As speech recognition technologies have become more sophisticated, the recognition system itself now contributes more to the noise robustness problem than the signal model. Hence, it is often difficult to isolate signal modeling algorithm enhancements.

In addition, signal models that are good for one type of application may not necessarily be optimal for another. For example, in speaker independent speech recognition targeted for a single environment (for example, continuous digit recognition for telecommunications applications), certain types of statistical variations of the channel and speakers can be safely predicted and accounted for *a priori* (*e.g.*, the bandwidth of the channel). In speaker dependent or speaker identification applications, learning unique characteristics of the user and the user's acoustic environment is important. Though this difference might seem to necessitate different signal modeling approaches, most approaches discussed in this paper work well in both types of applications.

1.2 Terminology

Throughout this paper[1], we will avoid the overworked and all encompassing term "feature extraction" for two reasons. First, most often this term conveys some connotation that the amount of information has been reduced (distilled). Salient features of the speech signal are strongly context dependent. No feature extraction algorithm can magically normalize all variations in the observed data without some knowledge of the context of the sound. We often prefer methods that preserve spectral variation in the data, rather than those that attempt to remove it in early stages of the processing. Our disposition is to let the speech recognizer deal with statistical variation in the data.

Second, the term "feature extraction" somehow implies we know what we are looking for (in the signal). At this early stage in speech recognition history, there are no absolutes. The merit of a signal model must be measured in the context of a recognition task. Various objective measures of modeling accuracy or efficiency, such as distortion, have no strong correlation with recognition performance. In fact, the best feature set can often be a function of the recognition algorithm and the task. The end goal is to preserve those dimensions in the data that represent dimensions in which fundamental sound units can be discriminated. The rather grim reality is that many signal models in use today are great achievements in empirical optimization.

Having said this, what term should we use? We prefer the use of the simple term **signal model**. A signal model will have three internal components: measurements — basic spectral and temporal measurements;

1. Since no field today is worth its salt without a plateful of jargon, a few brief comments on terminology are made in this section.

parameters — parametrically collated and smoothed versions of these measurements; and observations — the output of some form of statistical model of the parameters. The signal model's observations are of course intimately interrelated with the speech recognition technology. These internal components are shown in Fig. 1.

Let us now describe each of these steps in greater detail. We note that it is a shame that very few, if any, speech recognition systems are capable of exhaustively comparing many variants of signal models in a controlled manner. Hence, it is often the case that motivations for choosing a particular approach are not always scientific. Therefore, we conclude this paper with an overview of common signal models used in today's state of the art speech recognition systems, and make a few comments on the respective author's claims about the merits of their approach. Excellent papers on this and other related topics can be found in [7-26][2].

II. SPECTRAL SHAPING

Spectral shaping involves two basic operations: **A/D conversion —** conversion of the signal from a sound pressure wave to a digital signal; and **digital filtering —** emphasizing important frequency components in the signal. This conversion process is shown in Fig. 2. A good discussion of general principles of sampling and A/D conversion can be found in [30]. We will not discuss the choice of a signal sample frequency and the implications of such choices in this tutorial, though choice of an appropriate sample frequency obviously plays an important part in the signal modeling problem.

The microphone used in the A/D conversion process usually introduces undesired side effects, such as line frequency noise ("50/60 Hz hum"), loss of low and high frequency information, and nonlinear distortion. The A/D converter also introduces its own distortion in the form of a less than ideal frequency response and nonlinear input/output transfer function, and fluctuating DC bias. An example of the frequency response of a typical telephone grade channel (including A/D conversion) is shown in Fig. 3. The sharp attenuation of low and high frequencies often causes problems for the subsequent parametric spectral analyses algorithms.

Because of the limited frequency response of analog telecommunications channels, and the widespread use of 8 kHz sampled speech in digital telephony, the most popular sample frequency for the speech signal in telecommunications is 8 kHz. With the recent emergence of broadband digital networks, however, we may soon see new telecommunications applications that utilize higher quality audio input. In non-telecommunications applications, in which the speech recognition subsystem has access to high quality speech, sample frequencies of 10 kHz, 12 kHz, and 16 kHz have been used. These sample frequencies give better time and frequency resolution [31].

The main purpose of the digitization process is to produce a sampled data representation of the speech signal with as high a Signal to Noise ratio (SNR) as possible. Telecommunications systems today regularly deliver SNRs in excess of 30 dB for speech recognition applications, more than sufficient for obtaining high performance. Variations in

2. The references in this paper have been selected mainly for their worth as general introductions to mainstream work in this area, rather than their authenticity as an original reference on the subject. It was not our intention to discredit particular research in this area (though that is probably unavoidable). Excellent comprehensive discussions of many topics presented in this paper can be found in [27-29].

transducers, channels, and background noise, however, can each contribute significantly to problematic performance in such environments.

Once signal conversion is complete, the last step of digital postfiltering is most often executed using a Finite Impulse Response (FIR) filter:

$$H_{pre}(z) = \sum_{k=0}^{N_{pre}} a_{pre}(k) z^{-k} . \quad (1)$$

Normally, a one coefficient digital filter, known as a **preemphasis** filter, is used:

$$H_{pre}(z) = 1 + a_{pre} z^{-1} . \quad (2)$$

A typical range of values for a_{pre} is $[-1.0, -0.4]$. Values close to -1.0 that can be efficiently implemented in fixed point hardware, such as -1 or $-(1 - 1/16)$, are most common in speech recognition. A range of frequency responses for the preemphasis filter of Eq. (2) is shown in Fig. 4. The preemphasis filter is intended to boost the signal spectrum approximately 20 dB per decade (an order of magnitude increment in frequency).

There are two common explanations of the advantages of using this filter. First, voiced sections of the speech signal naturally have a negative spectral slope (attenuation) of approximately 20 dB per decade due to physiological characteristics of the speech production system [28,31]. The preemphasis filter serves to offset this natural slope before spectral analysis, thereby improving the efficiency of the analysis [31,32].

An alternate explanation is that hearing is more sensitive above the 1 kHz region of the spectrum. The preemphasis filter amplifies this area of the spectrum, assisting the spectral analysis algorithm in modeling the most perceptually important aspects of the speech spectrum [31] (see Section 3.3.4 for more details).

We also note that such preemphasis filters also raise frequencies above 5 kHz, a region in which the auditory system becomes increasingly less sensitive. However, frequencies above 5 kHz are naturally attenuated by the speech production system and normally are assigned a significantly smaller weight in a typical speech recognition system.

More sophisticated preemphasis algorithms have been proposed. One such notable approach is adaptive preemphasis, in which the spectral slope is automatically flattened [31] before spectral analysis. Other algorithms utilize shaping filters that attenuate areas of the spectrum known to be quite noisy [33,34]. More recently, speech/noise classification algorithms based on adaptive filtering are being employed [35]. However, none of these approaches have yet enjoyed widespread success in speech recognition applications. In fact, recently, many speech recognition systems have eliminated the preemphasis stage altogether and compensate for the spectral slope as part of the speech recognition statistical model (see Section VI).

III. SPECTRAL ANALYSIS

For pedagogical reasons, let us classify the types of spectral measurements used in speech recognition systems into two classes: power — measures of the gross spectral (or temporal) power of the signal; spectral amplitude — measures of power over particular frequency intervals in the spectrum. A typical parameter set in speech recognition will include each of these measurements. Recently there has been resurgence of interest[3] in fundamental frequency for use as a prosodic feature [36], for use in speech recognition of

tonal languages (e.g. Chinese) or languages that have some tonal components (e.g., Japanese), and as a measure of speaker identity or authenticity [37]. Let us first briefly review fundamental frequency and power calculations, and then focus on spectral amplitude estimation.

3.1 Fundamental Frequency

Fundamental frequency[4] is defined as the frequency at which the vocal cords vibrate during a voiced sound [38,39]. Fundamental frequency (f_0) has long been a difficult parameter to reliably estimate from the speech signal. Previously, it has been neglected in speech recognition systems for numerous reasons, including the large computational burden required for accurate estimation, the concern that unreliable estimation would be a barrier to achieving high performance, and the difficulty in characterizing complex interactions between f_0 and suprasegmental phenomena.

There are four major classes of algorithms in use today. One of the first algorithms to appear, and one of the simplest, is an algorithm that uses multiple measures of periodicity in the signal, and votes between them to determine the voicing state and fundamental frequency. This algorithm was originally known in the speech processing literature as the Gold-Rabiner algorithm [40], and motivated many other variants based on time-domain measurements [41]. The Gold-Rabiner algorithm is still popular mainly because of its simplicity and ease of reliable implementation. Unfortunately, it does not work very well.

Second, the U.S. National Security Agency (NSA), as part of a program to develop secure digital telephones based on low bit rate voice coding, has developed a robust algorithm for telecommunications applications [42,43]. This algorithm is based on the average magnitude difference function [32], and a discriminant analysis of multiple voicing measures. It is a published government standard and publicly available in the U.S.

A third class of algorithms, similar in nature to the previous class, are based on dynamic programming concepts [44]. These algorithms have been shown to provide high performance across a wide range of environments, including noisy telecommunications channels. This class of algorithms uses a sophisticated optimization procedure that evaluates several measures of correlation and spectral change in the signal, and computes an optimal fundamental frequency pattern and voicing pattern simultaneously.

Finally, an algorithm that is rarely used in real-time speech systems, but often used for research experimentation, is an algorithm that operates on the cepstrum of the speech signal [45]. This algorithm is still popular today as an accurate method for estimating the fundamental frequency in extremely quiet laboratory recording conditions.

Fundamental frequency is often processed on a logarithmic scale, rather than a linear scale, to match the resolution of the human auditory system. For reference purposes, let us define a measure of the fundamental frequency as:

$$F(n) = \log_{10}(f_0(n)) \quad , \tag{3}$$

where n represents discrete time.

3. Of course, fundamental frequency is still rarely used in practical speech recognition systems.

4. This section is intended to serve only as a reference guide to major work in this area. The details of such algorithms are beyond the scope of this paper. Good tutorials on the subject can be found in [38,39]

Normally, $50 \text{ Hz} \le f_0 \le 500 \text{ Hz}$ for voiced speech. For unvoiced speech, f_0 is undefined, and by convention $F \equiv 0$. Often, fundamental frequency is normalized by the speaker's average f_0 value, or some physiologically-motivated transformation of a nominal value during the corresponding voiced segment of speech.

3.2 Power

The use of some sort of power measure(s) in speech recognition is fairly standard today. **Power** is rather simple to compute:

$$P(n) = \frac{1}{N_s} \sum_{m=0}^{N_s-1} \left(w(m)\, s\left(n - \frac{N_s}{2} + m \right) \right)^2 , \qquad (4)$$

where N_s is the number of samples used to compute the power, $s(n)$ denotes the signal, $w(m)$ denotes a weighting function, and n denotes the sample index (discrete time) of the center of the window. Rather than using power directly, many speech recognition systems use the logarithm of the power multiplied by 10, defined as the power in dB, in an effort to emulate the logarithmic response of the human auditory system [46].

The weighting function in Eq. (4) is referred to as a window function. Window theory was once a very active topic of research in digital signal processing [31,32]. There are many types of windows including rectangular, Hamming, Hanning, Blackman, Bartlett, and Kaiser. Today, in speech recognition, the Hamming window is almost exclusively used. The Hamming window is a specific case of the Hanning window. A generalized Hanning window is defined as:

$$w(n) = \frac{\alpha_w - (1 - \alpha_w) \cos(2\pi n / (N_s - 1))}{\beta_w} , \qquad (5)$$

for $0 \le n < N_s$, and $w(n) \equiv 0$ elsewhere. α_w is defined as a window constant in the range $[0, 1]$, and N_s is the window duration in samples. To implement a Hamming window, $\alpha_w = 0.54$.

β_w is a normalization constant defined so that the root mean square (RMS) value of the window is unity. β_w is defined as:

$$\beta_w = \sqrt{\frac{1}{N_s} \sum_{n=0}^{N_s-1} w^2(n)} . \qquad (6)$$

In Fig. 5 we show various realizations of the Hanning window.

In practice, it is desirable to normalize the window so that the power in the signal after windowing is approximately equal to the power of the signal before windowing. Equation (6) describes such a normalization constant. This type of normalization is especially convenient for implementations using fixed point arithmetic hardware. Note that the computational burden of a window is relatively small, because the window coefficients are precomputed at system initialization.

The purpose of the window is to weight, or favor, samples towards the center of the window. This characteristic, coupled with the overlapping analysis discussed next, performs an important function in obtaining smoothly varying parametric estimates. It is important that the width of the main lobe in the frequency response of the window be as small as possible, or the windowing process can have a detrimental effect on the subsequent spectral analysis. See [31,32,47] for good discussions of this topic.

Power, like most parameters in a speech recognition system (including fundamental frequency mentioned in the last section) is

computed on a frame by frame basis. **Frame duration**, T_f, is defined as the length of time (in seconds) over which a set of parameters are valid. Frame period is a similarly used term that denotes the length of time between successive parameter calculations. Frame rate, yet another common term, is the number of frames computed per second (Hz).

In Eq. (4), n is updated by the frame duration in samples. Frame duration typically ranges between 20 msec and 10 msec in practical systems. Values in this range represent a trade-off between the rate of change of spectrum and system complexity. The proper frame duration is ultimately dependent on the velocity of the articulators in the speech production system (rate of change of the vocal tract shape). While some speech sounds (such as stop consonants or diphthongs) exhibit sharp spectral transitions which can result in spectral peaks shifting as much as 80 Hz / msec [31], frame durations less than approximately 8 msec are normally not used.

Equally important, however, is the interval over which the power is computed. The number of samples used to compute the summation, N_s, is known as the **window duration** (in samples). Window duration, T_w is normally measured in units of time (seconds).

Window duration controls the amount of averaging, or smoothing, used in the power calculation. The frame duration and window duration together control the rate at which the power values track the dynamics of the signal [32]. Frame duration and window duration are normally adjusted as pair: a window duration of 30 msec is most common with a frame duration of 20 msec, while a window duration of 20 msec is used with a frame duration of 10 msec. Generally speaking, since a shorter frame duration is used to capture rapid dynamics of the spectrum, the window duration should also be correspondingly shorter so that the detail in the spectrum is not excessively smoothed.

The process of frame based analysis is depicted in Fig. 6. This type of analysis is often referred to as an overlapping analysis, because with each new frame, only a fraction of the signal data changes. The amount of overlap to some extent controls how quickly parameters can change from frame to frame. The percentage overlap is given by:

$$\%Overlap = \frac{(T_w - T_f)}{T_w} \times 100\% \ , \qquad (7)$$

where T_w is the window duration (in seconds) and T_f is the frame duration. If $T_w < T_f$, the percentage overlap is zero.

The combination of a 20 msec frame duration and a 30 msec window duration correspond to a 33% overlap. Some systems use as much as 66% overlap. One goal of such large amounts of overlap is to reduce the amount of noise introduced in the measurements by such artifacts as window placement and nonstationary channel noise [32]. On the other hand, excessively smoothed estimates can obscure true variations in the signal.

Frame-based power computations can also be computed recursively [32]. This technique is most easily viewed as a filtering operation of the squared amplitude of the signal:

$$P(n) = -\sum_{i=1}^{N_a} a_{pw}(i) P(n-i) + \sum_{j=0}^{N_b} b_{pw}(j) s^2(n-j) \ , \qquad (8)$$

where $\{a_{pw}\}$ and $\{b_{pw}\}$ represent the coefficients of a digital low pass filter. Most often, a first order filter (leaky integrator),

$N_a = 1, N_b = 0$, or a second order filter (biquad section), $N_a = 2, N_b = 2$, are used. The design of the system represented in Eq. (8) to produce smoothed power estimates is a classical control problem. A good discussion of the design of such controllers can be found in [48].

In Fig. 7, we demonstrate the use of these parameters. At the bottom of the figure, a speech signal is shown. The first four waveforms starting from the top of the figure show power contours for: (a) $T_f = 5$ msec, $T_w = 10$ msec; (b) $T_f = 10$ msec, $T_w = 20$ msec; (c) $T_f = 20$ msec, $T_w = 30$ msec; (d) $T_f = 20$ msec, $T_w = 30$ msec, and a Hamming window was used; (e) $T_f = 20$ msec, $T_w = 60$ msec; (f) a recursive filter approach in which a 50 Hz low pass filter was implemented using a second order section.

The application of a Hamming window, as shown in Fig. 7(d), helps produce a smoothed estimate of the power through regions where the power changes rapidly (note the point in the waveform marked by an arrow). Note that at $t = 0.3$ secs there is a subtle rise in power. This rise in power is reproduced only in (a) and (e), the two analyses with the greatest ability to respond to rapid changes in the signal's power.

The recursive technique in Fig. 7(f) produces an oscillatory power contour. The second order filter used in the implementation is not capable of sufficiently damping high frequencies in the signal's amplitude/ instantaneous power contour. Hence, the output tends to be too sensitive to the short-term power level of the signal. For this reason, such filters are often used only as postprocessors to frame-based analyses, in which case the power contour is extremely smooth to begin with. Careful design of these circuits are required to make sure the adaptation speed is appropriate for the given application.

Recursive formulations are used to implement algorithms for adaptive gain control, peak signal power estimation, and signal endpoint detection. Equation (8) can also be applied as a postprocessor to Eq. (4) to provide additional smoothing of the power estimates.

3.3 Spectral Analysis

There are six major classes of spectral analysis algorithms used in speech recognition systems today. The procedures for generating these analyses are summarized in Fig. 8. Filter bank methods (implemented in analog circuits) were historically the first methods introduced. Linear prediction methods were introduced in the 1970's, and were the dominant technique through the early 1980's. Currently, both Fourier transform and linear prediction techniques enjoy widespread use in various speech processing applications. In this section, we will discuss each of these techniques, beginning with the digital filter bank.

3.3.1 Digital Filter Bank

The digital filter bank is one of the most fundamental concepts in speech processing. A filter bank can be regarded as a crude model of the initial stages of transduction in the human auditory system. There are two main motivations for the filter bank representation. First, the position of maximum displacement along the basilar membrane for stimuli such as pure tones is proportional to the logarithm of the frequency of the tone. This hypothesis is part of a theory of hearing called the "place theory" [49].

Second, experiments in human perception have shown that frequencies of a complex sound within a certain bandwidth of some

nominal frequency cannot be individually identified. When one of the components of this sound falls outside this bandwidth, it can be individually distinguished. We refer to this bandwidth as the critical bandwidth [50]. A critical bandwidth is nominally 10% to 20% of the center frequency of the sound.

We can define a mapping of acoustic frequency, f, to a "perceptual" frequency scale, as follows [51]:

$$Bark = 13\,\mathrm{atan}\left(\frac{0.76f}{1000}\right) + 3.5\,\mathrm{atan}\left(\frac{f^2}{(7500)^2}\right) . \quad (9)$$

The units of this perceptual frequency scale are referred to as critical band rate, or *Bark*. The *Bark* scale is shown in Fig. 9(a).

A more popular approximation to this type of mapping in speech recognition is known as the *mel* scale [51]:

$$mel\ frequency = 2595 \log10 (1 + f/700.0) . \quad (10)$$

The *mel* scale attempts to map the perceived frequency of a tone, or pitch, onto a linear scale. This scale is displayed in Fig. 9(b). It is often approximated as a linear scale from 0 to 1000 Hz, and then a logarithmic scale beyond 1000 Hz.

An expression for critical bandwidth is:

$$BW_{critical} = 25 + 75\,[1 + 1.4\,(f/1000)^2]^{0.69} . \quad (11)$$

This transformation can be used to compute bandwidths on a perceptual scale for filters at a given frequency on *Bark* or *mel* scales. The critical bandwidth function is also displayed in Fig. 9(c).

Both the *Bark* scale and the *mel* scale can be regarded as a transformation of the frequency scale into a perceptually meaningful scale that is linear. The combination of these two theories gave rise to an analysis technique known as the critical band filter bank. A **critical band filter bank** is simply a bank of linear phase FIR bandpass filters that are arranged linearly along the *Bark (or mel)* scale. The bandwidths are chosen to be equal to a critical bandwidth for the corresponding center frequency.

One such filter bank, originally defined in [52] has become somewhat of a standard, and is shown in Table 1. The center frequencies and bandwidths for these filters are shown in the second and third columns of Table 1. The center frequencies correspond to those frequencies for which Eq. (9) yields the integer index value in the table (for example, the frequency corresponding to an index of 2 produces a *Bark* value of 2). The bandwidth is then computed using Eq. (11).

In many speech processing applications, the first filter is omitted, because its range is beyond the capabilities of the A/D converter. Often the sampled data collected in the corresponding frequency range is extremely noisy. Telephone grade speech is often processed using a filter bank consisting of 16 bands (indices 2 - 17).

Another equally important filter bank in the speech recognition literature is a filter bank based on the *mel* scale. The frequency/ bandwidths for this filter bank [26] are given in fourth and fifth columns of Table 1. In this design, ten filters are assigned linearly from 100 Hz to 1000 Hz. Above 1000 Hz, five filters are assigned for each doubling of the frequency scale (octave). These filters are spaced logarithmically (equally spaced on a log scale). The bandwidths are assigned such that the 3 dB point is halfway between the current bin and the previous/next bin. The shaded entries in the table are shown only for comparison purposes. Normally, only the first 20 samples from the filter bank are used.

Each filter in the digital filter bank is usually implemented as a linear phase filter so

that the group delay for all filters is equal to zero, and the output signals from the filters will be synchronized in time. The filter equations for a linear phase filter implementation can be summarized as follows:

$$s_i(n) = \sum_{j=-\left(N_{FB_i}-1\right)/2}^{\left(N_{FB_i}-1\right)/2} a_{FB_i}(j)\, s(n+j) \quad , \qquad (12)$$

where $a_{FB_i}(j)$ denotes the jth coefficient for the ith critical band filter. The filter order is normally odd for a linear phase filter.

Processing of speech through two filters in this type of filter bank is demonstrated in Fig. 10. A speech signal is shown, along with the outputs from two bandpass filters, one centered at 250 Hz, and one centered at 2500 Hz. Note that the power of the outputs varies depending on the type of sound spoken. This is the basic merit of the filter bank: certain filter outputs can be correlated with certain classes of speech sounds.

The filter outputs are normally processed using any of the power estimation methods previously discussed. The digital filter bank is most frequently used in systems that attempt to emulate auditory processing [53,54]. Recursive-in-time computations are particularly convenient for postprocessing in these applications.

The output of this analysis is a vector of power values (or power/frequency pairs) for each frame of data. These are usually combined with other parameters, such as total power, to form the signal measurement vector. The filter bank attempts to decompose the signal into a discrete set of spectral samples that contain information similar to what is presented to higher levels of processing in the auditory system. Because the analysis is largely based entirely on linear processing (as opposed to the nonlinear techniques discussed in Section 3.3.4), the technique is generally robust to ambient noise.

We conclude this section with one historical note. Long before computer hardware was capable of performing complex mathematical operations in real time, analog filter banks similar to the digital filter bank previously discussed were used in speech recognition. The filter banks were often built from discrete components and needed to be carefully tuned by adjusting resistors and capacitors. At that time, researchers dreamed of the days when speech recognition system parameters could be adjusted from software. The analog filter bank is one of the oldest approaches used in speech recognition. Ironically, the analog filter technique has generated some of the lowest cost implementations of speech recognizers to date.

3.3.2 The Fourier Transform Filter Bank

We have previously discussed the advantages in using non-uniformly spaced frequency samples. One of the easiest and most efficient ways to compute a non-uniformly spaced filter bank model of the signal is to simply perform a Fourier transform on the signal, and sample the transform output at the desired frequencies. The **Discrete Fourier Transform** (DFT) of a signal is defined as:

$$S(f) = \sum_{n=0}^{N_s-1} s(n)\, e^{-j\left(\frac{2\pi f}{f_s}\right)n} \quad , \qquad (13)$$

where f denotes the frequency in Hz, f_s denotes the signal sample frequency, and N_s denotes the window duration in samples.

The filter bank can be implemented by using Eq. (13) to sample the spectrum at the frequencies listed in Table 1. However, often the spectrum is oversampled at a finer resolution than that described in Table 1, and each output of the filter bank (a power spectral

magnitude) is computed as a weighted sum of its adjacent values:

$$S_{avg}(f) = \frac{1}{N_{os}} \sum_{n=0}^{N_{os}} w_{FB}(n) S(f + \delta f(f, n)) \quad , \qquad (14)$$

where N_{os} represents the number of samples used to obtain the averaged value, $w_{FB}(n)$ represents a weighting function, and $\delta f(f, n)$ represents some function that describes the frequencies in the neighborhood of f to be used in computing the average. Note that the averaging method presented in Eq. (14) is just one particular method of implementing a spectral smoothing function.

Averaging is often performed in the *mel* scale frequency domain if a DFT is used (since the added computational burden is minimal). Averaging also is usually performed in the log domain (log power values) rather than on spectral amplitudes. The benefit of using averaged values for spectral analysis is demonstrated in Fig. 11.

