Mixture Autoregressive Hidden Markov Models for Speech Signals

BIING-HWANG JUANG, MEMBER, IEEE AND LAWRENCE R. RABINER, FELLOW, IEEE

Abstract—**In this paper a signal modeling technique based upon finite mixture autoregressive probabilistic functions of Markov chains is developed and applied to the problem of speech recognition, particularly speaker-independent recognition of isolated digits. Two types of mixture probability densities are investigated: finite mixtures of Gaussian autoregressive densities (GAM) and nearest-neighbor partitioned finite mixtures of Gaussian autoregressive densities (PGAM). In the former (GAM), the observation density in each Markov state is simply a (stochastically constrained) weighted sum of Gaussian autoregressive densities, while in the latter (PGAM) it involves nearest-neighbor decoding which in effect, defines a set of partitions on the observation space. In this paper we discuss the signal modeling methodology and give experimental results on speaker independent recognition of isolated digits. We also discuss the potential use of the modeling technique for other applications.**

I. Introduction

SIGNAL modeling based upon hidden Markov models (HMM's) may be viewed as an effective technique that extends conventional stationary spectral analysis principles to the analysis of time-varying signals [1]–[2]. It uses a Markov chain to model the changing statistical characteristics that are only probabilistically manifested through the actual observations. Therefore, a hidden Markov process is a doubly stochastic process in which there is an unobservable Markov chain, each state of which is associated with a probabilistic function, characterized by a probability distribution or a density function. The Markov chain is defined by a state transition matrix, while the probabilistic functions, designated as the observation probabilities or densities, may be nonparametric or parametric representations.

The primary concern in the hidden Markov modeling technique is the estimation of model parameters from observed sequences. A reestimation algorithm due to Baum and Eagon [3] is usually used for this purpose. The algorithm was first proposed in 1967 by Baum and Eagon for the estimation problem of HMM's with discrete observation densities (probability mass functions). Baum *et al.* [4] later extended the algorithm to continuous density HMM's with some limitations. In their work, they required that the density functions be strictly log concave. This requirement was relaxed by Liporace [5] who, by invoking a representation theorem due to Fan [6], successfully accommodated a broad class of elliptically symmetric density functions so that problematic densities, such as Cauchy, in the original development of [4] could be dealt with. Recently, Juang *et al.* [7] further expanded the estimation algorithm to cope with finite mixtures of strictly log concave and/or elliptically symmetric density functions. Since Gaussian mixture densities can be used to approximate any continuous probability density function (in the sense of minimizing the usual L^k error between two density functions) [8], the modeling capability of hidden Markov processes has thus been greatly enhanced.

In this paper, we concentrate primarily on mixtures of Gaussian autoregressive densities which are the bases in the maximum likelihood formulation of the ubiquitous linear prediction analysis [1], [2], [9]. Poritz was the first person to show how the ideas of linear prediction analysis could be welded into the hidden Markov model methodology [1]. However, in his work, Poritz only considered a single Gaussian autoregressive density per state. Our work extends this initial work to the case of a mixture of Gaussian autoregressive densities, retaining the theoretically consistent formulation of the problem proposed by Poritz.

We consider two types of mixture densities in this paper. The first type is simply a finite mixture of Gaussian autoregressive densities, denoted as GAM (Gaussian autoregressive mixture) for brevity, which allows straightforward maximum likelihood estimation based on the results in [1], [2], and [7]. The second type is a finite mixture of Gaussian autoregressive densities with nearest-neighbor decoding, denoted as PGAM (P for Partitioned). In this type of mixture density, the vector space is implicitly partitioned into regions. Each region is defined by a Gaussian autoregressive density. To evaluate the pdf for a point in the measure space, the appropriate region to which the point belongs is first found by a nearest-neighbor criterion. As will be explained later, this resembles a vector quantization operation in source coding. One such example is the finite-state vector quantizer (FSVQ) as discussed by Dunham *et al.* [22]. The goal is to further link the vector quantization technique for source coding to a superstructure, namely a Markov chain, so as to exploit the nonmemoryless nature of speech signals. The difference between FSVQ and PGAM hidden Markov model is rather subtle in that the FSVQ implies instantaneous decoding of the state sequence pertaining to the Markov chain while a PGAM hidden Markov model treats *each* state sequence as one probabilistic component in evaluating the likelihood of the observation sequence.

The paper is organized as follows. In Section II, we review the mathematical formulation of the problem and develop a reestimation algorithm for the mixture density

Manuscript received March 2, 1985; revised June 13, 1985.
The authors are with AT&T Bell Laboratories, Murray Hill, NJ 07974.

modeling technique. Then, in Section III, we discuss the way in which the model estimation procedure was implemented. In Section IV we report the results of our speaker independent isolated digit recognition experiments with the technique. We then summarize our findings and discuss other potential applications.