A **Fast Fourier Transform** (FFT) [55] also can be used as an alternate method of computing the spectrum of the signal. The FFT is a computationally efficient implementation of the DFT under the constraint that the spectrum is to be evaluated at a discrete set of frequencies that are multiples of f_s/N. These frequencies are referred to as orthogonal frequencies. The principal advantage of the FFT is that it is very fast: approximately $N \log N$ additions and $N \log N/2$ multiplications are required. (The DFT requires on the order of N^2 operations.) The principal disadvantage is that nonlinear frequency mappings, such as the filter bank in Table 1, must be adjusted to match the FFT orthogonal frequency constraints.

One additional processing step is often added. Based in part on our sketchy knowledge of human perception, we hypothesize that high amplitude areas of the spectrum are weighted more heavily in the auditory system than low amplitude regions. In noisy environments, noise often disproportionately degrades our estimates of the low amplitude areas of the spectrum. Stated another way, we are more confident of the reliability (and repeatability) of our estimates of the high amplitude areas of the spectrum.

For this reason, we often impose a limit on the dynamic range of the spectrum. This is depicted in Fig. 12. We refer to this lower limit as the **dynamic range threshold**. Rather than use noisy estimates of low amplitude regions of the spectrum, we simply clip, or discard, estimates below a certain threshold from the peak in the spectrum. For Fourier Transform-based techniques, this is straightforward to implement as a thresholding function on the spectral magnitude (measured in dB).

It is important that the spectral envelope be relatively flat before implementing such thresholding algorithms. Otherwise, useful low energy portions of the spectrum can be mistakenly eliminated. Recall that since the spectrum of the speech signal inherently drops 20 dB per decade, a threshold based on low frequency energies, where the peak to valley spectral amplitude difference is large, can easily remove useful signal energy at higher frequencies. Later, we will discuss more sophisticated methods for implementing thresholding of the spectrum based on parametric modeling techniques. For the moment, the reader might ponder the utility of letting the dynamic range threshold vary as a function of the background noise level in the spectrum or as a function of the local spectral peak to local background noise level.

3.3.3 Cepstral Coefficients

Since their introduction in the early 1970's, homomorphic signal processing techniques [27] have been of great interest in speech recognition. Homomorphic systems are a class of nonlinear systems that obey a generalized principle of superposition. Linear systems, such as those previously discussed, are a special case of a homomorphic system. The motivation for homomorphic processing in speech analysis is summarized in Fig. 13.

In speech processing, the homomorphic system we seek should have the following property:

$$D\left[\,[x_1(n)]^{\alpha} \bullet [x_2(n)]^{\beta}\right] = \alpha D[x_2(n)] + \beta D[x_2(n)] \ .$$

This is a superposition type operation with respect to multiplication, exponentiation, and addition. A logarithm function, of course, obeys the generalized superposition property.

Homomorphic systems were considered useful for speech processing [27] because they offered a methodology for separating the excitation signal from the vocal tract shape. Current approaches to speech recognition are primarily concerned with modeling the vocal tract characteristics. In the linear acoustics model of speech production, the composite speech spectrum, as measured by a Fourier transform, consists of the excitation signal filtered by a time-varying linear filter representing the vocal tract shape.

The process of separating the two components, often referred to as deconvolution, can be described as follows:

$$s(n) = g(n) \otimes v(n) \ ,$$

where $g(n)$ denotes the excitation signal, $v(n)$ denotes the vocal tract impulse response, and "$\otimes$" denotes convolution. The frequency domain representation of this process is:

$$S(f) = G(f) \cdot V(f) \ .$$

If we take the logarithm (complex) of both sides, we have:

$$\begin{aligned} Log(S(f)) &= Log(G(f) \cdot V(f)) \\ &= Log(G(f)) + Log(V(f)) \ . \end{aligned}$$

Hence, in the log domain, the excitation and the vocal tract shape are superimposed, and can be separated using conventional signal processing (in theory at least).

To compute the cepstrum, we first compute the log spectral magnitudes (averaged if necessary) from Eq. (14). Next we compute the inverse Fourier transform of the log spectrum:

$$c(n) = \frac{1}{N_s} \sum_{k=0}^{N_s-1} \log_{10}|S_{avg}(k)| e^{j\frac{2\pi}{N_s}kn} \ , \quad 0 \le n \le N_s - 1 \ . \tag{15}$$

$c(n)$ in Eq. (15) is defined as the **cepstrum**. We refer to cepstral coefficients computed via the Fourier transform (or analog filter bank) as **Fourier Transform-derived cepstral coefficients.**

Observe that $c(0)$ in Eq. (15) represents the average value of the spectrum, or the root mean square (rms) value of the signal. Initially, this term was an important part of the cepstral parameter vector. Later, it was observed that absolute power measures of the signal were somewhat unreliable[5], and use of $c(0)$ was deemphasized. Recently, however, since various alternative measures of power are explicitly added to the parameter vector in other stages of processing, $c(0)$ is no longer used. From this point forward, we will EXCLUDE this term from our discussion of the cepstral coefficient sequence.

5. Some systems [14] still use some form of absolute power along with various normalized power measures.

Equation (15) is also recognized as the inverse DFT of the log spectrum. This can be conveniently simplified by noting that the log magnitude spectrum is a real symmetric function. Hence, Eq. (15) can be simplified to:

$$c(n) = \frac{2}{N_s} \sum_{k=1}^{N_s} S_{avg}(I(k)) \cos \frac{2\pi}{N_s} kn \quad . \qquad (16)$$

$c(n)$ in Eq. (16) is normally truncated to an order much lower than N_s. $I(k)$ represents a mapping function that translates the integer k to the appropriate samples of S_{avg}. For efficiency, S_{avg} can also be computed using an oversampled FFT, rather than a non-uniformly spaced DFT.

We note that the cepstrum, as used in speech processing, is slightly different than the classical definition of the complex cepstrum found in the literature [27,28]. However, the definition presented here conveys all significant information needed in speech recognition.The cepstrum defined in Eq. (16) can be easily modified to be a *mel*-spaced cepstrum by sampling the Fourier Transform at appropriately spaced frequencies.

The cepstrum of two different speech signals is shown in Fig. 14. In Fig. 14(a) and Fig. 14(c), an unvoiced and a voiced speech waveform are shown, respectively. In Fig. 14(b) and Fig. 14(d), the corresponding cepstra are shown. The low order terms of the cepstrum correspond to short-term correlation in the speech signal (smooth spectral shape or vocal tract shape). The local maxima in the higher order terms in Fig. 14(d) demonstrate long term correlation, or periodicity, in the waveform (excitation information). The cepstrum in Fig. 14(b) of the unvoiced segment does not show any periodicity. In spectral analysis for speech recognition applications, normally only the low order terms ($n \leq 20$) are used.

Because the cepstrum is computed using a nonlinear operator, a logarithm function, it is generally believed to be sensitive to certain types of noise and signal distortions. While multiplicative noise processes can generally be dealt with, additive noise (such as background acoustic noise) can be troublesome. Cepstral parameters derived from high resolution spectral estimators, or parametric fits of the spectrum, are often preferred for applications in noisy environments.

3.3.4 Linear Prediction Coefficients

We now turn from Fourier Transform methods based on linear spectral analysis to a class of parametric modeling techniques that attempt to optimally model the spectrum as an autoregressive process. It is difficult to overstate the impact parametric models have made on speech processing since their introduction in the early 1970s [56,57]. By the late 1970s, almost every speech processing system used some sort of algorithm that parametrically fits the spectrum, whether for recognition, compression, or verification applications. Though parametric models today are less popular in recognition, they are still widely used in compression systems. Parametric models were the impetus for a transition to vastly more powerful statistical modeling techniques in speech recognition. In this section, we will discuss computation of a parametric model based on least mean squared error theory. This technique is known as linear prediction (LP).

The roots of linear prediction, as a least mean squared error algorithm can be traced to many diverse areas: system identification problems in modern control systems, time series analysis for economic applications, maximum entropy techniques, quantum physics, geophysics, adaptive filtering and spectral estimation in signal processing. Linear

prediction theory is well-documented in the literature. Some of the landmark academic texts include [31,32,57]. Here, we will (very) briefly review the mechanics of computing a linear prediction model, and then discuss the implications in speech recognition.

Given a signal, $s(n)$, we seek to model the signal as a linear combination of its previous samples. Let us define our signal model as:

$$s(n) = -\sum_{i=1}^{N_{LP}} a_{LP}(i)s(n-i) + e(n) \quad , \tag{17}$$

where N_{LP} represents the number of coefficients in the model (the order of the predictor), $\{a_{LP}\}$ are defined as the **linear prediction coefficients** (predictor coefficients), and $e(n)$ represents the error in the model (the difference between the predicted value and the actual measured value).

One obvious virtue of this model is that, if it is accurate, we should be able to predict future values of the signal based on our current set of measurements[6]. The error term should tell us something about the quality of our model (if the error is small, the model is accurate). It is also possible to show that a linear prediction model effectively models the spectrum of the signal as a smooth spectrum [31].

Equation (17) can be rewritten in Z-transform notation and shown to be a linear filtering operation:

$$E(z) = H_{LP}(z)S(z) \, ,$$

where $E(z)$ and $S(z)$ are the Z-transforms of the error signal and the speech signal, respectively, and

$$H_{LP}(z) = 1 + \sum_{i=1}^{N_{LP}} a_{LP}(i)z^{-i} \, ,$$

or,

$$H_{LP}(z) = \sum_{i=0}^{N_{LP}} a_{LP}(i)z^{-i} \, , \tag{18}$$

where $a_{LP}(0) \equiv 1$. $H_{LP}(z)$ is defined as the linear prediction inverse filter.

Under the constraint that we would like the mean-squared error to be as small as possible (seeking a solution that gives us the minimum error energy is reasonable), the coefficients (excluding $a_{LP}(0)$) of Eq. (18) can be obtained from the following matrix equation:

$$\bar{a}_{LP} = \underline{\Phi}^{-1}\bar{\phi} \, , \tag{19}$$

where,

$$\bar{a}_{LP} = \left[a_{LP}(1), ..., a_{LP}(N_{LP})\right]^{\dagger} \tag{20}$$

$$\underline{\Phi} = \begin{bmatrix} \phi_n(1,1) & \phi_n(1,2) & ... & \phi_n(1,N_{LP}) \\ \phi_n(2,1) & \phi_n(2,2) & ... & \phi_n(2,N_{LP}) \\ ... & ... & ... & ... \\ \phi_n(N_{LP},1) & \phi_n(N_{LP},2) & ... & \phi_n(N_{LP},N_{LP}) \end{bmatrix} , \tag{21}$$

$$\bar{\phi} = \left[\phi_n(1,0), \phi_n(2,0), ..., \phi_n(N_{LP},0)\right]^{\dagger} \, , \tag{22}$$

and,

$$\phi_n(j,k) = \frac{1}{N_s}\sum_{m=0}^{N_s-1} s(n+m-j)\,s(n+m-k) \quad . \tag{23}$$

The solution presented in Eqs. (19)-(23) is known as the Covariance Method. $\underline{\Phi}$ is referred to as a covariance matrix, and $\phi_n(j,k)$ is referred to as the covariance function for $s(n)$.

There are three basic ways to compute predictor coefficients: covariance methods based on the covariance matrix (also known as pure least squares methods), autocorrelation methods, and lattice (or harmonic) methods.

6. The naive reader will easily imagine uses for such models in predicting stock market prices. In fact, economic analysis was one of the earliest applications for such algorithms.

Good discussions of the differences in these approaches are given in [31,57]. In speech recognition, the autocorrelation method is almost exclusively used because of its computational efficiency and inherent stability. The autocorrelation method always produces a prediction filter whose zeroes lie inside the unit circle in the z-plane.

In the autocorrelation method, we modify Eq. (23) as follows:

$$\phi_n(j,k) = \phi_n(0,|j-k|) \quad , \tag{24}$$

or,

$$R_n(k) = \frac{1}{N_s}\sum_{m=0}^{N_s-1-k} s(n+m)\,s(n+m-k) \quad . \tag{25}$$

$R_n(k)$ is known as the autocorrelation function. This simplification results by constraining the evaluation interval to the range $[0, N-1]$, and assuming values outside this range are zero.

Because of this finite length constraint, it is important in the autocorrelation method to apply a window, such as that described in Eq. (5) to the signal. Normally, a Hamming window is used. Application of the window eliminates the problems caused by rapid changes in the signal at the edges of the window. In an overlapping analysis, it ensures a smooth transition from frame to frame of the estimated parameters.

This simplification allows the predictor coefficients to be computed efficiently using the Levinson-Durbin recursion [37]:

Initialization:

$$E_{LP}^{(0)} = R_n(0) \tag{26}$$

For $1 \le i \le N_{LP}$ **{**

$$k_{LP}(i-1) = -\frac{R_n(i) + \sum_{j=1}^{i-1} a_{LP}^{(i-1)}(j) R_n(i-j)}{E_{LP}^{(i-1)}} \tag{27}$$

$$a_{LP}^{(i)}(i) = k_{LP}(i-1) \tag{28}$$

For $1 \le j \le i-1$ **{**

$$a_{LP}^{(i)}(j) = a_{LP}^{(i-1)}(j) + k_{LP}(i-1)\, a_{LP}^{(i-1)}(i-j) \tag{29}$$

}

$$E_{LP}^{(i)} = (1 - k_{LP}(i-1)^2)\, E_{LP}^{(i-1)} \tag{30}$$

}

These equations compute the predictor coefficients with complexity proportional to N_{LP}^2, and allow the entire LP computation to be performed with a complexity of approximately $N_s N_{LP} + 3N_s + N_{LP}^2$.

The signal model is actually the inverse of $H_{LP}(z)$, and is given by:

$$S_{LP}(z) = \frac{G_{LP}}{H_{LP}(z)} \quad . \tag{31}$$

G_{LP} is the model gain, and is given by:

$$G_{LP} = \sqrt{E_{LP}^{(N_{LP})}} \quad . \tag{32}$$

Note that the gain term is also given by the expression:

$$G_{LP} = \sqrt{R_n(0)\prod_{i=1}^{N_{LP}} (1 - k_{LP}(i-1)^2)} \tag{33}$$

The gain term allows the spectrum of the LP model to be matched to the spectrum of the

original speech signal. The LP model computed from Eq. (19) is a normalized model (the values of the predictor coefficients are independent of the power of the signal).

There are three important observations to make about this form of the LP solution. First, the intermediate variables used in the computation, $\{k_{LP}\}$, are called **reflection coefficients**. They are bounded:

$$0 \le |k_{LP}(i-1)| \le 1, \quad \forall\ 1 \le i \le N_{LP} \ . \tag{34}$$

This is an extremely useful result for storage and compression applications involving LP models. For example, LP coefficients for speech applications can be compressed to as few as 30 bits/model without significant degradation [58]. LP coefficients can normally be stored in such as way that we can achieve an order of magnitude compression over the original speech data. This is an important consideration for speech recognition systems that must store large numbers of recognition models.

Second, the iterative solution computes the solution for all model orders $1 \le i \le N_{LP}$. This is convenient for signal processing applications that require estimation of the model order as part of the task. Normally, in speech recognition applications, the model order is a fixed system parameter.

Third, as the order increases, the model fit becomes better. Equation (30) represents the energy of the error. From this equation, we see that the error is monotonically decreasing as the order increases. The model itself attempts to match the overall spectrum as well as possible for the given order.

We demonstrate this fact in Fig. 15, where we show a speech spectrum, and two corresponding LP models. Note that as the order is increased, the model produces a better match of the original spectrum. With a low order, only the gross spectral shape (or trend) is captured. With a higher order, finer detail in the spectrum is represented.

As we observed in previous sections, the spectral model in low energy areas of the signal spectrum is often inaccurate. We would like to somehow impose a dynamic range threshold similar to that imposed in Fig. 12. There are several ways to do this in an LP model: a stabilized covariance method [59] that reduces the dynamic range in the spectrum, a perceptual-weighting method [60] that broadens the bandwidths of the LP model slightly, or a stabilized autocorrelation method [44] in which a small amount of noise is added to the autocorrelation function.

The latter of these approaches is simple and effective. The autocorrelation function of Eq. (25) is modified before the LP computation as follows:

$$\begin{aligned} R_{nw}(0) &= (1+\gamma_{nw})\,R_n(0) \\ R_{nw}(i) &= R_n(i) \qquad\qquad i>0 \ . \end{aligned} \tag{35}$$

The dynamic range threshold is normally specified in dB:

$$\gamma_{nw_{dB}} = 10 \log_{10}\gamma_{nw} \ . \tag{36}$$

A typical value of the dynamic range threshold is -10 dB.

This stabilization process is equivalent to adding uncorrelated white noise to the speech signal before LP analysis. The effect of this noise is to prevent the LP model from modeling sharp nulls (or zeroes) in the spectrum. This is demonstrated in Fig. 16. Observe that some distortion in the form of spectral smoothing is also introduced into the model at higher energy areas of the spectrum (broadening the bandwidths of the resonances of the LP model sometimes causes neighboring spectral resonances to collapse into one broad resonance).

Let us make one more important observation about the LP model. By adding power and fundamental frequency information to the LP coefficients, it is possible to reconstruct an audio version of the speech signal [56]. Listening to parametric descriptions of the speech signal, particularly speech recognition models, is very useful for diagnosing problems [61]. Some parametric transformations, such as cepstral coefficients, do not have a one-to-one mapping with the original LP data. Hence, it is somewhat harder to assess the validity of a parameter set.

Speech recognition systems historically first used LP parameters directly in the recognition process. Since then, more sophisticated transformations of these parameters have been devised. However, it is important to remember that generating an accurate LP model is an important first step in spectral analysis. Because LP analysis is a nonlinear operation, performance in noisy environments is sometimes problematic. For this reason, some systems still use a Fourier transform based filter bank analysis.

In Fig. 17, we summarize the LP modeling process by presenting LP models for several analysis conditions in the form of a spectrographic display [62]. Note that as the frame duration decreases (and the window duration is proportionately decreased) the temporal resolution in the spectrogram increases. Frame durations of 20 msec used to be most common in speech recognition systems. Recently, as the speech recognition research focus has shifted towards phonetic recognition, frame durations on the order of 10 msec have become common. The movement towards finer time resolution will continue as phonetic recognition technology matures.

3.3.5 LP-Derived Filter Bank Amplitudes

Obviously, we can combine the notion of a filter bank, such as that described in Section 3.3.1, with the LP model. **Linear prediction-derived filter bank amplitudes** are defined as filter bank amplitudes resulting from sampling the LP spectral model (rather than the signal spectrum) at the appropriate filter bank frequencies. The astute reader might ask: what is the benefit of this?

It has been argued that use of the LP model, or high resolution model as it is often referred to, gives more robust spectral estimates [57]. Often, the spectral smoothing inherent in the LP model provides more stable parameters to subsequent stages of the processing. However, as speech recognition and DSP technologies have progressed, the differences in these approaches are not as great as they once might have been.

How can we efficiently sample the spectrum given the LP model? A straightforward technique to compute filter bank amplitudes from the LP model involves direct evaluation of the LP model:

$$S_{LP}(f) = \frac{G_{LP}}{\sum_{i=0}^{N_{LP}} a_{LP}(i) e^{-j2\pi (f/f_s)}} , \qquad (37)$$

where f_s represents the sample frequency. This method requires on the order of $4p + 3$ multiply/accumulate operations per frequency sample. As described in Section 3.3.1, the spectrum is typically oversampled and averaged estimates are generated for actual filter bank amplitudes.

Another popular approach is to compute the power spectrum from the autocorrelation of the impulse response of $H_{LP}(z)$. The impulse

response of $H_{LP}(z)$ can be computed directly from the LP coefficients [32]:

$$R_{LP}(k) = \begin{cases} \sum_{m=0}^{N_{LP}-|k|} a_{LP}(m)\, a_{LP}(m+|k|) & |k| \le N_{LP}, \\ 0 & |k| > N_{LP}. \end{cases} \quad (38)$$

The power spectral density can be efficiently computed from the autocorrelation function by observing that the autocorrelation function is an even real function. Hence, its Fourier transform is real, and is given by:

$$S_{LP}(f) = R_{LP}(0) + 2\sum_{k=1}^{N_{LP}} R_{LP}(k)\cos\left(2\pi\frac{f}{f_s}k\right). \quad (39)$$

Equation (38) requires a total of approximately $N_{LP}^2 - (3/2)N_{LP}$ multiply/accumulate operations, while Eq. (39) requires N_{LP} multiply/accumulate operations per frequency sample.

With either approach, nonlinearly warped spectra can be easily implemented by appropriate choices of the filter bank sample frequencies. Also, even though the LPC model supplies a smoothed spectral fit, it is often still advantageous to oversample the spectrum so that sharp peaks in the frequency response will be accurately characterized by the filter bank (which tends to coarsely quantize the spectrum).

3.3.6 LP-Derived Cepstral Coefficients

Finally, we discuss our last signal measurement technique. Recall that in the last section, we leveraged the LP model to compute LP-derived filter bank amplitudes. Another logical step in this direction would be to use the LP model to compute cepstral coefficients. Again, the astute reader might wonder: can cepstral coefficients be computed directly from the LP model?

If the linear prediction filter is stable (and it is guaranteed to be stable in the autocorrelation analysis), the logarithm of the inverse filter can be expressed as a power series in z^{-1} [63]:

$$\begin{aligned} C_{LP}(z) &= \sum_{i=0}^{N_c} c_{LP}(i) z^{-i} \\ &= \log H(z) \\ &= \log\left(\frac{G_{LP}}{\sum_{j=0}^{N_{LP}} a_{LP}(j) z^{-j}}\right). \end{aligned} \quad (40)$$

We can solve for the coefficients by differentiating both sides of the expression with respect to z^{-1}, and equating coefficients of the resulting polynomials. This results in the following recursion [32,56,64]:

Initialization:

$$c_{LP}(1) = -a_{LP}(1) \quad (41)$$

For $2 \le i \le N_c$ {

$$c_{LP}(i) = -a_{LP}(i) - \sum_{j=1}^{i-1}\left(1 - \frac{j}{i}\right) a_{LP}(j)\, c_{LP}(i-j). \quad (42)$$

}

The coefficients $\{c_{LP}\}$ are referred to as **LP-derived cepstral coefficients**.

Historically, $c_{LP}(0)$ has been defined as the log of the power of the LP error [31]. For now, we note that since power will be dealt with as a separate parameter, there is no need to include it in the equations above. We can regard the cepstral model as a normalized model, much like an LP model, in which $c_{LP}(0) = \log 1 = 0$. We will discuss this issue in more detail later

(Section 5.2) when we consider comparison of signal models.

There is one minor complication in the cepstral coefficient recursion. We do not specify the number of cepstral coefficients, N_c, to compute. Since they are, in fact, an inverse Fourier transform of the impulse response of the LP model, and the LP model of the signal is an infinite impulse response filter, we can, in theory, compute an infinite number of cepstral coefficients. However, the number of cepstral coefficients computed is usually comparable to the number of LP coefficients: $0.75p \le N_c \le 1.25p$.

The cepstral coefficients computed with the recursion described above reflect a linear frequency scale. One drawback to the LP-derived cepstral coefficients is that we must work a little harder to introduce the notion of a nonlinear frequency scale. The preferred approach is based on a method used to warp frequencies in digital filter design. This method uses a very important transform in digital signal processing: the bilinear transform [27].

A bilinear transform is defined as:

$$f_{new} = 2\pi\frac{f}{f_s} + 2\tan^{-1}\left(\frac{\alpha_{bt}\sin\left(2\pi\frac{f}{f_s}\right)}{1-\alpha_{bt}\cos\left(2\pi\frac{f}{f_s}\right)}\right), \quad (43)$$

where α_{bt} is the frequency warping parameter. When $0.4 \le \alpha_{bt} \le 0.8$, the frequency warping of the bilinear transform is similar to the *mel* scale. This is demonstrated in Fig. 18. A common value of α_{bt} is 0.6.

We can use this transformation at one of several places in the LP-derived cepstral coefficient computation: on the autocorrelation function, on the predictor parameters, or on the LP-derived cepstral parameters. In [20], it was shown that postprocessing the cepstral coefficients was the most effective method (and the simplest).

Equation (43) describes a frequency domain procedure. If implemented using a sampled Z-transform approach, the computation would involve an inverse Fourier transform of the cepstral coefficients. Instead, we would prefer a direct recursive computation using the cepstral coefficients. Such a recursion fortunately exists [65].

This recursion can be viewed as a sequence of cascaded linear shift-invariant filtering operations, and can be implemented recursively as follows:

For $0 \le n \le N_c$ {

$$c_{bt}^{(n)}(0) = \alpha_{bt}\left[c_{bt}^{(n-1)}(0) - 0\right] + c_{LP}(N_c - n) \quad (44)$$

$$c_{bt}^{(n)}(1) = \alpha_{bt}\left[c_{bt}^{(n-1)}(1) - 0\right] + (1-a^2)\, c_{bt}^{(n-1)}(0) \quad (45)$$

For $2 \le k \le N_{bt}$ {

$$c_{bt}^{(n)}(k) = \alpha_{bt}\left[c_{bt}^{(n-1)}(k) - c_{bt}^{(n)}(k-1)\right] + c_{bt}^{(n-1)}(k-1) \quad (46)$$

}

}

where all initial conditions are zero. Since $c_{LP}(0) = 0$, processing can begin with $c_{LP}(1)$ at $n = 0$.

In this recursion, we iterate over all $c_{bt}(k)$ first, and update these values for each n (the second iteration). The results after N_c iterations are the final transformed coefficients. This recursion requires on the order of $N_c \times N_{bt}$ multiply/accumulate operations. It is about the same complexity as the LP solution.

Again, we need to consider truncation. The bilinearly transformed cepstral sequence, which is the result of a truncated cepstrum being processed through a nonlinear frequency translation, is inherently infinite in duration also. However, practically speaking, if the cepstral sequence is of finite duration, the resulting transformed sequence will asymptotically exponentially decay (poles inside the unit circle). Hence, it is possible to truncate the transformed sequence with little distortion. Normally, $N_{bt} \leq 1.25 N_c$.

In Fig. 19 we demonstrate the combined effects of cepstral analysis and bilinearly transformed coefficients. A speech spectrum for an $8\ kHz$ sampled signal is shown, along with its 10^{th} order LP model. The log magnitude spectrum of the cepstral coefficients is also shown for a 12^{th} order cepstral analysis. Similarly, a log magnitude spectrum for the bilinearly transformed cepstrum is shown. In this example 12 cepstral coefficients were converted to 16 bilinearly transformed coefficients.

The proper amount of compression for the bilinear transform is to some degree a function of the sample frequency. In several studies involving a $16\ kHz$ sample frequency [20,66], a compression factor of 0.6 was used. The effective bandwidth of the speech signal at a $16\ kHz$ sample frequency is small (the majority of the energy appears in the lower quarter of the frequency scale for sonorants).

We speculate that for a number of reasons, the bilinear transform is not as useful at a sample frequency of $8\ kHz$. First, the speech signal now occupies the majority of the available bandwidth. There is less "empty space" (the only available space is between approximately $3.0\ kHz$ and $4\ kHz$) to utilize in stretching the spectrum. Useful information can be deemphasized if extreme amounts of scaling are performed.

Second, the compression factor must more than likely be lowered to maintain the *mel* scale approximation. Yet, as this factor is lowered, the amount of compression performed decreases. This leads one to speculate that this transform may not be the best way to approximate the *mel* scale.

We have now discussed all major signal measurement techniques used in speech recognition systems today. Next, we will discuss how these parameters are smoothed and concatenated to form signal parameters.

IV. PARAMETER TRANSFORMS

In the previous section, we discussed several methods of computing absolute measurements. In this section, we will discuss the next step in the chain of operations depicted in Fig. 1: parameter transformations. Signal parameters are generated from signal measurements through two fundamental operations: differentiation and concatenation. The output of this stage of processing is a parameter vector containing our raw estimates of the signal. An overview of the operations that constitute the parameter vector construction is given in Fig. 20.

4.1 Differentiation

As computational power increased in the 1980s, the use of auxiliary measures of the speech spectrum in dynamic time warping systems became feasible. As part of a continuing trend to better characterize temporal variations in the signal, higher order time derivatives of signal measurements [14,18,21] were added to the signal model. The absolute measurements previously discussed can be thought of as zeroth order derivatives. Here, we investigate the

addition of the time derivatives of these measurements to our signal model.

In digital signal processing, there are several ways in which a first-order time derivative can be approximated. Three popular approximations are [27,67,68]:

$$\dot{s}(n) \equiv \frac{d}{dt}s(n) \approx s(n) - s(n-1) \ , \tag{47}$$

$$\dot{s}(n) \equiv \frac{d}{dt}s(n) \approx s(n+1) - s(n) \ , \tag{48}$$

$$\dot{s}(n) \equiv \frac{d}{dt}s(n) \approx \sum_{m=-N_d}^{N_d} m\ s\,(n+m) \ . \tag{49}$$

(Note that we have dropped superfluous normalization factors in these equations.) The first two equations are known as backward and forward differences, respectively. Eq. (49) represents a linear phase filter approximation to an ideal differentiator. This is often referred to as regression analysis.

The signal output from this differentiation process is denoted a **delta parameter**. The second-order time derivative can be similarly approximated by reapplying Eq. (49) to the output of the first-order differentiator. This is shown in Fig. 20. This output is often referred to as a **delta-delta** parameter. Obviously, we can extend this process to higher order derivatives.

We have seen Eq. (47) before in the form of a preemphasis filter in the spectral shaping portion of our system (see Eq. (2)). Recalling the primary purpose of the preemphasis filter was to amplify high frequency portions of the spectrum, we must be cognizant of the reality that differentiation is inherently a noisy process. Differentiation filters tend to amplify noise in the signal measurements. Often, it is desirable to compute derivatives of smoothed parameters, rather than the raw measurements, so that the noise level in the output measurement is decreased.

There are several popular ways to achieve this result. Regression analyses, as shown in Eq. (49), spline interpolation, and bandlimited differentiation are a few of the common techniques. We observe that since Eq. (49) computes differences symmetrically placed around the sample at time n, it is using a combination of N_d previous samples in each direction to compute the current value. Hence, some measure of smoothing is inherent in this calculation.

There are two trends emerging in the use of Eq. (49). Many systems today [2,66,113] use a simple first order difference: $N_d = 1$. These systems typically operate at frame durations in the range of $10 \text{ msec} \le T_f \le 20 \text{ msec}$. The range of time over which the derivative is computed is relatively small: $\Delta T_f \le 40 \text{ msec}$. A second group of systems [11,67] uses a larger number of terms: $5 \le N_d \le 7$. In these systems, $8 \text{ msec} \le T_f \le 10 \text{ msec}$. The period over which the derivative is computed is rather large: $56 \text{ msec} \le \Delta T_d \le 75 \text{ msec}$.

The frequency responses of several realizations of a differentiation filter are shown in Fig. 21. The low frequency portion of each filter is designed to approximate a linear function (a ramp function) that favors higher frequency information (indicative of temporal variation). Observe that as the order of the differentiator increases, the filter begins to deemphasize high frequencies, and introduces more ripple in the spectrum. The property of attenuating high frequencies is considered a form of noise reduction — beyond a certain point high frequency information is considered unreliable and needs to be attenuated.

4.2 Concatenation

The common thread throughout most of the measurement techniques we have discussed is the use of linear filtering to achieve parameter smoothing. Most systems postprocess the measurements in such a way that the operations can be easily explained in terms of linear filtering theory. In this section, we will generalize this notion in the form of a matrix operator.

Let us define a signal measurement matrix for a signal as follows:

$$\underline{X} = \begin{bmatrix} x(0,0) & x(0,1) & \dots & x(0,N_x-1) \\ x(1,0) & x(1,1) & \dots & x(1,N_x-1) \\ \dots & \dots & \dots & \dots \\ x(N_f-1,0) & x(N_f-1,1) & \dots & x(N_f-1,N_x-1) \end{bmatrix} \tag{50}$$

where $x(n,m)$ denotes the m^{th} signal measurement at frame n (or time $\left(n+\frac{1}{2}\right)T_f$), N_f denotes the total number of frames in the signal, and N_x denotes the total number of signal measurements for each frame.

The signal measurement matrix, $\underline{X}$, contains all measurements of a signal for all time. In many practical systems, the signal is processed frame by frame in real-time. Accumulation of the signal into one large matrix adds delay to the system. However, for research purposes, it is convenient to view the signal model as a matrix of measurements.

Note that the signal measurement matrix usually contains a mixture of measurements: power and a set of cepstral coefficients. N_x represents the dimension of the vector that is the composite of these measurements. From this point, we will consider these measurements as a group, rather than individually, and not refer to specific types of measurements. In some analyses, it is nevertheless useful that common measurements be grouped together in adjacent columns in $\underline{X}$ (to facilitate sub-matrix operations).

We will define two auxiliary matrices related to the parametric smoothing process. First, we define a matrix of lags (delays), denoted $\underline{\tau}$, that will represent a time delay (or advance) from the current time:

$$\begin{aligned} \underline{\tau} &= \begin{bmatrix} \tau(0,0) & \tau(0,1) & \dots & \tau(0,N_{\tau_0}-1) \\ \tau(1,0) & \tau(1,1) & \dots & \tau(1,N_{\tau_1}-1) \\ \dots & \dots & \dots & \dots \\ \tau(N_p-1,0) & \tau(N_p-1,1) & \dots & \tau(N_p-1,N_{\tau_p}-1) \end{bmatrix} \\ &= \left[\bar{\tau}_0\ \bar{\tau}_1\ \dots\ \bar{\tau}_{N_p-1}\right]^{\dagger} . \end{aligned} \tag{51}$$

where $\bar{\tau}_i$ denotes the i^{th} lag vector, and N_p denotes the total number of signal parameters. The lag vectors can be of different dimensions, depending on how many measurements will be used in each particular signal parameter computation (the lag matrix is actually a vector of vectors). Hence, we denote the dimension of each row by the term N_{τ_i}.

Next, we define a weighting matrix, $\underline{W}$, that holds the weights of filters to be applied to the measurements. These weights have a one-to-one correspondence with the lag matrix. The weight matrix is defined as follows:

$$\begin{aligned} \underline{W} &= \begin{bmatrix} w(0,0) & \dots & w(0,N_{\tau_0}-1) \\ \dots & \dots & \dots \\ w(N_p-1,0) & \dots & w(N_p-1,N_{\tau_p}-1) \end{bmatrix} \\ &= \left[\bar{w}_0\ \bar{w}_1\ \dots\ \bar{w}_{N_p-1}\right]^{\dagger} . \end{aligned} \tag{52}$$

where $\bar{w}_i$ denotes the i^{th} coefficient vector whose dimension is equal to the corresponding vector in $\underline{\tau}$.

We also define an indexing vector, $\underline{I}$, that for each row in $\underline{W}$ defines a corresponding column in $\underline{X}$:

$$\underline{I} = \left[I_0 \; I_1 \; \ldots \; I_{N_x - 1} \right] . \tag{53}$$

We define the process of filtering the measurement vector as a pseudo-convolution operator:

$$\underline{V} = \underline{X} \; ✪ \; \underline{W} \; ,$$

where the operator "✪" is defined as follows:

For $0 \le n \le N_f - 1$ {

  For $0 \le i \le N_x - 1$ {

$$V[n, i] = \sum_{j=0}^{N_{\tau_j} - 1} W(i, j) \; X\left[n + \tau(i, j), I_i \right] \tag{54}$$

  }

}

Equation (54) simply represents a sequence of linear filtering operations iterated over each element of the signal parameter vector for each frame of signal measurement data. This is expressed using a flexible indexing scheme to account for the fact that different types of features will require different filters. (This is more of an implementation issue than a conceptual issue.)

Note that through the use of the indexing array $\underline{I}$ we can derive multiple parameters from the same measurement (e.g., average power and delta-power from the same power value). Also, the coefficient matrix $\underline{W}$ can be used to realize all of the filtering operations previously discussed, including differentiation, averaging, and weighting. We refer to the operation described in Eq. (54) as **concatenation**: the creation of a single parameter vector per frame that contains all desired signal parameters.

Some parameters, such as power, are often normalized before the computation in Eq. (54). It is common to simply divide the power by the maximum value observed over an utterance (or subtract the log of the power). This approach has a drawback that one must wait until the utterance has been identified (or completed) before such a value is available.This delay is often unacceptable in real-time applications.

Peak power values can also be computed using recursive-in-time filtering approaches such as those described in Section 3.2. In this case, the frame-based power estimate is often postfiltered by an adaptive gain control circuit that attempts to dynamically monitor the peak power level. This adds delay to the system because the algorithm needs time to react to changes in the signal (and settle). Adaptation times for such estimators are typically on the order of 0.25 seconds [66].

Historically, when recognition systems were very simple, signal models often consisted of heavily smoothed parameters. "Noisy parameters", that is, parameters that amplified dynamics in the spectrum, were believed to be unreliable. With the emergence of Markov modeling techniques that provide a mathematical basis for characterizing sequential (or temporal) aspects of the signal, the reliance upon dynamic features has grown. Today, dynamic features are considered essential to developing a good phonetic recognition capability [69], because rapid change in the spectrum is a major cue in classification of a phonetic-level unit.

Differential parameters also gained popularity as researchers struggled to find signal models invariant to drastic changes in the speaker's behavior (often induced by the application) [2,70]. For example, in command and control applications, a speaker's acoustic data can change significantly as the speaker

encounters physical or mental stress related to the tasks at hand. These changes often manifest themselves more in the absolute spectra than the differential spectra. Parameters derived from differential spectral information are believed to make a system more invariant to these types of gross changes in a speaker's data.

Before we leave this section, let us discuss one particular form of parameter weighting used with cepstral coefficients. Early research into cepstral processing techniques suggested a means of performing linear filtering operations directly on the cepstral coefficients to enhance those portions of the cepstrum representing vocal tract information [71]. This technique has come to be known as liftering (a term coined because of the similarities to linear filter theory).

The liftering process is simple and is defined as follows:

$$c_{Lift}(m) = c(m)\, w_{Lift}(m) \ , \tag{55}$$

where

$$w_{Lift}(m) = 1 + \frac{N_c}{2} \sin \frac{\pi m}{N_c} \ . \tag{56}$$

Equation (55) describes a "time-domain" windowing operation (the time scale of the cepstrum is actually called quefrequency). Equation (56) describes the weighting (or window) function. At this point, we merely note that this is a static weighting function that can be applied directly to any set of cepstrum coefficients. In the next section, we will discuss a method of computing such a weighting in a statistically optimal manner.

V. STATISTICAL MODELING

In our last section on signal modeling, we turn our attention to the problem of statistical models for the signal parameters. In this section, we assume the signal parameters were generated from some underlying multivariate random process. We would like to learn, or discover, the nature of this process. Our approach will be to impose a model on the data, optimize (or train) the model, and then measure the quality of the approximation. The only information we will have about the process are its observed outputs, the signal parameters that have been computed. For this reason, the parameter vector output from this stage of processing is often called the **signal observations**. The collection of these vectors for the entire signal is referred to as the signal observation matrix.

This last processing step is, ironically, just the first step in statistical modeling in speech recognition. Often, this step is contained entirely within the speech recognition system [14,18]. The techniques described here only represent the most basic approaches. Speech recognition systems use extremely sophisticated statistical models — this is one of the fundamental functions of a speech recognizer. Nevertheless, the techniques presented here have been found to be useful in a wide variety of speech processing applications, and form the basis for the more sophisticated algorithms. An overview of the various types of transformations discussed in this section is given in Fig. 22.

5.1 Multivariate Statistical Models

As we have previously mentioned, in a typical set of heterogenous signal parameters, we mix quantities such as power and cepstral coefficients that have completely different numerical scales: the range and variance of the power term will be much larger than the range and variance of a cepstral coefficient. Variances of the time derivatives of the cepstral coefficients will be larger than the cepstral coefficients. If we compare two

parameter vectors using a simple operator such as a Euclidean distance, the result will likely be dominated by the terms with large amplitudes and variances, even though the true information may lie in the smaller amplitude parameters.

Similarly, if we consider a measurement such as a filter bank amplitude, it is easy to understand that filter bank amplitudes from adjacent bins are likely to be correlated with one another. The filter bank, in fact, is specifically designed to produce this type of correlation. This is similar to the way human hearing operates: a group of hair cells on the basilar membrane will respond to a given tone and produce correlated outputs.

We can illustrate the problem with performing direct comparisons in a vector space in which the dimensions have unequal variances with the simple two-dimensional example shown in Fig. 23(a). In the original coordinate system, the distance between points a and b is equal to the distance between points c and d (both are one unit). Yet, from a signal processing perspective, we would consider the former distance to be greater than the latter distance, because it is a larger percentage of the variance of the parameter. One cautionary note: the argument presented in Fig. 23 assumes that the observed variance is not due to a large noise component, and that it, in fact, represents meaningful variation in the parameter.

While the solution to the above problem of variance weighting is straightforward, elimination of correlation is more subtle. We would like to remove correlation from our measurements for two reasons. First, correlation implies redundancy. The actual number of "true" parameters required to describe the information might be much less than the number of measurements. We might be able to achieve some level of compression or reduction by selecting a subset of the features (or linear combinations of the features). Extraneous dimensions in signal processing problems are often the source of trouble (the problem becomes less well-conditioned). Second, we would like to use simple techniques to compare vectors. The presence of correlated parameters makes the development of an optimal statistical metric much more difficult.

5.1.1 Prewhitening Transformations [72]

There is a straightforward method of decorrelating parameters in a statistically optimal sense for a multivariate Gaussian process. Let us define a multivariate Gaussian probability distribution as:

$$\mathcal{P}(\bar{v}) = \mathcal{N}[\bar{v}, \bar{\mu}_v, \underline{C}_v] = \frac{1}{\sqrt{(2\pi)^{N_v}|C_v|}} e^{-\frac{1}{2}(\bar{x}-\bar{\mu}_v)\underline{C}_v^{-1}(\bar{x}-\bar{\mu}_v)^{\dagger}} . \quad (57)$$

We will assume that our parameters obey this type of statistical model (or stated another way, that our parameters can be modelled sufficiently accurately by such a process).

We can compute a linear transformation that will simultaneously normalize and decorrelate the parameters. Let us define a transformed vector, $\bar{y}$, as:

$$\bar{y} = \underline{\Psi}(\bar{v} - \bar{\mu}_v) , \quad (58)$$

where $\bar{v}$ denotes the input parameter vector, and $\bar{\mu}_v$ denotes the mean value of the input parameter vector. We define $\underline{\Psi}$ as a **prewhitening transformation** [18,*72*], based on the fact that we desire the output of this transformation to be an uncorrelated (or white) Gaussian random vector. To achieve this result, it can be shown that $\underline{\Psi}$ is given by:

$$\underline{\Psi} = \underline{\Lambda}^{-1/2}\underline{\Phi}^{\dagger} , \quad (59)$$

where $\underline{\Lambda}$ denotes a diagonal matrix of eigenvalues, and $\underline{\Phi}$ denotes a matrix of eigenvectors of the covariance matrix of $\bar{v}$.

A complete discussion of the significance of Eq. (59), the "true-meaning" of eigenvectors, and other deep mysteries of life related to Eq. (59) would take us too far afield from our task at hand. However, we cannot understate the importance of appreciating the need to statistically normalize parameters. This concept has been a recurring theme throughout modern speech recognition systems [73,74]. The eigenvalues and eigenvectors described above are the key to the whole computation, in that they describe a linear transformation of the input vector space to a new space in which normal Euclidean distances can be computed.

The eigenvalue and eigenvectors can be shown to satisfy the following relation:

$$\underline{C}_v = \underline{\Phi}\underline{\Lambda}\underline{\Phi}^{\dagger} \ , \tag{60}$$

where $\underline{C}_v$ is the covariance matrix for $\underline{v}$. Each element in $\underline{C}_v$, $C_v(i,j)$, can be computed as follows:

$$C_v(i,j) = \frac{1}{N_f}\sum_{m=0}^{N_f-1} (v_m(i) - \mu_v(i))\,(v_m(j) - \mu_v(j)) \ . \tag{61}$$

We have delayed disclosing the computation of the eigenvalues and eigenvectors (for good reason). This computation is algorithmically complex. While the procedure has a simple interpretation in linear algebra, it is somewhat of a nasty thing to program. We do point out that, since a covariance matrix is a real, symmetric matrix, and that covariance matrices for speech parameters usually are well-conditioned, the solution is normally well-behaved.

As pointed out in [75] (an excellent discussion on this topic), this is one of those problems in life best left to "canned software." One version we recommend, despite the fact that it is written in a very Fortranish-looking C, is a function based on the Jacobi transformation of a symmetric matrix. This function can be found in the widely distributed software package Numerical Recipes in C [75], and is named *jacobi*. This widely used software is very stable and has generally provided satisfactory performance on this task for many years.

There is one VERY important simplification of Eq. (60) that needs to be discussed. If the parameters are uncorrelated, then the covariance matrix in Eq. (60) reduces to a diagonal matrix. In this case, the transformation in Eq. (60) simplifies to a diagonal matrix:

$$\underline{\Psi} = \begin{bmatrix} \frac{1}{\sigma_{v(0)}} & 0 & \cdots & 0 \\ 0 & \frac{1}{\sigma_{v(1)}} & \cdots & 0 \\ \cdots & \cdots & \cdots & \cdots \\ 0 & 0 & 0 & \frac{1}{\sigma_{v(N_v-1)}} \end{bmatrix} \ , \tag{62}$$

where $\sigma_{v(i)}$ is the standard deviation of the i^{th} component of the parameter vector $\bar{v}$. This is readily recognized as normalizing the parameters by their standard deviations, thereby making each parameter count equally in the calculation. We alluded to this in our discussion of Fig. 23.

It has been observed over the years that certain parameter sets, namely cepstral coefficients, can be regarded (to an approximation) as uncorrelated [66,74]. This is convenient, because it significantly reduces the number of parameters one needs to estimate in the system. The so-called "variance-weighted

cepstral coefficients" are very popular in speech recognition systems today. Other parameters, such as filter bank amplitudes, display very strong correlations of the first off-diagonal components. Sometimes it is advantageous in such situations, in an effort to reduce computational noise in the system and to reduce the complexity of the system, to approximate the covariance matrix as such a banded matrix [18].

A noise reduction technique that is often incorporated in this context is a procedure in which we discard the least significant features. If we define N_v eigenvalues $\lambda_0, \lambda_1, ..., \lambda_{N_v-1}$ ordered in decreasing order, an important relationship can be shown to hold true:

$$\sum_{i=0}^{N_v-1} \lambda_i = trace\ C_v = \sum_{i=0}^{N_v-1} \sigma_i^2 \quad . \tag{63}$$

From this relationship, we see that the eigenvalues and the variance of the process are related.

We can define the amount of the variance accounted for by each eigenvalue/eigenvector pair as:

$$\zeta_i = \frac{\lambda_i}{\sum_{i=0}^{N_v-1} \lambda_i} \times 100\% \tag{64}$$

and the total percentage of the variance accounted for by the first N_y dimensions (one dimension corresponds to an eigenvector/ eigenvalue pair) as:

$$\zeta_{N_y} = \frac{\sum_{j=0}^{N_y-1} \lambda_j}{\sum_{i=0}^{N_v-1} \lambda_i} \times 100\% \quad . \tag{65}$$

In this case, the transformation matrix, $\underline{\Psi}$, is an $N_y \times N_v$ rectangular (but not square) matrix. Equations (64) and (65) are often used to guide decisions about how many dimensions to retain.

If a single Gaussian distribution is not a sufficient model, we note that we can model the data as a weighted sum (or mixture) of Gaussian distributions [74]. We will not discuss this issue in great detail here, but will observe that we can get an asymptotically good match to the parameter distribution with such a linear combination of Gaussian density functions.

We can gain some intuition into the prewhitening transformation from Fig. 24. Three eigenvectors from a transformation designed for telephone bandwidth speech are shown. These were computed for filter bank amplitude outputs from a *mel* scale filter bank. Note that each eigenvector attempts to model a different aspect of the speech spectrum. The first few eigenvectors of the transformation matrix often tend to model gross spectral characteristics of the channel (which is constantly changing in telecommunications applications). Note that this weighting function, when applied to filter bank amplitudes, can be viewed as a filtering operation in the time domain (frequency domain windowing is equivalent to time domain filtering).

Before we unilaterally invoke the power of this multivariate Gaussian model, remember the associated cost: we must learn (or train) this transformation matrix. Usually, this is done by collecting mean and covariance statistics across a large amount of speech data. Often this can be an art, because we must insure that statistics of the training data accurately match the underlying statistics of the process we are modeling. Insufficient training data sets (i.e.,

too little data or recording conditions that do not match the application environment) can lead to inaccurate transformations.

5.1.2 Vector Quantization [22]

In the previous section we alluded to the fact that parametric fits based on multivariate Gaussian statistics might not be appropriate. Recall that in a Gaussian process, all moments other than the first-order or second-order moments (mean and variance) are zero. For speech parameters, higher order moments are often significant[7], providing evidence that a Gaussian model of the source may not always be sufficient. We can perform a nonparametric fit of the data by simply hypothesizing a discrete probability distribution of arbitrary shape, and by forcing the system to learn the shape of this distribution. This does not come without an associated cost — we introduce another learning (or training) phase.

This type of discrete distribution model is referred to as a **vector quantizer** [22], a reference to the fact that the procedure can be regarded as a compression, or quantization, technique. One of the most convincing arguments for the use of such a technique is based on a model of speech production [77] that proposes the vocal tract shape as the key measurement. In this view of speech production, there are a small set of physically realizable vocal tract shapes (or elementary sounds) in a language. Hence, we should be able to model our continuous-valued vector with a finite set of vectors representing these unique vocal tract shapes. If we can use measurements that are normalized with respect to individual speaker physiology (vocal tract length, volume, etc.), we should be able to model all vocal tract shapes across all speakers with a small codebook.

Another equally compelling argument is based on rate distortion theory [22], and shows that we should be able to model our parameter vector using a finite number of discrete values with vanishingly small error. The main question is: how many symbols will be required? As we will see, the number of vectors used for such models in speech recognition typically ranges from 32 to 1024, depending on the task.

Let us define a vector quantizer as a composite of two items: an $N_{vq} \times N_y$ matrix, $\underline{Q}$, and a discrete probability distribution $p(\bar{q}_i)$. $\underline{Q}$ is referred to as the **vector quantization codebook**. Its rows are parameter vectors. $p(\bar{q}_i)$ is referred to as the *a priori* symbol probability distribution. Its elements ($0 \le i \le N_{vq} - 1$) are the probabilities of observing a given parameter vector (or row) in $\underline{Q}$. We will refer to this process of vector quantizing an input vector $\bar{y}$ as $Q[\bar{y}]$.

We also need to define a distance measure (or similarity measure): a means of determining the distance between two vectors. Let us defer this issue until the next section, and simply define a general distance measure:

$$D(\bar{y}_1, \bar{y}_2) = f(\bar{y}_1, \bar{y}_2) \quad . \tag{66}$$

We will see that a Euclidean distance is one common function used for Eq. (66).

The vector quantization process consists of two main tasks. First, as with the prewhitening transformation previously described, there is a training problem: how do we estimate $Q[\ ]$ such that the distortion introduced by replacing

7. Higher order moments have yet to make a significant impact on speech processing, though it is clear these calculations produce moments that are non-zero. Application of higher order spectral estimates and statistical estimates to speech recognition is still an area of active research. We conjecture that higher order moments might contain information about long term spectral behavior of the signal. This information could be useful for a wide variety of applications such as speaker normalization and speaker identification.

the input vector by a codebook vector is minimized? Second, there is the quantization problem: how do we estimate the probability of observing $\bar{y}$ given a codebook? This latter problem is essentially a pattern recognition problem — we seek to maximize $P(\bar{y}|Q)$.

The latter problem is relatively simple: we choose the index, i, according to a nearest neighbor rule:

$$i = argmin\,[D(\bar{y}, \bar{q}_j)] \qquad 0 \le j < N_{vq}\ , \tag{67}$$

and,

$$P(\bar{y}|Q) = p(\bar{q}_i)\ . \tag{68}$$

$P(\bar{y}|Q)$ can be estimated by computing the probability of each vector in the codebook can occur (this is usually estimated on a large training database).

The first problem is slightly more complicated. A training sequence is required — normally the same training database of speech used for recognition technology development. No closed-form solution exists for computing the optimal set of codebook vectors. Fortunately, several iterative techniques for finding a codebook exist.

The most popular of these is the K-MEANS algorithm [76]. The name alludes to the fact that this algorithm attempts to organize the data into K groups, and replace the data in each group with the mean, or centroid, of the group. This process is demonstrated in Fig. 25. Design parameters for this algorithm are the number of codebook vectors, N_{vq}, and some sort of termination condition. Here, we will use a limit on the maximum number of iterations, denoted M_{max}, and a threshold on the change in average distortion, $\Delta\varepsilon_{vq}$.

The algorithm is as follows:

Initialization:

Assign a set of N_{vq} initial vectors to $\bar{q}_i^{(0)}$

$\varepsilon^{(0)} = 1$

For $1 \le m \le M_{max}$ {

$\eta[j] = 0 \qquad 0 \le j < N_{vq}$

For $0 \le n < N_f$ {

$i = argmin\left[D(\bar{y}_n, \bar{q}_j^{(m-1)})\right] \qquad 0 \le j < N_{vq}$

$\bar{q}^{(m)}[j] = \bar{q}^{(m)}[j] + \bar{y}_n[j] \qquad 0 \le j < N_y$

$\eta[i] = \eta[i] + 1$

}

$\bar{q}_i^{(m)} = \bar{q}_i^{(m)} / \eta[i] \qquad 0 \le i < N_{vq}$

$\varepsilon^{(m)} = \varepsilon^{(m)} + Q[\bar{y}_n] \qquad 0 \le n < N_f$

if $\dfrac{\varepsilon^{(m)}}{\varepsilon^{(m-1)}} < \Delta\varepsilon_{vq}$, break

}

Termination:

$p(\bar{q}_i) = \dfrac{\eta[i]}{N_{db}} \qquad 0 \le i \le N_{vq}$

$\underline{Q}^{(m)}$ contains the final codebook upon termination.

The initialization step in this process is somewhat important. The initial guesses for cluster centers should span the entire data space. A simple iterative procedure [79] to select these centers is to search for N_{vq} initial vectors in the data that are a distance ε from each other. Initially, ε is set to some large number, and then slowly reduced until N_{vq} vectors satisfying the minimum distance constraint are found.

Since the K-MEANS algorithm is an iterative algorithm, and since we must provide guesses for the initial cluster centers, this algorithm is not guaranteed to converge to an optimal solution. The iteration procedure can

get trapped in local maxima and produce a suboptimal solution. However, in practice, on speech problems convergence is usually swift and strong. Even for large codebooks, convergence is often reached within 10 iterations.

The algorithm presented here computes the new cluster centers as the arithmetic average of the elements in the cluster. This is computationally and memory efficient, and makes the iteration proceed smoothly towards convergence. Other centroid recalculation strategies have been proposed [78] based on a min/max criterion. These algorithms, in practice, are generally comparable in performance to the one described here.

The quality of the codebook can be computed by averaging the distortion over the entire training database:

$$\varepsilon_{avg} = \frac{1}{N_f} \sum_{n=0}^{N_f-1} D(\bar{y}_n, Q[\bar{y}_n]) \quad . \tag{69}$$

This value is actually computed at each stage of the K-MEANS iteration. The average distortion usually decreases logarithmically with the size of the codebook, as shown in Fig. 26.

The quantization process actually becomes a search problem once the codebook has been computed. The nearest neighbor rule is a linear search, requiring N_{vq} distance comparisons per input vector. It is possible to reduce the search time by generating a structured codebook, though the codebook in this case is slightly sub-optimal. There are two popular variants of the K-MEANS algorithm that produce structured codebooks. The LBG algorithm [77] produces a codebook that is structured as an N-level tree (most often a binary tree). Procedures described in [80] produce lattice structured quantizers.

Recently, a new class of neural network based algorithms has emerged, referred to as Learning Vector Quantizers (LVQ) [81]. These algorithms combine the training and recognition problems into one massively parallel computational structure based on a neural network architecture. The LVQ approach has been shown to be theoretically capable of producing an optimal quantizer design. Nevertheless, this approach has not yet been shown to produce significantly better performance on speech problems.

5.2 Distance Measures

In an anticlimactic fashion, we will now discuss the problem that is at the root of speech recognition: the distance measure. It is most interesting to view this topic from an historical perspective. First, however, what is a distance measure? A distance measure should obey the following properties [82]:

(1) Nonnegativity:

$$D(\bar{x}_1, \bar{x}_2) > 0 \qquad x_1 \neq x_2$$

$$D(\bar{x}_1, \bar{x}_2) = 0 \qquad x_1 = x_2$$

(2) Symmetry:

$$D(\bar{x}_1, \bar{x}_2) = D(\bar{x}_2, \bar{x}_1)$$

(3) Triangle Inequality:

$$D(\bar{x}_1, \bar{x}_3) \leq D(\bar{x}_1, \bar{x}_2) + D(\bar{x}_2, \bar{x}_3) \quad .$$

A Euclidean distance measure is perhaps the most famous distance measure that satisfies these relations.

One of the first distance measures introduced into speech recognition was a measure based on minimum prediction error and spectral matching principles. This measure is known as the log-likelihood measure [83]. This measure computes the energy of the difference in the spectra of two LP parameter sets. It essentially evaluates the likelihood of the test data being generated from a statistical model based on the reference LP parameter set.

(Hence, it is often referred to as a probabilistic distance measure.) This distance measure is given by:

$$D(\bar{y}_1, \bar{y}_2) = \frac{a_{LP_1}{}^{\dagger} R_2 a_{LP_1}}{a_{LP_2}{}^{\dagger} R_2 a_{LP_2}} \quad . \tag{70}$$

R_2 represents the autocorrelation matrix used to generate the LP parameters for $\bar{x}_2$.

When performing a task such as vector quantization (or speech recognition) in which repeated comparisons of the test data will be made against the entire reference vector set, the denominator in Eq. (70) can be discarded (or computed only once). The numerator has a computationally efficient form [31]:

$$D(\bar{y}_1, \bar{y}_2) = \frac{\sum_{k=0}^{N_{LP}-1} R_1(k) R_2(k)}{a_{LP_2}{}^{\dagger} R_2 a_{LP_2}} \quad , \tag{71}$$

where

$$R_a(k) = \sum_{i=0}^{N_{LP}-k} a_{LP}(i) a_{LP}(i+k) \quad . \tag{72}$$

$R_a(k)$ represents the autocorrelation of the LP inverse filter impulse response.

The log-likelihood measure is readily seen to be an asymmetric distance measure (violating our previous definition of a distance measure). This asymmetry has not proven to be a significant problem however. The log-likelihood measure is not commonly used today. Nevertheless, it was a very important turning point in speech recognition research, because it initiated the adoption of a probabilistic framework for distance measures in speech recognition.

While the log-likelihood measure is well-suited to LP coefficients, what type of measure should we use for our general parameter vector? One of the most elegant derivations in statistical signal processing is a derivation of the Mahalanobis distance [72]. In this derivation, it is shown that the likelihood of a vector belonging to a multivariate Gaussian distribution can be expressed as a weighted Euclidean distance:

$$D(\bar{y}, \bar{\mu}) = (\bar{y} - \bar{\mu}) \underline{C}^{-1} (\bar{y} - \bar{\mu})^{\dagger} \quad , \tag{73}$$

where μ and $\underline{C}$ are the mean and covariance of the distribution. Obviously, if we operate on vectors that have been processed through a prewhitening transformation, the covariance matrix will be an identity matrix, and Eq. (73) degenerates to a plain squared Euclidean distance.

For this reason, Euclidean distances are the most common distance measure used today. Part of the reason for this is that many parameter sets used are based on implicitly decorrelated parameters, such as cepstral coefficients. Also, speech recognition systems today have evolved to invoke these types of transformations implicitly [73,74].

In many applications, such as vector quantization, it is possible to use a factored form of the Euclidean distance:

$$\begin{aligned} D(\bar{y}_1, \bar{y}_2) &= \left\| \bar{y}_1 - \bar{y}_2 \right\|^2 \\ &= (\bar{y}_1 - \bar{y}_2)\,(\bar{y}_1 - \bar{y}_2)^{\dagger} \\ &= \left\| \bar{y}_1 \right\|^2 + \left\| \bar{y}_2 \right\|^2 - 2\,(\bar{y}_1 \cdot \bar{y}_2) \quad . \end{aligned} \tag{74}$$

We see a Euclidean distance is the sum of the magnitudes of the vectors minus twice the dot product. Suppose we wish to vector quantize the input vector, $\bar{y}_1$. Let $\bar{y}_2$ denote each entry in the codebook (against which the input must be compared). The first term is constant with respect to each codebook entry, and can be discarded. The second term is a scalar, and can be added to the result of the third term. If each codebook entry has the same magnitude, the

second term can be discarded also.

The third term is a dot product and must be evaluated once per codebook entry. This is called the dot-product form of the Euclidean distance. This type of calculation can be used in most speech processing applications, which include vector quantization and speech recognition[8].

We have now completed our discussion of techniques for signal modeling in speech recognition. We have shown that a signal model can be constructed from a sequence of three operations: signal measurement, parameter smoothing, and statistical modeling. In the next section, we will discuss how these concepts are put to work in state of the art speech recognition systems, and comment on the relative merits of these approaches.

VI. PRACTICAL EXAMPLES

There are, needless to say, a large number of combinations and permutations of the signal models we have discussed in use today. Let us begin our discussion by simply enumerating some state-of-the-art (perhaps modern day is a better way to phrase this) speech recognition systems, along with the signal modeling techniques used. An overview of this data is given in Table 2. The table is divided into four sections: affiliation (for reference purposes); signal measurements, signal parameters, and statistical models. We have discussed each of these topics in previous sections of this paper. Note that a list of abbreviations and their meanings is contained at the end of the table. Since this table is fairly lengthy, we will review its contents first, and then draw some conclusions about the data.

6.1 Table Overview

The affiliation section of the table is supplied merely as an identifier. Institutions are listed in alphabetical order. The reference associated with the entry usually contains a sufficient amount of detail about the signal model used in the most recently published recognition system by the institution. Every attempt has been made to keep the reference current, so that the data in the table reflects the current state of the particular system. Many of the systems in Table 2 have evolved over several years. In some cases, multiple entries for a given affiliation are shown, usually to contrast different types of recognition technology being explored at the same institution.

The second column contains a brief overview of the type of application being pursued, at least in terms of vocabulary size. Obviously, there are other equally important dimensions to the problem. “Small” refers to a speech recognition task using a small vocabulary, usually less than 100 words. Continuous digit recognition and recognition of spoken letters in English fall into this category. Similarly, “Medium” refers to tasks on the order of 1000 words, and “Large” refers to tasks greater than 5000 words[9].

We also attempt to quantify the application in terms of the acoustic environment. “Office” refers to a system developed on a database collected in either a normal office environment (typically about 70 dB SPL) or an

8. A consultant from a famous supercomputing company once told me that “everything in life reduces to one of two operations: a dot product or a vector multiply/add.” He had found through his extensive experience optimizing code for supercomputing that 99% of the time the core operations required could be shown to be one of these two.

9. We readily admit this is an oversimplification of the problem. Confusability of the words and the complexity of the language model are equally important dimensions, but more difficult to concisely quantify. There are some major differences between signal models in systems used in “small” and “large” vocabulary applications.

acoustically-treated room (such as an anechoic chamber). "Telecom" refers to technology developed for telecommunications applications using standard analog telephone lines. "Mil." refers to military applications that often involve a wide range of ambient noise and speaking styles.

The signal measurement section is split into five parts. First, we show the sample frequency of the system. Though most systems today have software-selectable sample frequencies, this entry represents the sample frequency of the experimentation database in the corresponding reference. It is provided for comparison purposes. Sample frequency is most often dictated by the application (systems for telecommunications applications must operate at or below 8 kHz).

The next three columns specify the spectral analysis conditions: a_{pre} is the preemphasis filter constant for a first order filter; "Frame Dur." is the frame duration of the analysis; "Wind. Dur." is the analysis window duration. We do not explicitly show the type of window used on the signal before spectral analysis, because all of the systems presented here use some form of a generalized Hanning window (most use a Hamming window, while a few use a Hanning window).

The last column in this section, titled "Spectral Analysis," refers to the sequence of operations involved in generating the measurements. For example, under the entry "AT&T [7]," LP(8) and CEP(12) indicate that an LP analysis of order 8, followed by a cepstral analysis of order 12, was used to generate the cepstral signal measurements. Most systems that use LP-derived cepstral coefficients will have multiple entries in this column, to indicate that LP and Cepstral analyses were performed.

The next section, comprising the 8th column in Table 2, contains the elements of the signal parameter vector. Here, we simply list the salient features (as they might be concatenated in the vector) of the type of analysis used. These were discussed in Section III. Usually, an entry consists of a set of absolute measurements, such as "Mel-Cep." (denoting *mel* warped cepstral coefficients), and time derivatives of these absolute measurements, such as "D-Cep." (which denotes the derivative of the cepstral parameters). See the abbreviations at the end of the table for an explanation of all the terms.

The last section of the table (cols. 9 and 10) contains a description of the statistical models used in the systems. The type of speech recognition technology used with the signal model is shown mainly for reference purposes[10]. The term "VQ" refers to vector quantization. The term "Variance" refers to the variance-weighting form of the Prewhitening transformation. The term "PT" refers to a prewhitening transformation in which the full rank of the matrix is used. The term "MS-VQ" refers to multi-stage vector quantization: an approach in which a separate codebook is maintained for each type of signal parameter (often there are three codebooks: one for absolute measurements, one for time derivative measurements, and one for power measurements).

6.2 Comments

There are several conclusions we can draw from the agglomeration of data presented in Table 2. First, Neural Network (NN) based systems tend to use filter bank amplitudes directly. We will avoid elaborating on this

10. These technologies have not been discussed in this paper. There are two main classes of technology referenced in this table: Hidden Markov Model (HMM) and Neural Network (NN). See [27] for more information on this topic.

point — it would take us too far afield. It suffices to say that the NN systems are attempting to develop models that emulate human hearing. Filter bank amplitudes are perhaps the simplest type of stimulus to present to such a system that will achieve this goal and keep the system complexity low. Readers interested in this topic are encouraged to pursue more rigorous discussions in [87,91].

Second, as previously mentioned several times in this paper, cepstral coefficients are by far the dominant acoustic measurement. For example, twenty-one out of the thirty-one systems in Table 2 use some form of cepstral coefficients. This ratio is even higher (21 out of 26) if we exclude Neural Net based systems.

Third, FFT-derived *mel*-scaled cepstral coefficients are the most common form of cepstral analysis used. LP-derived cepstral coefficients are used by only a third of the systems using cepstral analysis. There is a definite preference towards using *mel* scaling. Ironically, two institutions that are notable advocates of cepstral coefficients are the only institutions to use their respective techniques: liftered cepstral coefficients and bilinearly transformed cepstral coefficients.[11]

A fourth observation, somewhat beyond the scope of this paper, is that systems performing large vocabulary speech recognition tend to use discrete density approaches based on vector quantization, while systems performing small vocabulary speech recognition in harsh environments tend to use some sort of prewhitening filter. The most common form of vector quantization today is the multi-stage codebook, used in conjunction with cepstral and time-derivatives of the cepstral coefficients. The most common form of prewhitening filter is the degenerate case: variance-weighted coefficients. We find few systems actually using a fully populated covariance matrix.

The popularity of FFT-based spectral analysis continues to be based on the FFT's immunity to noise. We find it somewhat surprising that a large percentage of systems today do not rely on LP analysis for spectral analysis. LP analysis was almost exclusively used in the 1970's and early 1980's. Since then, it seems the trend is towards the FFT based analysis. We speculate that this is due to the ease with which the *mel* scaling can be imposed.

In the systems reviewed in Table 2, the use of time differentiation to postprocess signal measurements can be classified into two groups. Most systems use a simple first order difference. Several systems, most notably those in [7] and [11], use a five frame regression analysis. It is argued that this provides a smoother, more stable representation of the parameter.

Very little comprehensive data exists on comparative analyses of signal modeling in speech recognition. There are two major reasons for this. First, only recently have large scale speech recognition experiments become feasible. Databases are now large enough to support statistically significant comparisons; computers are now fast enough to do parametric evaluations. Unfortunately, software technology lags: many research organizations are only able to simulate subsets of many of the competing approaches (and have not researched the others extensively). Convincing comparative data on large speaker independent continuous speech recognition tasks is simply not currently available. Two of the highly referenced works in this area

11. These two institutions have also consistently delivered high performance systems with their signal models, which makes the lack of adoption of these techniques interesting.

are [23,84]. Other, more recent studies include [113,114].

VII. SUMMARY

We have presented several popular signal analysis techniques in a common framework that emphasized the importance of accurate spectral analysis and statistical normalization. When viewed in this common framework, the differences amongst these competing approaches seems small when compared to the enormous challenges we still face in the speech recognition problem. All approaches share some important basic attributes: time-derivative information, perceptually-motivated transformations, and parameter normalization.

The survey of contemporary systems demonstrated the fact that FFT-derived cepstral coefficients seem to be dominating the field. LP analysis, once the cornerstone of speech recognition, is now relegated to a secondary role. A signal parameter vector consisting of cepstral coefficients, the first derivative of the cepstral coefficients, power, and the derivative of the power has become a *de facto* standard. Variance-weighting of this parameter vector is the most popular normalization technique.

It will be the subject of further research to quantify the differences in these approaches in a reasonable recognition task. There still remain some important questions to be quantified: robustness to noise? invariance to sample frequency? invariance to recognition task? It is interesting to note that despite the seemingly vast algorithmic differences in these approaches, many of these approaches have enjoyed widespread success. Often, the significant differences in the recognition systems lie in details beyond the signal model.

This not to say that the problem of signal modeling is solved. Performance of current speech recognition systems is still far below human performance. For example, on digit recognition tasks, where the vocabulary is small and a premium is placed on acoustic modeling, state-of-the-art performance is still at least two orders of magnitude below human performance on the same task [73,115]. In adverse ambient environments, such as analog telecommunications systems or cellular telephony in an automobile, the performance gap between humans and machines is even greater.

As mentioned at the beginning of this paper, we have also steered clear of such topics as robustness in noise. We have presented some simple approaches for dealing with noise that generally work equally well for clean and noisy environments. We have not presented techniques specifically designed to improve robustness in adverse conditions — this is a topic unto itself. Recently, several promising algorithms have appeared for improving signal models in noisy environments [116]. As speech recognition systems are being moved from the laboratory to the field, such practical problems are receiving increasing attention. Perhaps this paper will motivate a future tutorial on the topic. Clearly, robustness-in-noise issues strongly interact with signal model design.

Finally, a major driving force today in signal model design is the minimization of the number of degrees of freedom in the system. Because speech recognition systems today have a large number of free variables (more than 10,000 variables is common), insufficient amounts of training data are a very real problem. One thing we have learned over the years: badly trained parameters are often cited as the major contributor to bad performance.

Hence, approaches that minimize the number of parameters, like variance-weighting, are preferred over approaches that are statistically optimal, e.g., prewhitening transformations, but require large amounts of training data.

VIII. REFERENCES

1. D.S. Pallet, "Speech Results on Resource Management Task," in *Proceedings of the February 1989 DARPA Speech and Natural Language Workshop*, Morgan Kaufman Publishers, Inc., Philadelphia, PA, USA, pp. 18-24, February 1989.
2. D. Paul, "The Lincoln Robust Continuous Speech Recognizer," in *Proceedings IEEE International Conference on Acoustics, Speech, and Signal Processing*, pp. 556-559, Glasgow, Scotland, May 1989.
3. J.G. Wilpon, R.P. Mikkilineni, D.B. Roe, and S. Gokcen, "Speech Recognition: From the Laboratory to the Real World," *AT&T Technical Journal*, vol. 69, no. 5, pp. 14-24, October 1990.
4. J.G. Wilpon, D.M. DeMarco, R.P. Mikkilineni, "Isolated Word Recognition Over the DDD Telephone Network - Results Of Two Extensive Field Trials," in *Proceedings IEEE International Conference on Acoustics, Speech, and Signal Processing*, pp. 55-57, New York, NY, USA April 1988.
5. B. Wheatley and J. Picone, "Voice Across America: Toward Robust Speaker Independent Speech Recognition For Telecommunications Applications", *Digital Signal Processing: A Review Journal*, vol. 1, no. 2, pp. 45-64, April 1991.
6. J. Picone, "The Demographics of Speaker Independent Digit Recognition", in *Proceedings IEEE International Conference on Acoustics, Speech, and Signal Processing*, pp. 105-108, Albuquerque, New Mexico, USA, April 1990.
7. J.G. Wilpon, C.H. Lee, and L.R. Rabiner, "Improvements in Connected Digit Recognition Using Higher Order Spectral and Energy Features," in *Proceedings IEEE International Conference on Acoustics, Speech, and Signal Processing*, pp. 349-352, Toronto, Ontario, Canada, May 1991.
8. T. Matsuoka and K. Shikano, "Robust HMM Phoneme Modeling For Different Speaking Styles," in *Proceedings IEEE International Conference on Acoustics, Speech, and Signal Processing*, pp. 265-268, Toronto, Ontario, Canada, May 1991.
9. Y.T. Lee, "Information-Theoretic Distortion Measures for Speech Recognition: Theoretical Considerations and Experimental Results," *IEEE Transactions on Signal Processing*, vol. 39, no. 2, pp. 330-335, February 1991.
10. Y.T. Lee and D. Kahn, "Information-Theoretic Distortion Measures for Speech Recognition: Theoretical Considerations and Experimental Results," in *Proceedings IEEE International Conference on Acoustics, Speech, and Signal Processing*, pp. 785-788, Albuquerque, New Mexico, USA, April 1990.
11. S. Furui, "On the Use of Hierarchical Spectral Dynamics in Speech Recognition," in *Proceedings IEEE International Conference on Acoustics, Speech, and Signal Processing*, pp. 789-792, Albuquerque, New Mexico, USA, April 1990.
12. D. Mansour and B.H. Juang, "A Family of Distortion Measures Based Upon Projection Operation for Robust Speech Recognition," *IEEE Transactions on Acoustics, Speech, and Signal Process-*

ing, vol. 37, no. 11, pp. 1659-1671, November 1989.

13. Y. Tohkura, "A Weighted Cepstral Distance Measure For Speech Recognition," *IEEE Transactions on Acoustics, Speech, and Signal Processing*, vol. 35, no. 10, pp. 1414-1422, October 1987.
14. G.R. Doddington, "Phonetically Sensitive Discriminants for Improved Speech Recognition," in *Proceedings IEEE International Conference on Acoustics, Speech, and Signal Processing*, pp. 556-559, Glasgow, Scotland, May 1989.
15. M.J. Hunt and C. Lefebvre, "A Comparison of Several Acoustic Representations for Speech Recognition with Degraded and Undegraded Speech," in *Proceedings IEEE International Conference on Acoustics, Speech, and Signal Processing*, pp. 262-265, Glasgow, Scotland, May 1989.
16. B.H. Juang, L.R. Rabiner, and J.G. Wilpon, "On the Use of Bandpass Liftering in Speech Recognition," *IEEE Transactions on Acoustics, Speech, and Signal Processing*, vol. 35, no. 7, pp. 947-954, July 1987.
17. V.N. Gupta, M. Lennig, and P. Mermelstein, "Integration Of Acoustic Information In A Large Vocabulary Word Recognizer," in *Proceedings IEEE International Conference on Acoustics, Speech, and Signal Processing*, pp. 697-700, Dallas, Texas, USA, April 1987.
18. E.L. Bocchieri and G.R. Doddington, "Frame Specific Statistical Features for Speaker-Independent Speech Recognition," *IEEE Transactions on Acoustics, Speech, and Signal Processing*, vol. 34, no. 4, pp. 755-764, August 1986.
19. P.K. Rajasekaran, G.R. Doddington, J. Picone, "Recognition of Speech Under Stress and in Noise," in *Proceedings IEEE International Conference on Acoustics, Speech, and Signal Processing*, pp. 733-736, Tokyo, Japan, April 1986.
20. K. Shikano, "Evaluation of LPC Spectral Matching Measures for Phonetic Unit Recognition," TM No. CMU-CS-86-108, Computer Science Department, Carnegie-Mellon University, Pittsburgh, PA, US, 15213, February 3, 1986.
21. S. Furui, "Speaker-Independent Isolated Word Recognition Using Dynamic Features of the Speech Spectrum," *IEEE Transactions on Acoustics, Speech, and Signal Processing*, vol. 34, no. 1, pp. 52-59, February 1986.
22. J. Makhoul, S. Raucos, and H. Gish, "Vector Quantization In Speech Coding", in *Proceedings of the IEEE*, vol. 73, no. 11, pp. 1551-1588, November 1985.
23. N. Nocerino, F.K. Soong, L.R. Rabiner, and D.H. Klatt, "Comparative Study Of Several Distortion Measures For Speech Recognition," in *Proceedings IEEE International Conference on Acoustics, Speech, and Signal Processing*, pp. 25-28, Tampa, Florida, USA, March 1985.
24. L.R. Rabiner, K.C. Pan, and F.K. Soong, "On the Performance of Isolated Word Speech Recognizers Using Vector Quantization and Temporal Energy Contours," *AT&T Bell Labs Technical Journal*, vol. 63, no. 7, pp. 1245-1260, September 1984.
25. L.R. Rabiner, S.E. Levinson, M.M. Sondhi, "On the Application of Vector Quantization and Hidden Markov Models to Speaker-independent, Isolat-

ed Word Recognition," *Bell System Technical Journal*, vol. 62, no. 4, pp. 1075-1105, April 1983.

26. S.B. Davis and P. Mermelstein, "Comparison of Parametric Representations of Monosyllabic Word Recognition in Continuously Spoken Sentences," *IEEE Transactions on Acoustics, Speech, and Signal Processing*, vol. 28, no. 4, pp. 357-366, August 1980.
27. A.V. Oppenheim and R.W. Schafer, *Digital Signal Processing*, Prentice-Hall, Englewood Cliffs, New Jersey, USA, 1975.
28. J.R. Deller, J.G. Proakis, J.H.L. Hansen, *Discrete Time Processing of Speech Signals*, MacMillian Publishing Co., New York, New York, USA, 1993.
29. L. Rabiner and B.H. Juang, *Fundamentals of Speech Recognition*, Prentice-Hall, Englewood Cliffs, New Jersey, USA, 1993.
30. J.G. Proakis, *Digital Communications*, McGraw-Hill, 2nd Ed., New York, New York, USA, 1989.
31. J. Markel and A.H. Gray, Jr., *Linear Prediction of Speech*, Springer-Verlag, New York, New York, USA, 1980.
32. L.R. Rabiner and R.W. Schafer, *Digital Processing of Speech Signals*, Prentice-Hall, Englewood Cliffs, New Jersey, USA, 1978.
33. A.M. Noll, "Problems of Speech Recognition in Mobile Environments," in *Proceedings of the International Conference on Spoken Language Processing*, pp. 1133-1136, Kobe, Japan, November 1990.
34. Y. Nakadai and N. Sugamura, "A Speech Recognition Method For Noise Environments Using Dual Inputs," in *Proceedings of the International Conference on Spoken Language Processing*, pp. 1141-1144, Kobe, Japan, November 1990.
35. S. Tamura, "An Analysis of a Noise Reduction Neural Network," in *Proceedings IEEE International Conference on Acoustics, Speech, and Signal Processing*, pp. 2001-2004, Glasgow, Scotland, May 1989.
36. J.L. Hieronymus, D. McKelvie, and F.R. McInnes, "Use of Acoustic Sentence Level and Lexical Stress in HSMM Speech Recognition," in *Proceedings IEEE International Conference on Acoustics, Speech, and Signal Processing*, pp. 225-227, San Francisco, California, USA, March 1992.
37. T. Matsui and S. Furui, "A Text-Independent Speaker Recognition Method Robust Against Utterance Variations," in *Proceedings IEEE International Conference on Acoustics, Speech, and Signal Processing*, pp. 377-380, Toronto, Ontario, Canada, April 1991.
38. W. Hess, *Pitch Determination of Speech Signals*, Springer-Verlag, New York, New York, USA, 1983.
39. P. Papamichalis, *Practical Approaches To Speech Coding*, Prentice-Hall, Inc., Englewood Cliffs, New Jersey, USA, 1987.
40. B. Gold and L.R. Rabiner, "Parallel Processing Techniques for Estimating Pitch Periods of Speech in the Time Domain," *Journal of the Acoustical Society of America*, vol. 46, no. 2, pt. 2, pp. 442-448, August 1969.
41. R.S. Sukkar, J.L. LoCicero, and J. Picone, "Design and Implementation of a Parallel Processing Based Pitch Detector," *IEEE Journal on Selected Areas of Communications*, vol. 6, no. 2, pp. 441-451, February 1988.
42. J. Campbell and T.E. Tremain, "Voiced/ Unvoiced Classification of Speech with Applications to the U.S. Govern-

ment LPC-10E Algorithm," in *Proceedings IEEE International Conference on Acoustics, Speech, and Signal Processing*, pp. 473-476, Tokyo, Japan, April 1986.

43. V.C. Welch,T.E. Tremain, and J.P. Campbell, Jr., "A Comparison of U.S. Government Standard Voice Coders," *IEEE Military Communications Conference Record*, pp. 269-273, September 1989.
44. J. Picone, G.R. Doddington, and B.G. Secrest, "Robust Pitch Detection in a Noisy Telephone Environment," in *Proceedings IEEE International Conference on Acoustics, Speech, and Signal Processing*, pp. 1442-1445, Dallas, Texas, USA, April 1987.
45. A.M. Noll, "Cepstrum Pitch Determination," *Journal of the Acoustical Society of America*, vol. 41, no. 2, pp. 293-309, February 1967.
46. G. von Békésy, *Experiments in Hearing*, McGraw-Hill Book Company, New York, New York, USA, 1960.
47. J. Picone, "Analytic Signal Processing," Ph.D. Dissertation, Illinois Institute of Technology, Chicago, Illinois, USA, December 1983.
48. K. Ogata, *Modern Control Engineering*, Prentice-Hall, Englewood Cliffs, New Jersey, USA, 1970.
49. J.O. Pickles, *An Introduction to the Physiology of Hearing*, Academic Press, New York, New York, USA, 1988.
50. A.R. Møller, *Auditory Physiology*, Academic Press, New York, New York, USA, 1983.
51. D. O'Shaughnessy, *Speech Communication: Human and Machine*, Addison Wesley, New York, New York, USA, 1987.
52. E. Zwicker and E. Terhardt, "Analytical expressions for critical-band rate and critical bandwidth as a function of frequency," *Journal of the Acoustical Society of America*, vol. 68, no. 5, pp. 1523-1525, December 1980.
53. J.B. Allen, "Cochlear Modeling," *IEEE ASSP Magazine*, vol. 3, no. 3, pp. 3-29, September 1985.
54. S. Seneff, "A Joint Synchrony/Mean-Rate Model of Auditory Speech Processing," *Journal of Phonetics*, vol. 16, no. 1, pp. 55-76, January 1988.
55. O.E. Brigham, *The Fast Fourier Transform*, Prentice-Hall, Englewood Cliffs, New Jersey, USA, 1974.
56. B.S. Atal and S.L. Hanauer, "Speech analysis and synthesis by linear prediction of the speech wave," *Journal of the Acoustical Society of America*, vol. 50, no. 2, pp. 637-655, March 1971.
57. S.L. Marple, Jr., *Digital Spectral Analysis With Applications*, Prentice-Hall, Englewood Cliffs, New Jersey, USA, 1987.
58. V.R. Viswanathan and J. Makhoul, "Quantization Properties of Transmission Parameters in Linear Predictive Systems," *IEEE Transactions on Acoustics, Speech, and Signal Processing*, vol. 23, no. 3, pp. 309-321, June 1975.
59. B.S. Atal, "Predictive Coding of Speech at Low Bit Rates," *IEEE Transactions on Communications*, vol. 30, no. 4, pp. 600-614, April 1982.
60. B.S. Atal and J.R. Remde, "A New Model of LPC Excitation for Producing Natural Sounding Speech at Low Bit Rates," in *Proceedings IEEE International Conference on Acoustics, Speech, and Signal Processing*, pp. 614-617, Paris, France, 1982.
61. G.R. Doddington, J. Picone, and J.J. Godfrey, "The LPC trace as an HMM development tool," *Journal of the*

Acoustical Society of America, Vol. 84, pg. S1-561A, Fall 1988.

62. H.F. Silverman and N.R. Dixon, "A Parametrically Controlled Spectral Analysis System for Speech," *IEEE Transactions on Acoustics, Speech, and Signal Processing*, vol. 22, no. 5, pp. 362-381, October 1974 (not the original reference by any means, but a good reference on digital computation of the spectrogram — original references on analog techniques date back to the 1940s).
63. R.V. Churchill, J.W. Brown, and R.F. Verhey, Complex Variables and Applications, McGraw-Hill, New York, New York, USA, 1976.
64. B.S. Atal, "Linear Prediction For Speaker Identification," *Journal of the Acoustical Society of America*, vol. 55, no. 6, pp. 1304-1311, June 1974.
65. A.V. Oppenheim and D.H. Johnson, "Discrete Representation of Signals," in *Proceedings of the IEEE*, vol. 60, no. 6, pp. 681-691, June 1972.
66. K.F. Lee, *Automatic Speech Recognition: the Development of the SPHINX System*, Kluwer Academic Publishers, Boston, Massachusetts, USA, 1989.
67. J.G. Wilpon, C.H. Lee, and L.R. Rabiner, "Application of Hidden Markov Models for Recognition of a Limited Set of Words in Unconstrained Speech," in *Proceedings IEEE International Conference on Acoustics, Speech, and Signal Processing*, pp. 254-257, Glasgow, Scotland, 1989.
68. R.W. Hamming, *Digital Filters*, Prentice-Hall, Englewood Cliffs, New Jersey, USA, 1989 (2nd Edition).
69. H. Ney, "Experiments on Mixture-Density Phoneme Modelling For the Speaker-Independent 1000-Word Speech Recognition DARPA Task," in *Proceedings IEEE International Conference on Acoustics, Speech, and Signal Processing*, pp. 713-716, Albuquerque, New Mexico, USA, April 1990.
70. D. Paul, "A Speaker-Stress Resistant Isolated Word Recognizer," in *Proceedings IEEE International Conference on Acoustics, Speech, and Signal Processing*, pp. 713-716, Dallas, Texas, USA, April 1990.
71. B.H. Juang, L.R. Rabiner, and J.G. Wilpon, "On the Use of Bandpass Liftering in Speech Recognition," *IEEE Transactions on Acoustics, Speech, and Signal Processing*, vol. 35, no. 7, pp. 947-954, July 1987.
72. K. Fukunaga, *Introduction To Statistical Pattern Recognition*, Academic Press, New York, New York, USA, 1972.
73. J. Picone, "Continuous Speech Recognition Using Hidden Markov Models," *IEEE ASSP Magazine*, vol. 7, no. 3, pp. 26-41, July 1990.
74. L.R. Rabiner, "A Tutorial on Hidden Markov Models And Selected Applications In Speech Recognition," in *Proceedings of the IEEE*, vol. 77, no. 2, pp. 257-285, February 1989.
75. W.H. Press, B.P. Flannery, S.A. Teukolsky, and W.T. Vettering, *Numerical Recipes in C: The Art of Scientific Programming*, Cambridge University Press, New York, New York, USA, 1988.
76. M.R. Anderberg, *Cluster Analysis for Applications*, Academic Press, New York, USA, 1973.
77. Y. Linde, A. Buzo, and R.M. Gray, "An Algorithm for Vector Quantizer Design," *IEEE Transactions on Communications*, vol. 28, no. 1, pp. 84-95, January 1980.
78. S.E. Levinson, L.R. Rabiner, A.E. Rosen-

berg, and J.G. Wilpon, "Interactive Clustering Techniques for Selecting Speaker-Independent Reference Templates for Isolated Word Recognition," *IEEE Transactions on Acoustics, Speech, and Signal Processing*, vol. 27, no. 2, pp. 134-141, April 1979.

79. J. Picone and G.R. Doddington, "Low Rate Speech Coding Using Contour Quantization," in *Proceedings IEEE International Conference on Acoustics, Speech, and Signal Processing*, pp. 1652-1655, Dallas, Texas, April 1987.
80. A. Gersho, "On the Structure of Vector Quantizers," *IEEE Transactions on Information Theory*, vol. 28, no. 2, pp. 157-166, March 1982.
81. T. Kohonen, *Self-Organization and Associative Memory*, Third Ed., Springer-Verlag, New York, New York, USA, 1989.
82. R.O. Duda and P.E. Hart, *Pattern Classification and Scene Analysis*, Academic Press, New York, New York, USA, 1973.
83. F. Itakura, "Minimum Prediction Residual Principle Applied To Speech Recognition," *IEEE Transactions on Acoustics, Speech, and Signal Processing*, vol. 23, no. 1, pp. 67-72, February 1975.
84. B. Dautrich, L.R. Rabiner, T.B. Martin, "On the Effects of Varying Filter Bank Parameters on Isolated Word Recognition," *IEEE Transactions on Acoustics, Speech, and Signal Processing*, vol. 31, no. 4, pp. 793-807, August 1983.
85. J. Picone, "Duration In Context Clustering for Speech Recognition," *Speech Communication*, vol. 9, No. 2, pp. 119-128, April 1990.
86. K. Kita, T. Kawabata, H. Saito, "HMM Continuous Speech Recognition Using Predictive LR Parsing," in *Proceedings IEEE International Conference on Acoustics, Speech, and Signal Processing*, pp. 703-706, Glasgow, Scotland, May 1989.
87. A. Waibel, T. Hanazawa, G. Hintom, K. Shikano, and K.J. Lang, "Phoneme Recognition Using Time-Delay Neural Networks," *IEEE Transactions on Acoustics, Speech, and Signal Processing*, vol. 37, no. 3, pp. 328-339, March 1989.
88. Y.L. Chow, M.O. Dunham, O.A. Kimball, M.A. Krasner, G.F. Kubala, J. Makhoul, P.J. Price, S. Roucos, and R.M. Schwartz, "BYBLOS: The BBN Continuous Speech Recognition System," in *Proceedings IEEE International Conference on Acoustics, Speech, and Signal Processing*, pp. 89-92, Dallas, Texas, USA, April 1987.
89. F. Kubala, M. Feng, J. Makhoul, and R. Schwartz, "Speaker Adaptation from Limited Training in the BBN BYBLOS Speech Recognition System," in *Proceedings of the DARPA Speech and Natural Language Workshop*, Morgan Kaufmann Publishers, Inc., Palo Alto, California, USA, pp. 100-105, February 1989.
90. M.M. Hochberg, L.T. Niles, J.T. Foote, and H.F. Silverman, "Hidden Markov Model/Neural Network Training Techniques for Connected Alphadigit Speech Recognition," in *Proceedings IEEE International Conference on Acoustics, Speech, and Signal Processing*, pp. 109-112, Toronto, Ontario, Canada, April 1991.
91. T.D. Harrison, and F. Fallside, "A Connectionist Model for Phoneme Recognition in Continuous Speech," in *Proceedings IEEE International Conference on Acoustics, Speech, and Sig-*

nal Processing, pp. 417-420, Glasgow, Scotland, May 1989.

92. P. Haffner, M. Franzini, and A. Waibel, "Integrating Time Alignment and Neural Networks for High Performance Continuous Speech Recognition," in *Proceedings IEEE International Conference on Acoustics, Speech, and Signal Processing*, pp. 105-109, Toronto, Ontario, Canada, April 1991.
93. L. Fissore, P. Laface, G. Micca, "Comparison of Discrete and Continuous HMMs in a CSR Task Over the Telephone," in *Proceedings IEEE International Conference on Acoustics, Speech, and Signal Processing*, pp. 253-256, Toronto, Ontario, Canada, April 1991.
94. L. Fissore, P. Laface, G. Micca, and R. Pieraccini, "Lexical Access to Large Vocabularies for Speech Recognition," *IEEE Transactions on Acoustics, Speech, and Signal Processing*, vol. 37, no. 8, pp. 1197-1213, August 1989.
95. S. Kimura, "100,000-Word Recognition Using Acoustic-Segment Networks," in *Proceedings IEEE International Conference on Acoustics, Speech, and Signal Processing*, pp. 61-64, Albuquerque, New Mexico, USA, April 1990.
96. A. Averbuch, L. Bahl, R. Bakis, P. Brown, A. Cole, G. Daggett, S. Das, K. Davies, S. De Gennaro, P. de Souza, E. Epstein, D. Fraleigh, F. Jelinek, S. Katz, B. Lewis, R. Mercer, A. Nadas, D. Nahamoo, M. Picheny, G. Shichman, and P. Spinelli, "An IBM-PC Based Large-Vocabulary Isolated Utterance Speech Recognizer," in *Proceedings IEEE International Conference on Acoustics, Speech, and Signal Processing*, pp. 53-56, Tokyo, Japan, April 1986.
97. F. Jelinek, "The Development of an Experimental Discrete Dictation Recognizer," in *Proceedings of the IEEE*, vol. 73, no. 11, pp. 1616-1624, November 1985.
98. L. Deng, V. Gupta, M. Lennig, P. Kenny, and P. Mermelstein, "Acoustic Recognition Component of an 86,000-word Speech Recognizer," in *Proceedings IEEE International Conference on Acoustics, Speech, and Signal Processing*, pp. 741-784, Albuquerque, New Mexico, USA, April 1990.
99. J. Koo, C.K. Un, H.S. Lee, H.R. Kim, and M.W. Koo, "A Recognition Time Reduction Algorithm for Large-Vocabulary Speech Recognition," in *Proceedings of the International Conference on Spoken Language Processing*, pp. 253-256, Kobe, Japan, November 1990.
100. V. Zue, J. Glass, M. Phillips, and S. Seneff, "Acoustic Segmentation and Phonetic Classification in the SUMMIT System," in *Proceedings IEEE International Conference on Acoustics, Speech, and Signal Processing*, pp. 389-392, Glasgow, Scotland, May 1989.
101. S. Mizuta and K. Kakajima, "An Optimal Discriminative Training Method for Continuous Mixture Density HMMs," in *Proceedings of the International Conference on Spoken Language Processing*, pp. 245-248, Kobe, Japan, November 1990.
102. K. Yoshida, T. Watanabe, and S. Koga, "Large Vocabulary Word Recognition Based on Demisyllable Hidden Markov Model Using Small Amount of Training Data," in *Proceedings IEEE International Conference on Acoustics, Speech, and Signal Processing*, pp. 1-4, Glasgow, Scotland, 1989.
103. D. Lubensky, "Word Recognition Using Neural Nets, Multi-State Gaussian and

K-Nearest Neighbor Classifiers," in *Proceedings IEEE International Conference on Acoustics, Speech, and Signal Processing*, pp. 141-144, Toronto, Ontario, Canada, April 1991.

104. S. Matsunaga, S. Sagayama, S. Homma, and S. Furui, "A Continuous Speech Recognition System Based on A Two-Level Grammar Approach," in *Proceedings IEEE International Conference on Acoustics, Speech, and Signal Processing*, pp. 589-592, Albuquerque, New Mexico, USA, April 1990.
105. T. Matsuoka and K. Shikano, "Robust HMM Phoneme Modeling for Different Speaking Styles," in *Proceedings IEEE International Conference on Acoustics, Speech, and Signal Processing*, pp. 265-268, Toronto, Ontario, Canada, April 1991.
106. Y. Zhao and H. Wakita, "Experiments With A Speaker-Independent Continuous Speech Recognition System on the TIMIT Database," in *Proceedings of the International Conference on Spoken Language Processing*, pp. 697-700, Kobe, Japan, November 1990.
107. V. Steinbiss, A. Noll, A Paeseler, H. Ney, H. Bergmann, C. Dugast, H.H. Hamer, H. Piotrowski, H. Tomaschewski, A. Zielinski, "A 10,000-Word Continuous Speech Recognition System," in *Proceedings IEEE International Conference on Acoustics, Speech, and Signal Processing*, pp. 57-60, Albuquerque, New Mexico, USA, April 1990.
108. M.J. Russel, K.M. Ponting, S.M. Peeling, S.R. Browning, J.S. Bridle, R.K. Moore, I. Galiano, P. Howell, "The ARM Continuous Speech Recognition System," in *Proceedings IEEE International Conference on Acoustics, Speech, and Signal Processing*, pp. 69-72, Albuquerque, New Mexico, USA, April 1990.
109. M. Weintraub, H. Murveit, M. Cohen, P. Price, J. Bernstein, G. Baldwin, and D. Bell, "Linguistic Constraints in Hidden Markov Model Based Speech Recognition," in *Proceedings IEEE International Conference on Acoustics, Speech, and Signal Processing*, pp. 699-702, Glasgow, Scotland, 1989.
110. H. Murveit, and M. Weintraub, "1000-Word Speaker-Independent Continuous-Speech Recognition Using Hidden Markov Models," in *Proceedings IEEE International Conference on Acoustics, Speech, and Signal Processing*, pp. 115-118, New York, NY, USA, April 1988.
111. W.S. Meisel, M.T. Anikst, S.S. Pirzadeh, J.E. Schumacher, M.C. Soares, and D.J. Trawick, "The SSI Large Vocabulary Speaker-Independent Continuous Speech Recognition System," in *Proceedings IEEE International Conference on Acoustics, Speech, and Signal Processing*, pp. 273-276, Toronto, Ontario, Canada, April 1991.
112. S. Makino, A. Ito, M. Endo, and K. Kido, "A Japanese Text Dictation System Based on Phoneme Recognition and a Dependency Grammar," in *Proceedings IEEE International Conference on Acoustics, Speech, and Signal Processing*, pp. 273-276, Toronto, Ontario, Canada, April 1991.
113. K. Shirai, N. Hosaka, E. Kitagawa, and T. Endou, "Speaker Adaptable Phoneme Recognition Selecting Reliable Acoustic Features based on Mutual Information," in *Proceedings of the International Conference on Spoken Language Processing*, pp. 353-356, Kobe, Japan, November 1990.
114. H.M. Meng and V.W. Zue, "A Comparative Study of Acoustic Representations of Speech for Vowel Classification Us-

ing Multi-Layer Perceptrons," in *Proceedings of the International Conference on Spoken Language Processing*, pp. 1053-1056, Kobe, Japan, November 1990.

115. R.G. Leonard, "A Database for Speaker-Independent Digit Recognition," in *Proceedings IEEE International Conference on Acoustics, Speech, and Signal Processing*, pp. 42.11.1-42.11.4, San Diego, California, USA, April 1984.

116. H. Hermansky and N. Morgan, "Towards Handling the Acoustic Environment in Spoken Language Processing," in *Proceedings of the International Conference on Spoken Language Processing*, pp. 85-88, Banff, Alberta, Canada, October 1992.

IX. FIGURE CAPTIONS

1. An overview of the signal modeling process is shown. Advances in speech recognition technology have blurred the distinction between signal modeling and statistical modeling by introducing context sensitive statistical models into the parameterization of the signal.
2. The sequence of operations in converting an analog signal to a digital signal suitable for spectral analysis are shown. Some components, such as a low quality A/D converter or a nonlinear microphone, can introduce unwanted artifacts in the signal.
3. The frequency response of a typical telephone grade A/D converter is shown.
4. The frequency responses of common preemphasis filters used in speech recognition systems are given. The motivation behind such filters is to spectrally flatten the speech signal, and to amplify important areas of the spectrum. Values of a_{pre} close to -1.0 are most common.
5. Temporal and frequency domain responses of the Hanning window are shown for a range of values. In this window design, the objective is to make the main lobe width small, and the stopband attenuation large, and also to have the window response near zero at the edge of the window. $\alpha_w = 0.54$ is the most common value used in speech recognition systems today.

5(a). Temporal Response

5(b). Frequency Response

6. A frame-based overlapping analysis is depicted. In this case, a 33% overlap is shown. One-third of the data used in each analysis frame is shared with the previous frame. Note that only one-third of the data are unique to the current frame — the remaining two-thirds are shared between adjacent frames.
7. Various power computations are shown for the speech signal (the word “tea”) in (g). In (a), a rectangular window analysis using a $5\ \text{msec}$ frame duration and $10\ \text{msec}$ window is shown. In (b), a rectangular window analysis with $10\ \text{msec}/20\ \text{msec}$ parameters are shown. In (c), a rectangular window analysis with $20\ \text{msec}/30\ \text{msec}$ are shown. In (d), a Hamming window has been used with $20\ \text{msec}/30\ \text{msec}$ values. In (e), heavy smoothing is demonstrated by using a window duration of $60\ \text{msec}$, while the frame duration is held fixed at $20\ \text{msec}$. Finally, in (f), the power is computed using a recursive-in-time approach using a second-order $50\ \text{Hz}$ low pass filter. The arrow indicates a point where the power changes rapidly. Note that the Hamming window applied in (d) helps smooth this transition.
8. The six major spectral analysis algorithms are shown. Cepstral parameters derived from either a Fourier Transform or Linear Prediction model are by far the most popular of these approaches. The Fourier Transform methods have traditionally been considered robust in severely noisy environments, and are popular for their similarity to the initial stages of the human auditory system.
9. The *Bark* and *mel* scales are shown as a function of acoustic frequency in (a) and (b) respectively. In (c), critical bandwidth as a function of frequency is shown. The *mel* scale is a popular approximation to the *Bark* scale, and is widely used in speech recognition.

9(a). The Bark scale transformation.

9(b). The mel scale transformation.

9(c). The critical bandwidth transformation.

10. Digital filter bank outputs for a speech signal shown in (a), consisting of the word "speech.". In (b), the output from a filter with a center frequency of $250\ \text{Hz}$ and a bandwidth of $100\ \text{Hz}$ is shown. In (c), the output from a filter centered at $2500\ \text{Hz}$ is shown. Note that the amplitude of the output for each filter varies depending on the nature of the sound. The final "ch" sound, for example, is mainly composed of high frequency information.

11. An oversimplified example of the benefit in oversampling the spectrum. A spectrum of a signal (computed by a DFT to be precise) is shown along with the frequency values at which it would be sampled using a filter bank consisting of 5 samples per bin (index 14 from Table 1 is shown). If the spectrum is sampled exactly at the center of the critical band the output value would be $0\ \text{dB}$. If an average of the spectrum across the critical band were used, the value would be $-1\ \text{dB}$. Oversampling the spectrum often results in more stable, or smoothed, amplitude estimates.

12. Low energy areas of the spectrum are often clipped in an effort to emphasize high energy portions of the spectrum in the signal model and limit the effect of areas of the spectrum that are not necessarily perceptually relevant. This clipping is normally executed after preemphasis so that high frequency components of the speech spectrum are not excessively truncated.

13. In the linear acoustics model of speech production [32], the speech signal is produced by filtering an excitation signal (produced in the sub-glottal system) with a time-varying linear filter (the vocal tract). The vocal tract can be decoupled from the excitation signal using homomorphic signal processing techniques. It should be noted that this model is not valid for all classes of speech sounds, such as frication, where excitation occurs above the glottis.

14. An example of the computation of the cepstrum is given. In (a), an unvoiced speech waveform is shown. In (b), a $1000\ \text{point}$ cepstrum is computed. In (c), a voiced speech waveform is shown. Finally, in (d), the corresponding cepstrum is shown. Note that the cepstrum in (d) indicates periodicity in the waveform by the presence of two local maxima. The low order terms in the cepstrum reflect the smooth spectral structure of the speech signal (vocal tract information).

15. A speech spectrum is shown along with LP models of order 4 and 8. Note that the model order 4 does not sufficiently model the detail in the spectrum. Model orders of 10 and 12 are often used in speech recognition systems.

16. Stabilization of the LP model is demonstrated. A speech spectrum is shown along with an LP model of order 10, and the same LP model with a stabilization factor of $-10\ \text{dB}$. Note that while the bottom of the spectral model is raised, the performance of the model around the spectral peaks is also significantly smoothed. In cases where two spectral resonances are close in frequency, stabilization sometimes tends to combine these into on broad spectral peak (a bandwidth broadening effect).

17. Spectral analysis for a speech signal is demonstrated by showing a wideband spectrogram of a speech signal, and three associated LP models. Normally, the analysis presented in (e) is sufficient to capture salient aspects of the individual sounds. However, as computational power increases, and phonetic recognition technology improves, $10\ \text{msec}$ frame durations may become more common, because of the need for better characterizations of dynamic sounds such as consonants.

17(a). A speech waveform (the word "speech") and its power contour.

17(b). A wideband spectrogram (6 msec window).

17(c). A spectrogram of the LP model ($T_f = 5$ msec, $T_w = 10$ msec).

17(d). A spectrogram of the LP model ($T_f = 10$ msec, $T_w = 20$ msec).

17(e). A spectrogram of the LP model ($T_f = 20$ msec, $T_w = 30$ msec).

18. The bilinear transform is compared to the mel scale for a range of values. The bilinear transform was computed using a sample frequency of 16 kHz. Note that positive values of α_{bt} produce compression of the frequency scale (shown here) while negative values produce expansion.

19. A speech spectrum along with its LP model are shown. In addition, the log magnitude spectrum of the LP-derived cepstral coefficients, and the log magnitude spectrum of the corresponding warped cepstral coefficients ($\alpha_{bt} = 0.25$) are shown. Similar results can be obtained for the LP spectrum by processing either the LP coefficients or the autocorrelation function through the iterative transform.

20. Conversion of signal measurements to a signal parameter vector usually consists of two steps: differentiation (optional) and collation. Most speech recognition systems today use the absolute measurements and an estimate of the first derivative of the measurements. Recently, estimates of the second derivative have been incorporated. The output of this stage of processing is a single parameter vector which is the concatenation of all parameters.

21. The frequency responses for three different realizations of a differentiator are shown. In (a), $N_d = 1$. In (b), $N_d = 3$. In (c), $N_d = 5$. Note that an ideal differentiator has a frequency response proportional to the log of the frequency. It is desirable however to attenuate high frequencies (it is important to not excessively amplify these components) because higher frequency components tend to be noisy. Hence, each of these designs attenuates high frequencies. The first-order difference, as shown in (a), is most common.

22. Statistical models in speech recognition are generally divided into two categories: parametric models (continuous distributions) and nonparametric models (discrete distributions). The types of models range from direct evaluation of the LP model to sophisticated likelihood models based on decorrelation transformations.

23(a). The elliptical region shows the range of allowable values of an order pair (x,y) (assume all points in this region are equally likely). Which distance is greater: (a) the distance from point a to point b, or (b) the distance from point c to point d? The answer is (a). Since the distance from a to b is a larger percentage of the variance in the vertical direction, we would have to believe this distance is “perceptually” larger than the distance from point c to point d. (Note that the distances as shown are exactly one unit.)

23(b). An important variation of the problem in Fig. 23(a): which distribution does the data point belong to? The distance from the center of each distribution to the data point are the same. However, since the shapes of the distributions are different, on what scale do we compare the two distances? [72]

24. An example of a transformation computed over speech data collected over the telephone. The original data was sampled at 8 kHz. The first curve corresponds to an eigenvector that weighs low frequency filter bank amplitudes heavily and deemphasizes high frequencies. The second curve depicts a dimension that focuses on the 1 kHz region of the spectrum. Both of these dimensions attempt to track first formant information for vowels. The third curve represents a dimension that favors high frequency sounds, such as sibilants. Use of these types of transformations to model specific classes of speech sounds is an area of on-going research in speech recognition.
25. The K-MEANS algorithm is demonstrated for a two-dimensional clustering problem. Input vectors are grouped according to a nearest neighbor rule into clusters. The centers for these clusters are recomputed based on the data in the cluster. The data is then reclassified using the new clusters. The procedure is repeated until the quality of the clustering is acceptable. The cluster centers then become the vectors in the codebook.
26. Codebook distortion is displayed as a function of the codebook size. Codebook size often ranges between 32 and 256 in speech recognition systems today. Beware that codebook distortion is at best weakly correlated with speech recognition performance.

X. TABLE CAPTIONS

1. Two critical band filter banks are shown. The first filter bank (cols. 2 and 3) is a design based on the Bark scale. The second (cols. 4 and 5) is a design based on the mel scale. The shaded entries are shown only for comparison purposes. Usually these bins are not included in the design. For the Bark scale filter bank, telephone grade speech is often processed using a filter bank consisting of 16 bands (indices 2 - 17).
2. A summary of the common signal modeling techniques used in speech recognition systems. See the notes at the end of the table for explanations of the various abbreviations. (This table extends for several pages.)

XI. FIGURES

Figure 1:

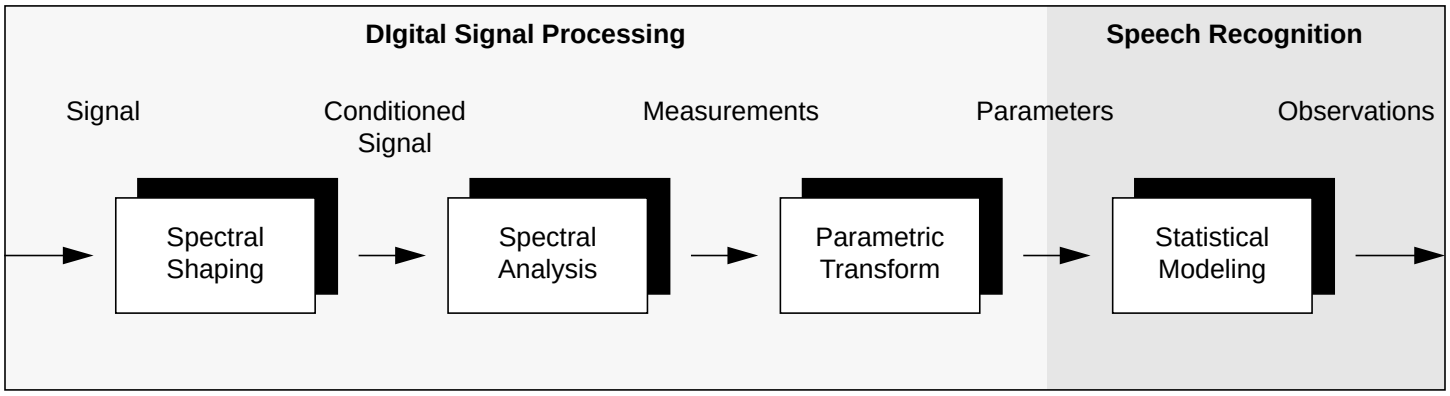


Figure 2:

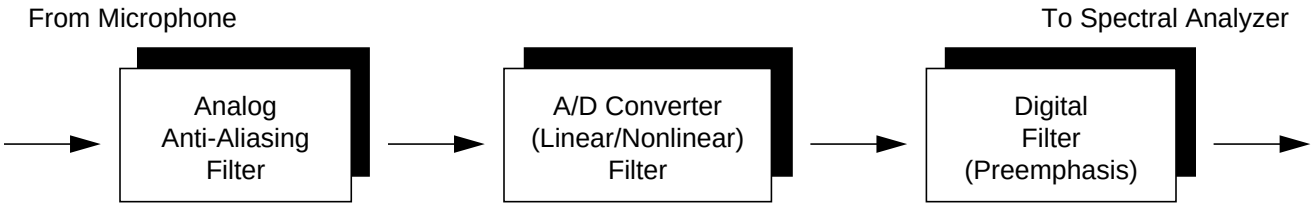

From Microphone
To Spectral Analyzer
Analog
Anti-Aliasing
Filter
A/D Converter
(Linear/Nonlinear)
Filter
Digital
Filter
(Preemphasis)

Figure 3:

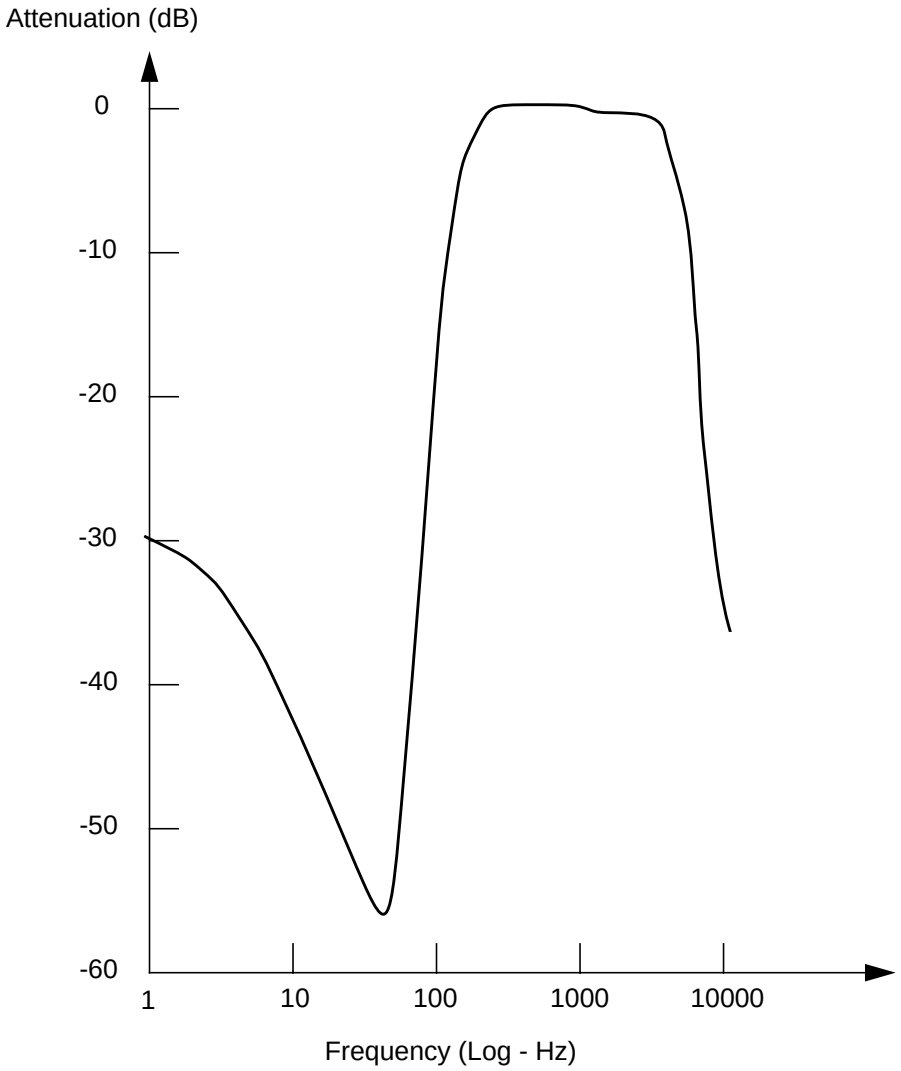

Attenuation (dB)
0
-10
-20
-30
-40
-50
-60
1
10
100
1000
10000
Frequency (Log - Hz)

Figure 4:

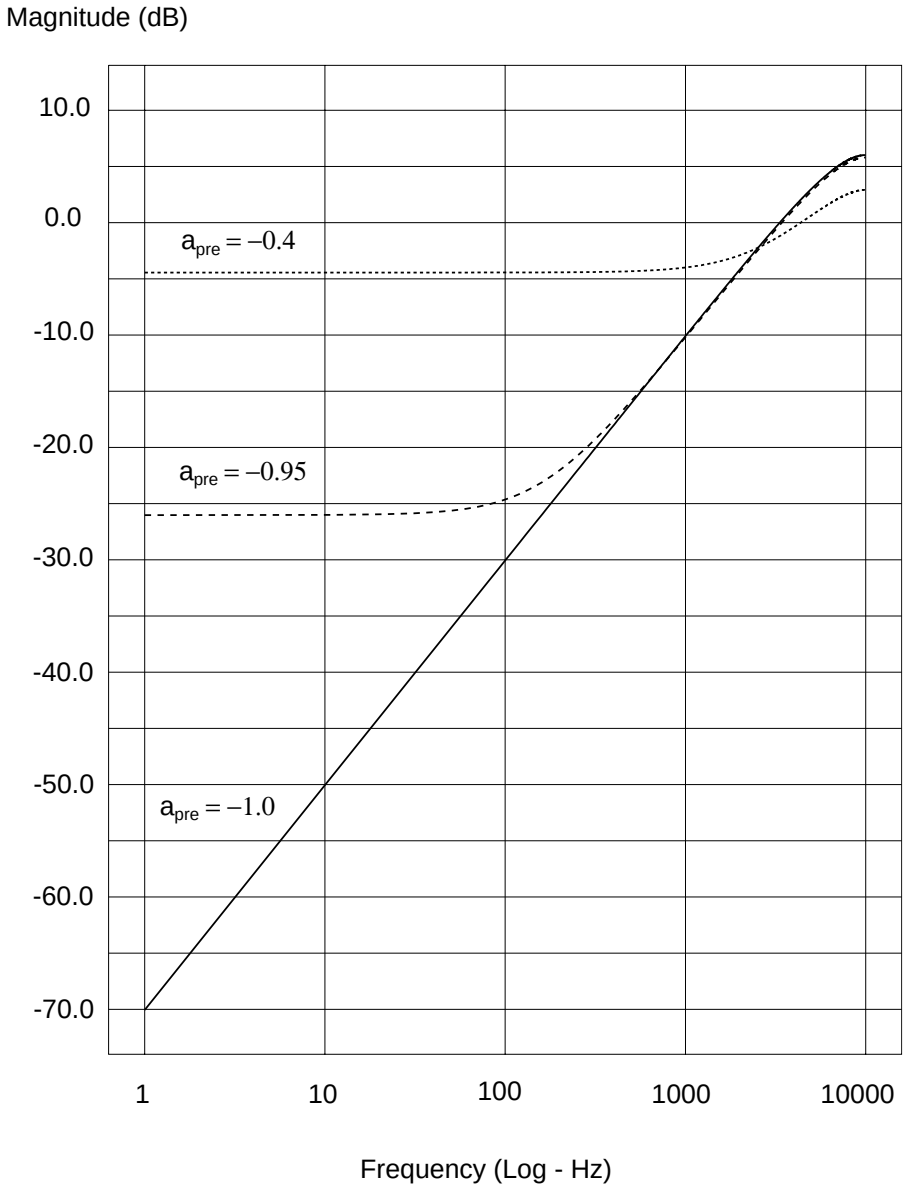

Magnitude (dB)
10.0
0.0
-10.0
-20.0
-30.0
-40.0
-50.0
-60.0
-70.0
$a_{pre} = -0.4$
$a_{pre} = -0.95$
$a_{pre} = -1.0$
1
10
100
1000
10000
Frequency (Log - Hz)

Figure 5:

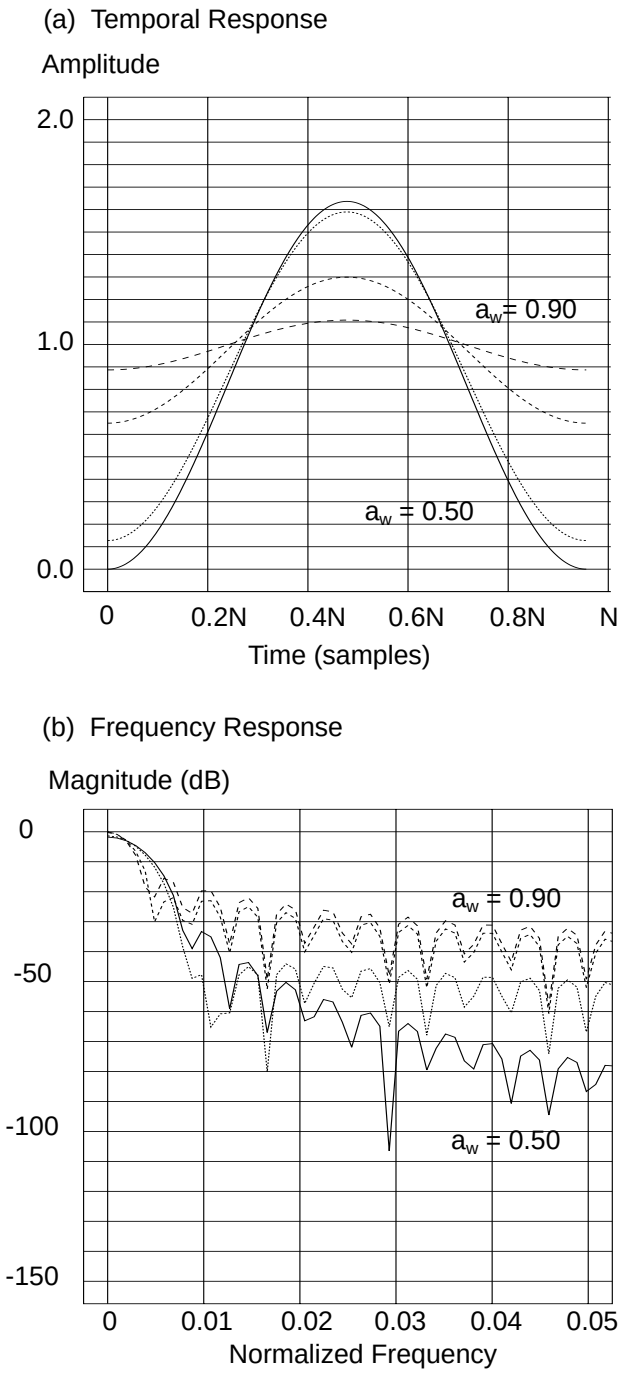

(a) Temporal Response
Amplitude
2.0
1.0
0.0
a_w = 0.90
a_w = 0.50
0
0.2N
0.4N
0.6N
0.8N
N
Time (samples)
(b) Frequency Response
Magnitude (dB)
0
-50
-100
-150
a_w = 0.90
a_w = 0.50
0
0.01
0.02
0.03
0.04
0.05
Normalized Frequency

Figure 6:

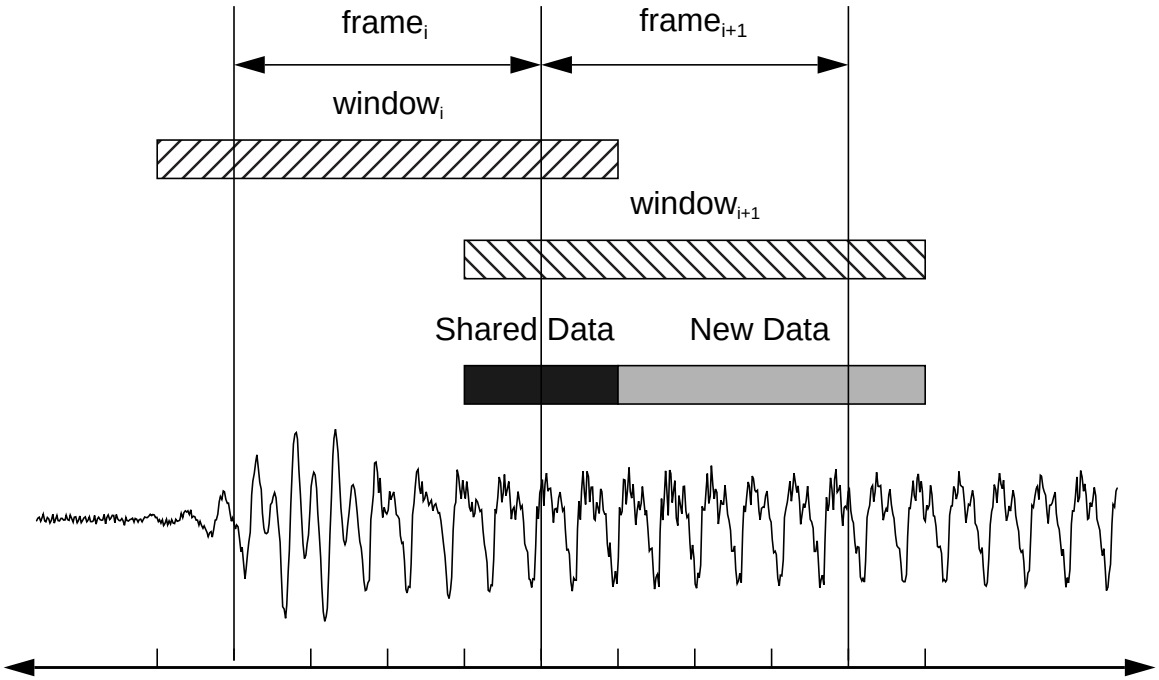


Figure 7:

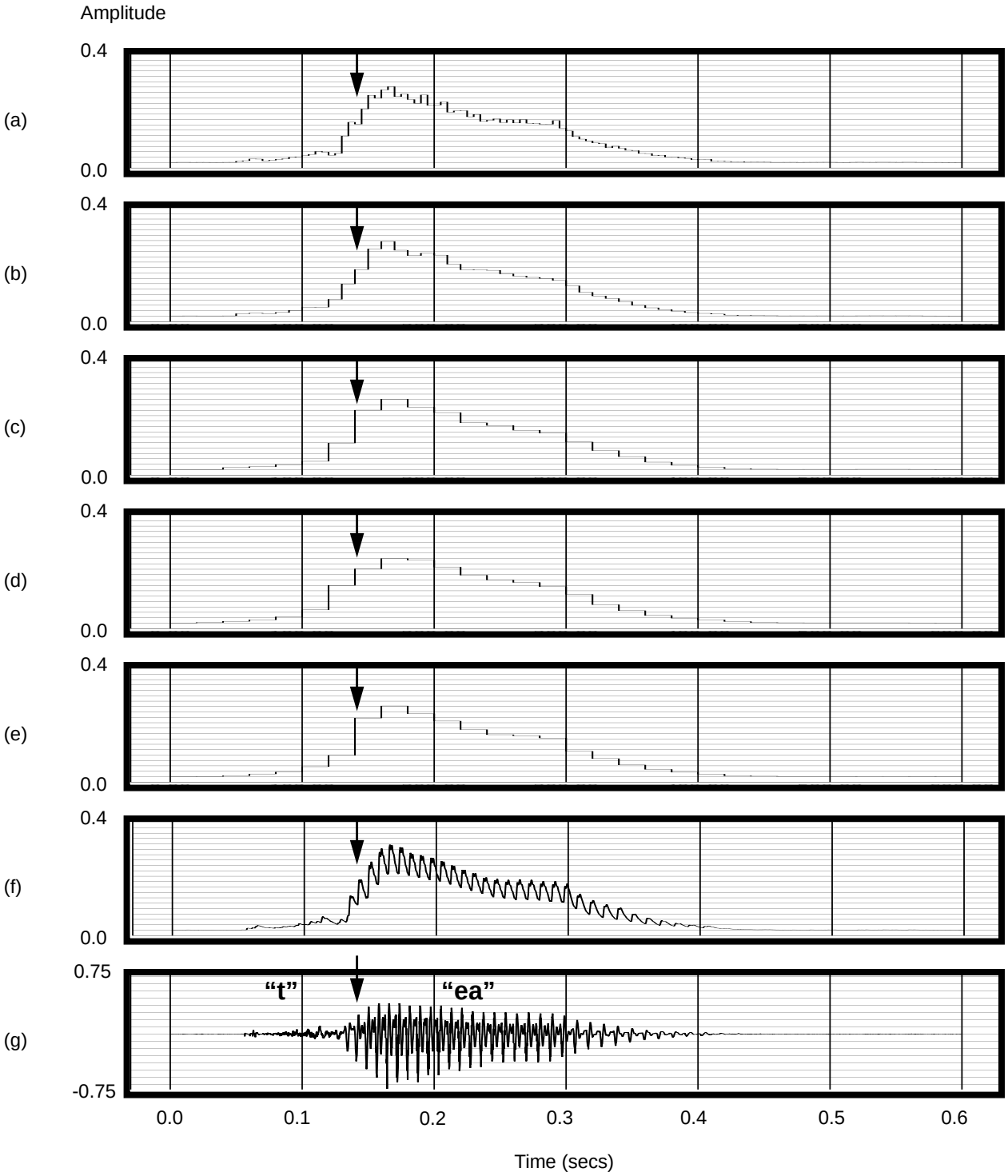

Amplitude
0.4
0.0
(a)
(b)
(c)
(d)
(e)
(f)
(g)
0.75
-0.75
“t”
“ea”
0.0
0.1
0.2
0.3
0.4
0.5
0.6
Time (secs)

Figure 8:

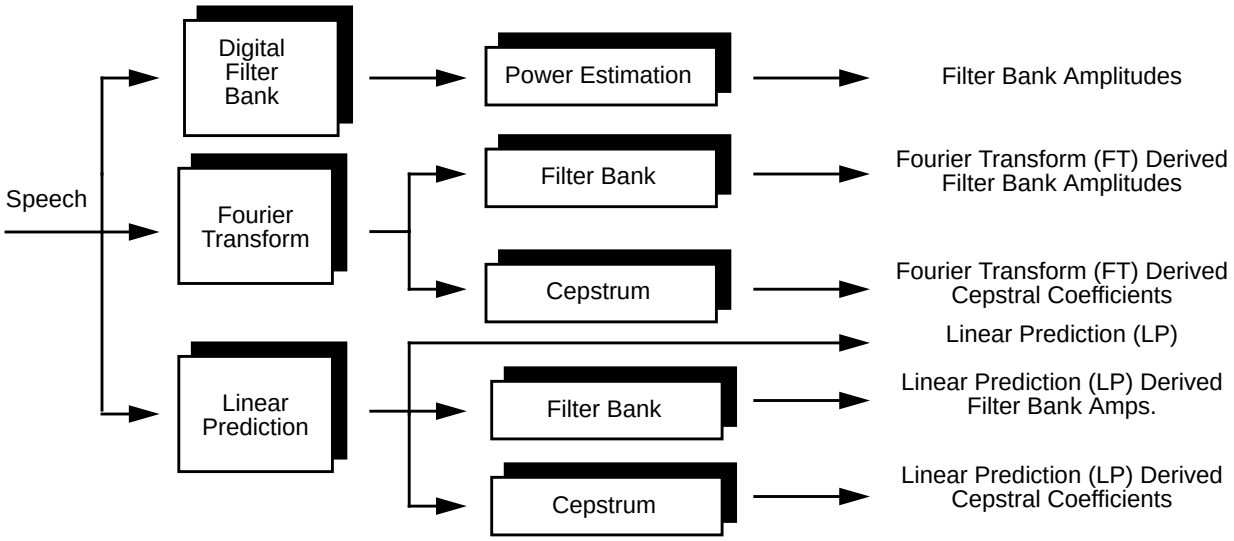

Speech
Digital Filter Bank
Fourier Transform
Linear Prediction
Power Estimation
Filter Bank
Cepstrum
Filter Bank
Cepstrum
Filter Bank Amplitudes
Fourier Transform (FT) Derived Filter Bank Amplitudes
Fourier Transform (FT) Derived Cepstral Coefficients
Linear Prediction (LP)
Linear Prediction (LP) Derived Filter Bank Amps.
Linear Prediction (LP) Derived Cepstral Coefficients

Figure 9:

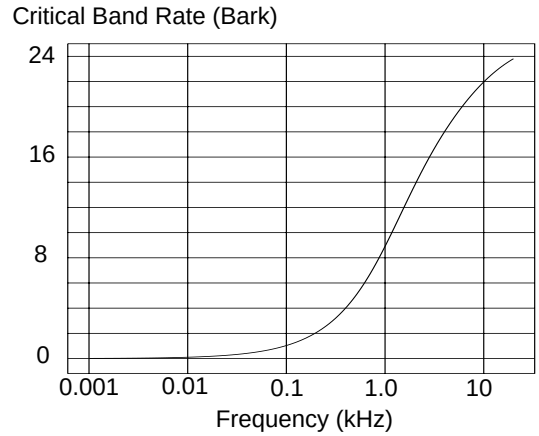


Figure 9(a). The Bark scale transformation.

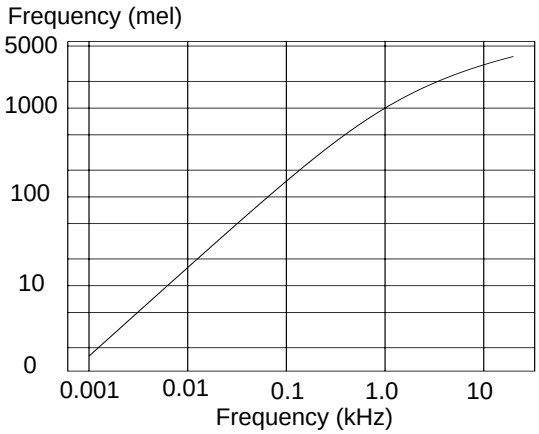


Figure 9(b). The mel scale transformation.

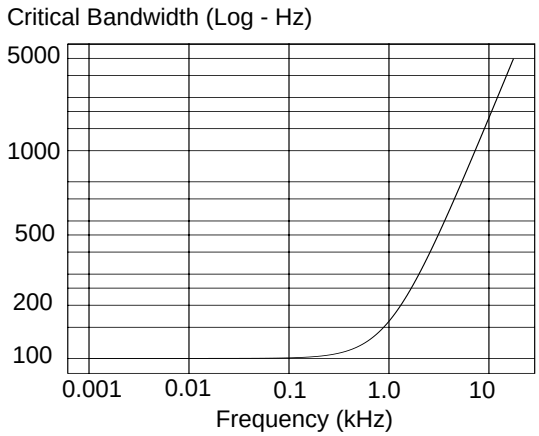


Figure 9(c). The critical bandwidth transformation.

Figure 10:

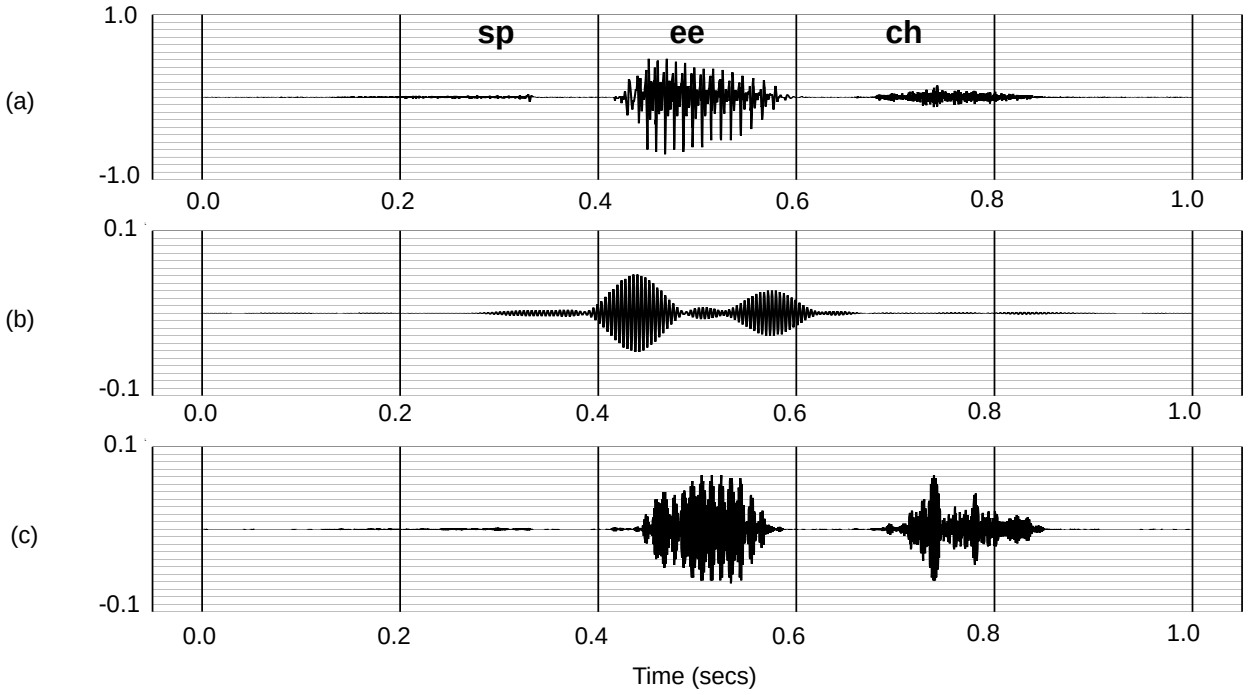


Figure 11:

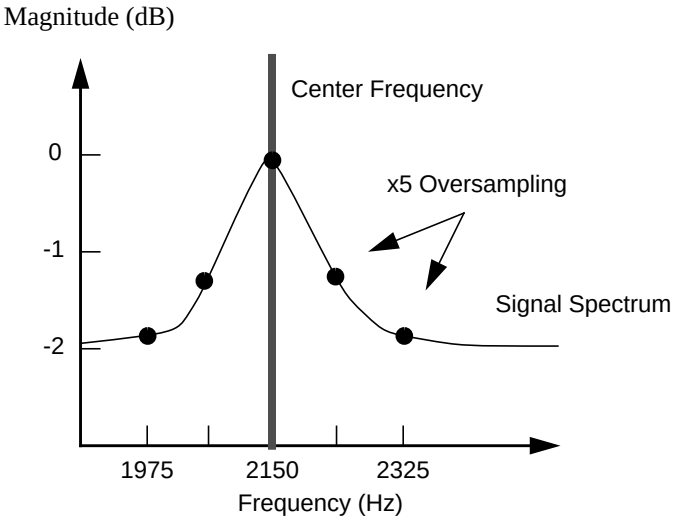

Magnitude (dB)
Center Frequency
0
x5 Oversampling
-1
Signal Spectrum
-2
1975
2150
2325
Frequency (Hz)

Figure 12:

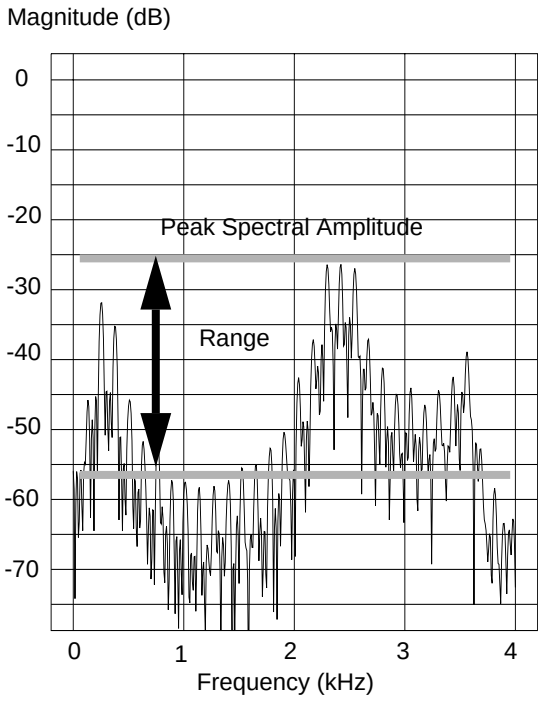

Magnitude (dB)
0
-10
-20
-30
-40
-50
-60
-70
Peak Spectral Amplitude
Range
0
1
2
3
4
Frequency (kHz)

Figure 13:

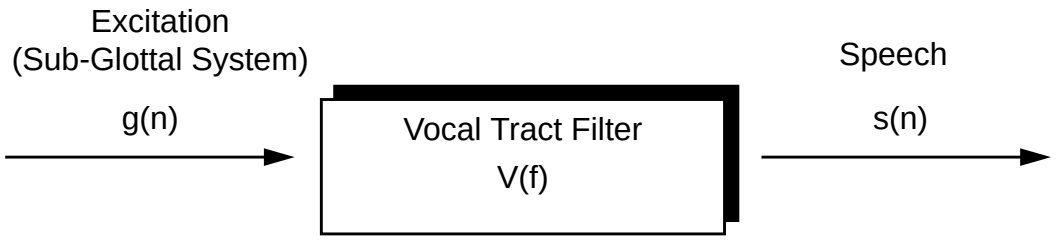

Excitation
(Sub-Glottal System)
g(n)
Vocal Tract Filter
V(f)
Speech
s(n)

Figure 14:

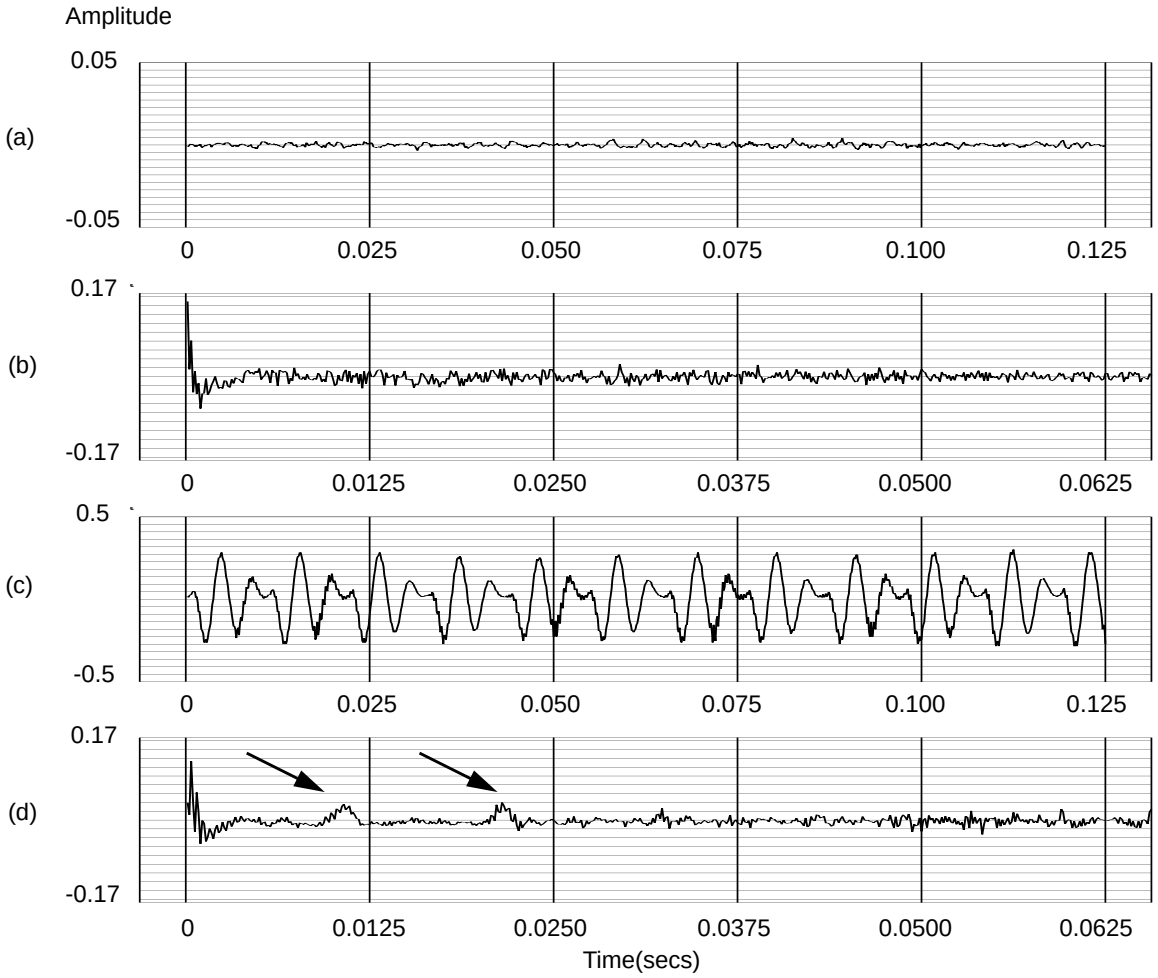

Amplitude
(a)
0.05
-0.05
0
0.025
0.050
0.075
0.100
0.125
(b)
0.17
-0.17
0
0.0125
0.0250
0.0375
0.0500
0.0625
(c)
0.5
-0.5
0
0.025
0.050
0.075
0.100
0.125
(d)
0.17
-0.17
0
0.0125
0.0250
0.0375
0.0500
0.0625
Time(secs)

Figure 15:

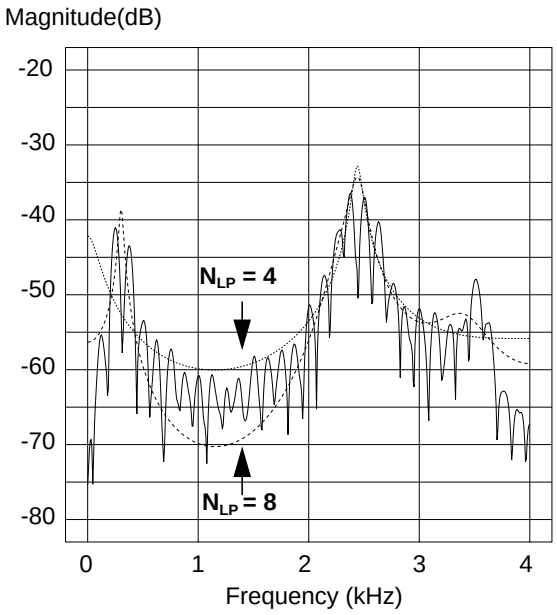

Magnitude(dB)
-20
-30
-40
-50
-60
-70
-80
$N_{LP} = 4$
$N_{LP} = 8$
0
1
2
3
4
Frequency (kHz)

Figure 16:

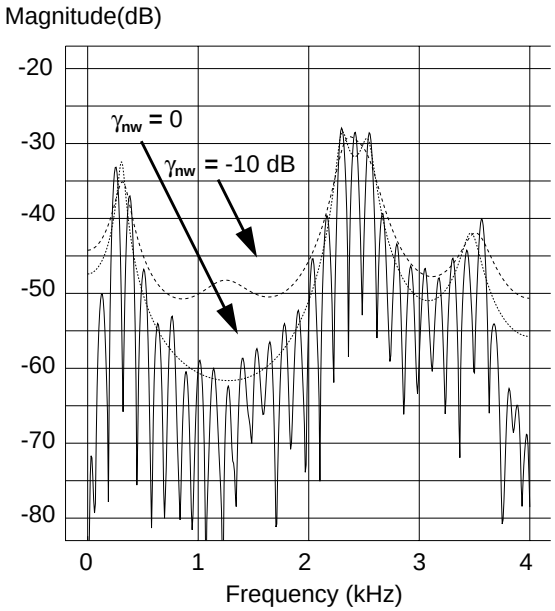
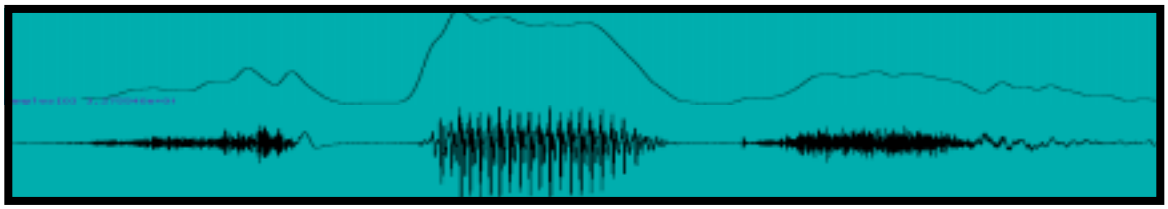
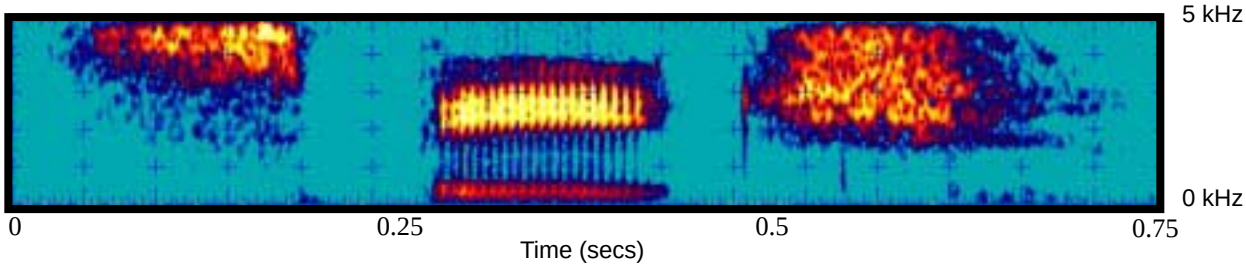

Magnitude(dB)
-20
-30
-40
-50
-60
-70
-80
γ_{nw} = 0
γ_{nw} = -10 dB
0
1
2
3
4
Frequency (kHz)

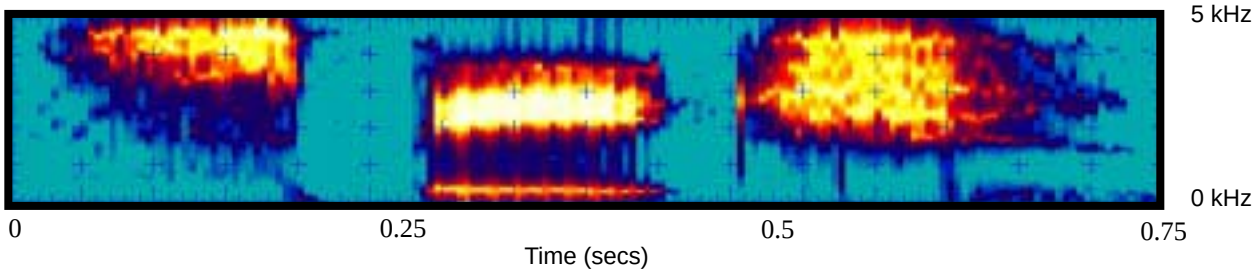
Figure 17:



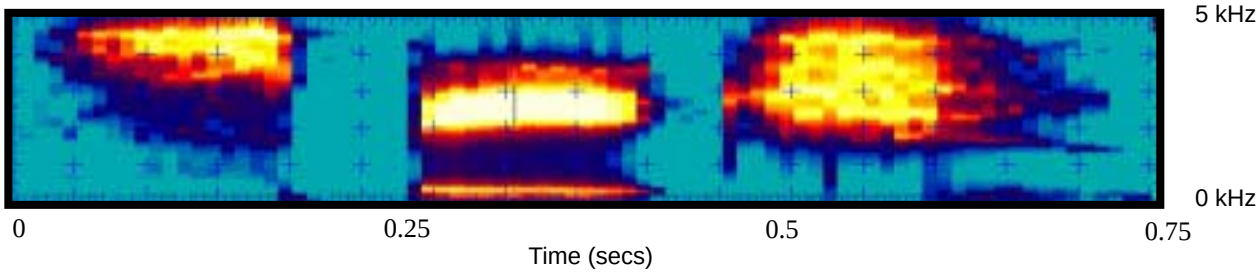
(a) A speech waveform (the word “speech”) and its power contour.



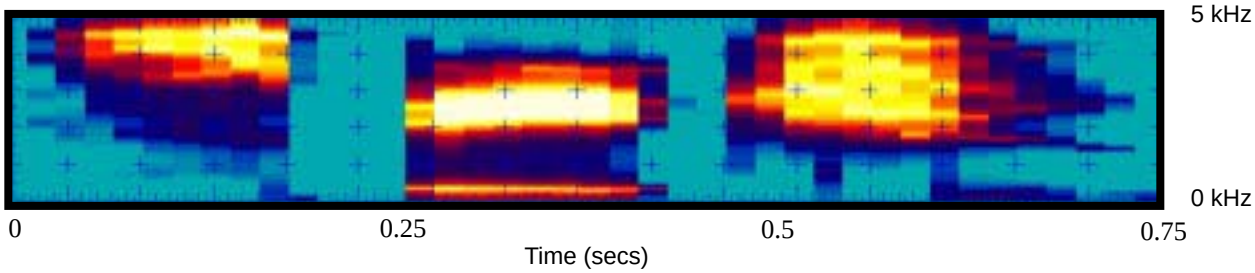
(b) A wideband spectrogram (6 msec window).



(c) A spectrogram of the LP model (T_f = 5 msec, T_w = 10 msec).



(d) A spectrogram of the LP model (T_f = 10 msec, T_w = 20 msec).



(e) A spectrogram of the LP model (T_f = 20 msec, T_w = 30 msec).

Figure 18:

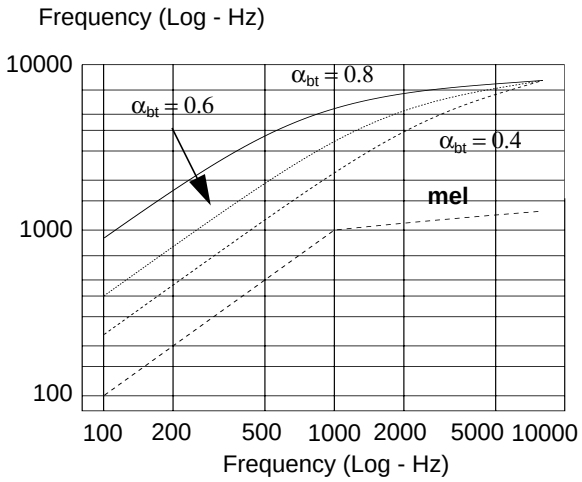

Frequency (Log - Hz)
10000
1000
100
$\alpha_{bt} = 0.8$
$\alpha_{bt} = 0.6$
$\alpha_{bt} = 0.4$
mel
100
200
500
1000
2000
5000
10000
Frequency (Log - Hz)

Figure 19:

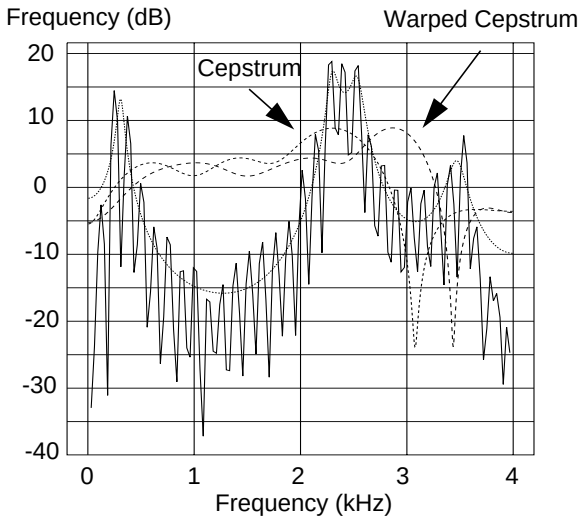

Frequency (dB)
Warped Cepstrum
Cepstrum
20
10
0
-10
-20
-30
-40
0
1
2
3
4
Frequency (kHz)

Figure 20:

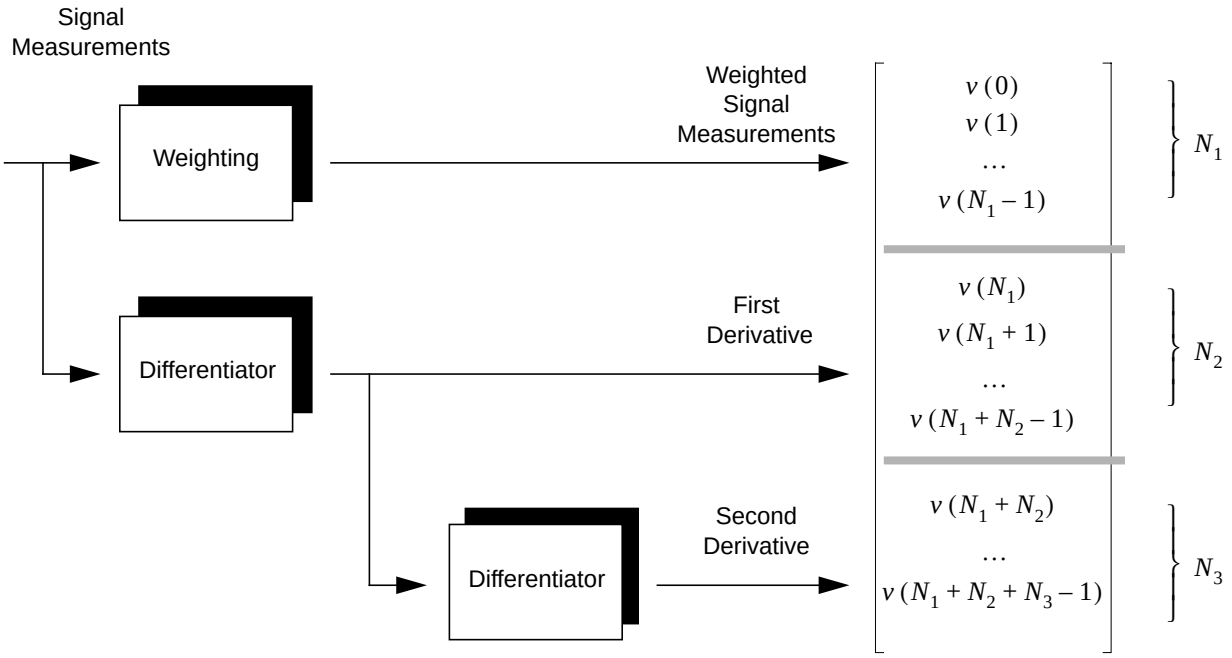

Signal
Measurements
Weighting
Differentiator
Differentiator
Weighted
Signal
Measurements
First
Derivative
Second
Derivative
$v(0)$
$v(1)$
…
$v(N_1-1)$
$v(N_1)$
$v(N_1+1)$
…
$v(N_1+N_2-1)$
$v(N_1+N_2)$
…
$v(N_1+N_2+N_3-1)$
N_1
N_2
N_3

Figure 21:

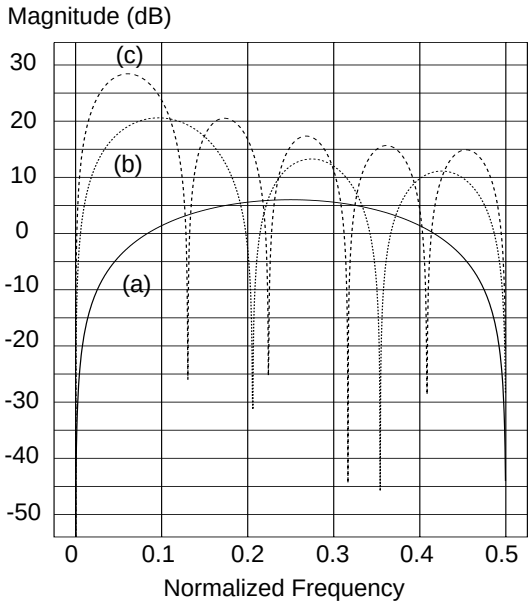

Magnitude (dB)
30
20
10
0
-10
-20
-30
-40
-50
(c)
(b)
(a)
0
0.1
0.2
0.3
0.4
0.5
Normalized Frequency

Figure 22:

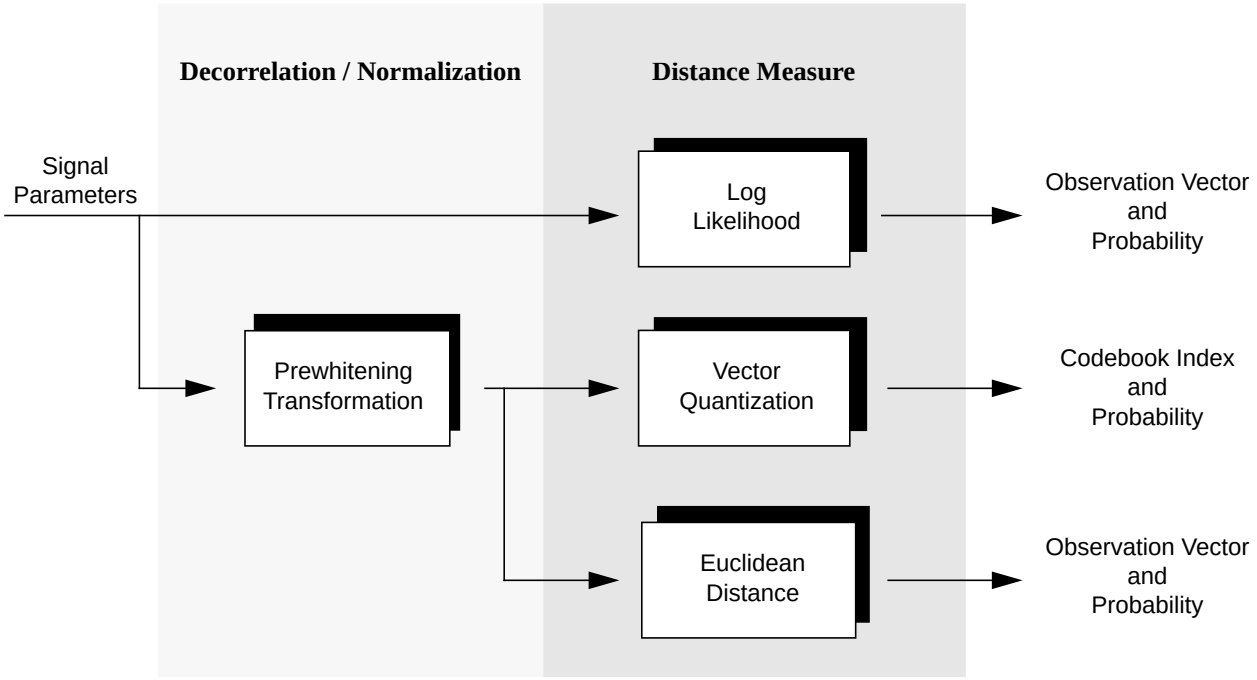

Decorrelation / Normalization
Distance Measure
Signal Parameters
Log Likelihood
Observation Vector and Probability
Prewhitening Transformation
Vector Quantization
Codebook Index and Probability
Euclidean Distance
Observation Vector and Probability

Figure 23(a):

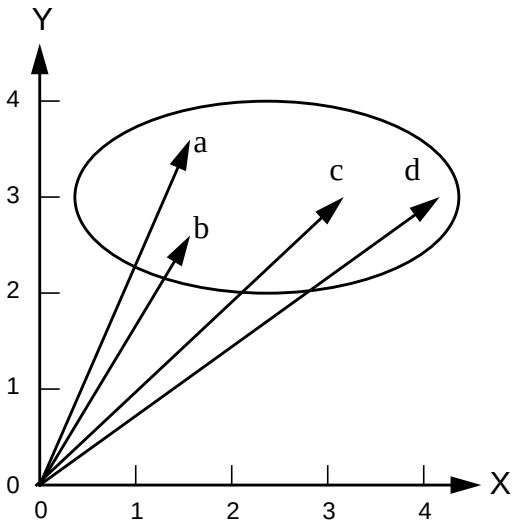


Figure 23(b):

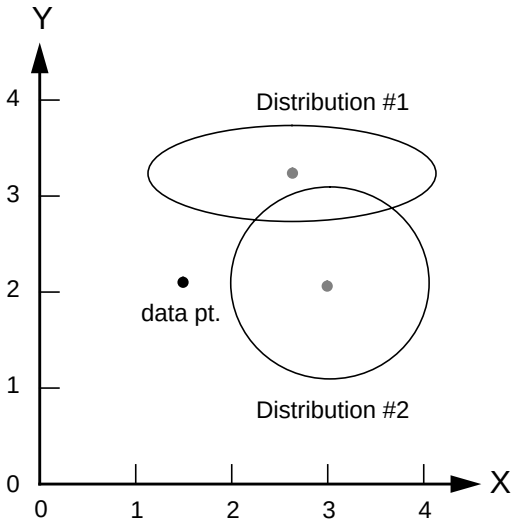


Figure 24:

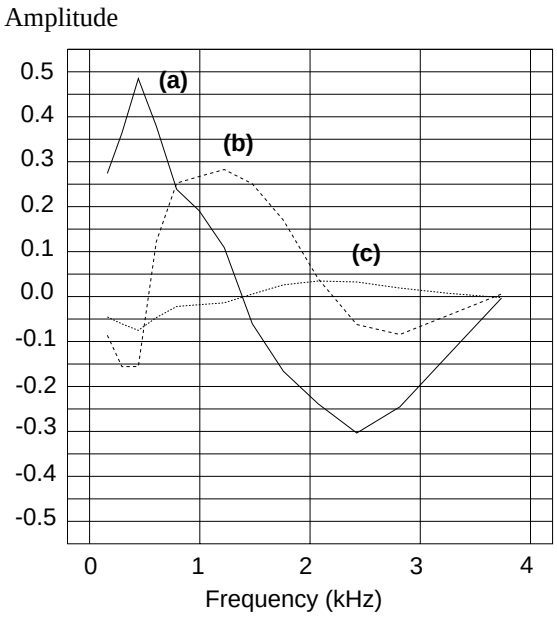

Amplitude
0.5
0.4
0.3
0.2
0.1
0.0
-0.1
-0.2
-0.3
-0.4
-0.5
(a)
(b)
(c)
0
1
2
3
4
Frequency (kHz)

Figure 25:

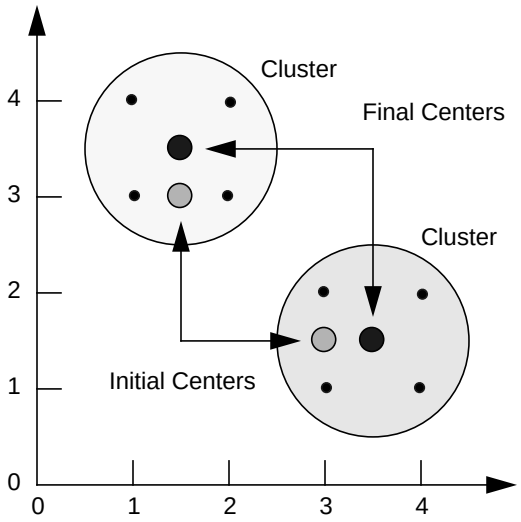
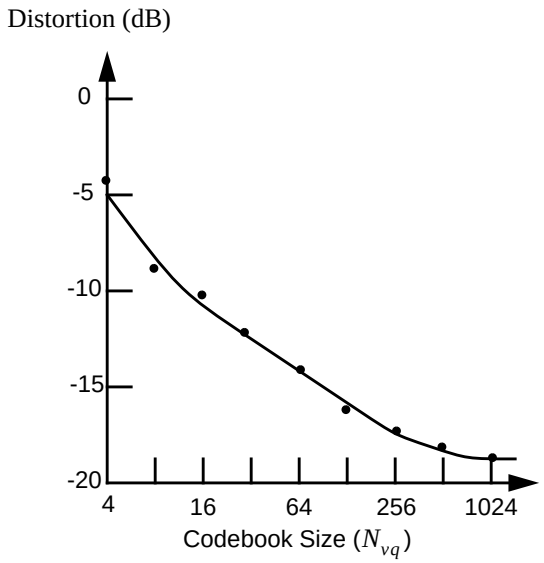

Cluster
Final Centers
Cluster
Initial Centers
0
1
2
3
4
0
1
2
3
4

Figure 26:



XII. TABLES

Table 1:

Index	Bark Scale		Mel Scale	
	Center Freq. (Hz)	BW (Hz)	Center Freq. (Hz)	BW (Hz)
1	50	100	100	100
2	150	100	200	100
3	250	100	300	100
4	350	100	400	100
5	450	110	500	100
6	570	120	600	100
7	700	140	700	100
8	840	150	800	100
9	1000	160	900	100
10	1170	190	1000	124
11	1370	210	1149	160
12	1600	240	1320	184
13	1850	280	1516	211
14	2150	320	1741	242
15	2500	380	2000	278
16	2900	450	2297	320
17	3400	550	2639	367
18	4000	700	3031	422
19	4800	900	3482	484
20	5800	1100	4000	556
21	7000	1300	4595	639
22	8500	1800	5278	734
23	10500	2500	6063	843
24	13500	3500	6964	969

Table 2:

General Information		Signal Measurements					Signal Parameters	Statistical Model	
Affiliation [Ref.]	Application Size/Task	f_s kHz	a_{pre}	Frame Dur. $msec$	Wind. Dur. $msec$	Spectral Analysis	Parameter Types	Model	Speech Recog. Technology
ATR [87]	Large Office	12	-0.97	3	21.3	LP(12)	LP Power	PWLR VQ(256)	DD-HMM
ATR [86]	Large Office	12	0.0	5	21.3	FFT(128)	Mel FB, D-FB, D-D-FB	-	TD-NN
AT&T [7]	Small Telecom	6.67	-0.95	15	45	LP(8) Cep(12)	Liftered-Cep. D-Cep. D-D-Cep. Power D-Power D-D Power	Variance	CD-HMM
AT&T [69]	Medium Office	8	-0.95	10	30	LP(10) Cep(12)	(Same)	(Same)	(Same)
BBN [88,89]	Large Office	20	0.0	10	20	FFT(512) Cep.(14)	Mel-Cep. D-Cep. Power D-Power	VQ	DD-HMM
Brown [90]	Small Office	16	0.0	10.0	40.0	LP(12) Cep(12)	Cep. D-Cep Power D-Power	MS-VQ (256/stage)	HMM-NN
Cambridge [91]	Large Office	10	FD	10	10	FFT(128)	FB	-	NN
CMU [66]	Large Office	16	-0.97	10.0	20.0	LP(14) Cep(12)	BT Cep. D-Cep. Power D-Power	MS-VQ (256/stage)	DD-HMM
CMU [92]	Large Office	16	-0.97	10.0	20.0	FFT(256)	Mel-FB D-FB D-D-FB	MS-VQ (256/stage)	TD-NN
CSELT [93]	Medium Telecom	16	0.0	10	20	FFT(256) Cep(12)	Mel-Cep. D-Cep. Power D-Power	Variance MS-VQ	CD-HMM DD-HMM
CSELT [94]	Large Office	12	0.0	10	20	FFT(256) Cep(18)	Mel-Cep.	VQ (128)	DD-HMM
Fujitsu [95]	Large Office	16	0.0	5	32	FFT(512)	Mel-FB Power	Identity	DP
IBM [96,97]	Large Office	16	0.0	10	20	-	Auditory Model (20)	-	DD-HMM
INRS [17]	Large Office	16	0.0	10	25.6	FFT(256)	Mel-Cep. D-Cep. Power	MS-VQ (64/stage)	CD-HMM
INRS [98]	Large Office	16	0.0	10	25.6	FFT(256)	(Same)	Variance	DD-HMM
KAIST [99]	Large Office	10	0.0	10	25.6	LP(12) Cep(12)	Cep. D-Cep.	MS-VQ (256/stage)	DD-HMM

(Continued on next page)

General Information		Signal Measurements					Signal Parameters	Statistical Model	
Affiliation [Ref.]	Application Size/Type	f_s kHz	a_{pre}	Frame Dur. $m$sec	Wind. Dur. $m$sec	Spectral Analysis	Parameter Types	Model	Speech Recog. Technology
LL [2,70]	Med./Lg. Office/Mil.	8	FD	10	20.0	FFT(256) Cep(14)	Mel-Cep. D-Cep.	Fixed	CD-HMM
MIT [54,100,113]	Large Office	16	FD	5	-	FB(40)	Auditory Model	PT	CD-CFG
Mitsubishi [101]	Large Office	10	-0.95	10	25.6	FFT(256)	Mel-Cep. D-Cep. Power D-Power	Variance	CD-HMM
NEC [102]	Large Office	16	0.0	5	32	FFT(512) Cep(10)	Mel-Cep.	Variance	CD-HMM
NYNEX [103]	Small Telecom	8	0.95	5	20	LP(10) Cep(10)	Cep. D-Cep. Power D-Power	PT -	HMM MLP-NN
NTT [11,104]	Large Office	12	0.0	10	30	LP(16) Cep(16)	Cep. D-Cep. D-Power	Variance	DD-FSA
NTT [105]	Large Office	12	0.0	8	32	(Same)	(Same)	Variance	CD-HMM
Panasonic [106]	Large Office	10.67	0.0	9.3	18.6	PLP(8)	Cep. D-Cep. Power D-Power	Fixed	CD-HMM
Phillips [107]	Large Office	16	0.0	10	25	FFT(512)	Mel-FB D-FB D-D-FB Power D-Power D-D-Power	MS-VQ	DD-HMM
RSRE [108]	Large Office/Mil.	20	0.0	10	-	FB(27)	Bark-Cep. Power D-Cep.	Fixed	CD-HMM
SRI [109,110]	Large Office	16	0.0	8	16	FFT(256)	Mel-Cep. D-Cep. Power, D-Power	MS-VQ	DD-CSG
SSI [111]	Large Office	16	0.0	6.6	-	FB(20)	Bark FB Max Ampl. Delta-Max Ampl. Pitch	-	CD-HMM
TI [6,14,18]	Small Telecom	8	-1.0	20	30.0	LP (10)	Mel-FB, D-FB, Power, D-Power	PT	CD-HMM
Tohoku Univ. [112]	Large Office	16	0.0	10	-	FB(29)	Cep., D-Cep.	-	LVQ2-NN
Waseda [113]	Large Office	12	0.0	10	20	LP(16)	Mel-Cep. D-Cep. Power D-Power	MS-VQ (256/stage)	DD-HMM

(Continued on next page)

Notes:

A. An explanation of abbreviations:

Small:	Small Sized Vocabulary (usually Digit Recognition or Alpha Digit Recognition)
Medium:	Medium-Size Vocabulary (usually < 5,000 words)
Large:	Large Vocabulary Speech Recognition (usually > 5,000 words)
Office:	Database was collected in a quiet or typical office environment (noise level is usually about ~70 dB SPL)
Telecom:	Telecommunications Data (data collected over standard telephone lines)
Mil.:	Military applications involving noisy environments and different speaking styles
FD:	Frequency Domain Preemphasis (applied directly to the spectrum - ~10-20 dB/decade)
LP:	Linear Prediction (order is shown in parentheses)
PLP:	Perceptually-Motivated Linear Prediction
FFT:	Fast Fourier Transform
FB:	Filter Bank
Cep.:	Cepstral Parameters
Power:	Signal Power (usually in dB)
Mel:	Mel Scale Parameters
Bark:	Bark Scale Parameters
Liftered:	Liftered Parameters
BT:	Mel Scale Parameters Computed Using the Bilinear Transform
D-FB:	Delta (or Time Derivative of) Filter Bank
D-D-FB:	Delta-Delta Filter Bank Parameters (Second Derivative)
D-Power:	Delta-Power (Time Derivative of Power)
D-D-Power:	Delta-Delta-Power (Second Derivative of Power)
PWLR:	Perceptually Weighted Log LIkelihood Distance Measure
VQ:	Vector Quantization
MS-VQ:	Multi-stage Vector Quantization (VQ with multiple codebooks)
PT:	Prewhitening Transformation
Variance:	Variance-Weighted Parameters (Diagonal components of the prewhitening transformation)
Identity:	Identity Matrix Weighted Parameters (No Weighting)
Fixed:	A Fixed *(a priori)* Weighting Matrix is used (sometimes called “Pooled” or “Grand”)
HMM:	Hidden Markov Model
NN:	Neural Network
TD-NN:	Time Delay Neural Network
DP:	Dynamic Programming
CD-HMM:	Continuous Density Hidden Markov Models
DD-HMM:	Discrete Density HMM
FSA:	Finite State Automaton (usually a regular grammar)
CFG:	Context Free Grammar
CSG:	Context Sensitive Grammar
LVQ2:	Learning Vector Quantizer (a Neural Network approach to vector quantization)
MLP:	Multi-Layer Perceptron Neural Network

B. An explanation of categories:

Affiliation:	Company/University principally responsible for the cited research
Application:	A brief summary of the type of database used in the cited publication
Signal Measurements:	Sample frequency, preemphasis, frame duration, and window duration of the spectral analysis. For some systems, preemphasis is performed directly in the frequency domain (indicated by a “yes”). Under spectral analysis, the sequence of operations is shown. Orders of analysis, where applicable, are shown in parentheses.
Signal Parameters:	Signal parameters used in the system. These are derived from the spectral analysis parameters.

Statistical Model:	The statistical model used in the speech recognition system. Some affiliations have multiple entries.