II. Mixture Autoregressive Hidden Markov Models

We consider an N-state homogeneous Markov chain with state transition matrix $A = [a_{ij}]$, $i, j = 1, 2, \cdots, N$. Associated with each state j of the unobservable Markov chain is a probability density function $b_j(\boldsymbol{x})$ of the observed K-dimensional random vector $\boldsymbol{x}' = [x_0, x_1, \cdots, x_{K-1}]$ (K consecutive samples of the speech signal). We will use the notation $\boldsymbol{b}_j$ to denote the parameters defining $b_j(\boldsymbol{x})$. Also let $\boldsymbol{O}$ be the observed sequence, $\boldsymbol{O} = (\boldsymbol{o}_1, \boldsymbol{o}_2, \cdots, \boldsymbol{o}_T)$, where T is the duration of the sequence and each $\boldsymbol{o}_t$ is an observed vector $\boldsymbol{x}$. The probability density function for $\boldsymbol{O}$ in the T-fold Cartesian product of the K-dimensional vector space, $\mathfrak{R}_K^T = \mathfrak{R}_K \times \mathfrak{R}_K \times \cdots \times \mathfrak{R}_k$, is then

$$f(\boldsymbol{O}|\lambda) = \sum_S f(\boldsymbol{O}, S|\lambda) = \sum_S \pi_{s_0} \prod_{t=1}^{T} a_{s_{t-1}s_t} b_{s_t}(\boldsymbol{O}_t). \tag{1}$$

In (1), we use λ to denote the hidden Markov model, $\lambda = (\pi, A, B)$ where π is the initial state probability vector, A is the transition matrix, and B is the set of parameters defining $\{b_j(\boldsymbol{x})\}$, $j = 1, 2, \cdots, N$. S is a state sequence, $S = (s_0, s_1, \cdots, s_t)$, $s_t \in \{1, 2, \cdots, N\}$, and the summation is over all possible state sequences S.

In GAM, the observation density $b_j(\boldsymbol{x})$, for $j = 1, 2, \cdots, N$, has the form

$$\text{GAM:} \quad b_j(\boldsymbol{x}) = \sum_{m=1}^{M} c_{jm} b_{jm}(\boldsymbol{x}) \tag{2}$$

where M is the number of mixture components, c_{jm} is the weight for the mth mixture component, and the double-subscripted function $b_{jm}(\boldsymbol{x})$ is the basis probability density function for the mth mixture component, all related to state j. The mixture weight c_{jm} must satisfy the stochastic constraint

$$\sum_{m=1}^{M} c_{jm} = 1, \qquad j = 1, 2, \cdots, N \tag{3}$$

so that

$$\int_{\mathfrak{R}_K} b_j(\boldsymbol{x})\, d\boldsymbol{x} = \sum_{m=1}^{M} c_{jm} \int_{\mathfrak{R}_K} b_{jm}(\boldsymbol{x})\, d\boldsymbol{x} = 1.$$

In PGAM, on the other hand, the observation density $b_j(\boldsymbol{x})$ assumes the form

$$\text{PGAM:} \quad b_j(\boldsymbol{x}) = \max_{m=1,2,\cdots,M} c_{jm} b_{jm}(\boldsymbol{x}). \tag{4}$$

It is clear that, in PGAM, only the most likely mixture component is chosen as the observation density for each particular vector $\boldsymbol{x}$. There is thus a built-in classification rule that implies a partition on $\mathfrak{R}_K$. Let Ω_{jm}, $m = 1, 2, \cdots, M$ be the partitioned regions. In each region Ω_{jm}, every $\boldsymbol{x} \in \Omega_{jm}$ has

$$c_{jm} b_{jm}(\boldsymbol{x}) \geq c_{jl} b_{jl}(\boldsymbol{x})$$

for all $l \neq m$. The stochastic constraint becomes

$$\int_{\mathfrak{R}_k} b_j(\boldsymbol{x})\, d\boldsymbol{x} = \sum_{m=1}^{M} c_{jm} \int_{\Omega_{jm}} b_{jm}(\boldsymbol{x})\, d\boldsymbol{x} = 1. \tag{5}$$

The constraint on c_{jm} that results from (5) is discussed below.

Parameters of the model to be estimated therefore include: 1) the transition matrix $[a_{ij}]$, $i, j = 1, 2, \cdots, N$; 2) the mixture weight (for GAM) $[c_{jm}]$, $j = 1, 2, \cdots, N$ and $m = 1, 2, \cdots, M$; and 3) all necessary parameters defining the set of basis probability densities $\{b_{jm}(\boldsymbol{x})\}$, $j = 1, 2, \cdots, N$ and $m = 1, 2, \cdots, M$. As will be shown shortly, the stochastic constraints on the transition probabilities, i.e., $\Sigma_{j=1}^{N} a_{ij} = 1$, as well as the mixture weights as required in (3) for the GAM case can be automatically satisfied in the reestimation algorithm. To satisfy the requirement of (5) for PGAM, nevertheless, is not a trivial task during reestimation where an increase in likelihood after each iteration must be maintained. We therefore use a simplified expression for $b_j(\boldsymbol{x})$ in the PGAM case

$$\text{PGAM:} \quad b_j(\boldsymbol{x}) = \frac{1}{M} \max_{m=1,2,\cdots,M} b_{jm}(\boldsymbol{x}). \tag{6}$$

Note that the use of (6) excludes the mixture weights as the model parameters since $c_{jm} = 1/M$ for all j and m is implied. In addition, we assume that

$$\int_{\Omega_{jm}} b_{jm}(\boldsymbol{x})\, d\boldsymbol{x} \simeq 1 \qquad \text{for all} \quad j, m$$

for the constraint of (5) to be approximately satisfied. Experimentally, we found this to be the case for typical values of N and M chosen in this study.

A. Gaussian Autoregressive Densities

We assume that the basis probability density function for the observation vectors is Gaussian autoregressive as discussed earlier. We also assume the order of autoregression to be p, so that $\boldsymbol{x}$ has the relationship

$$x_k = -\sum_{i=1}^{p} a_i x_{k-i} + e_k \tag{7}$$

where e_k, $k = 0, 1, 2, \cdots, K-1$ are Gaussian i.i.d. random variables with zero mean and unity variance and a_i, $i = 1, 2, \cdots, p$, are the autoregression coefficients or predictor coefficients. It can be shown [2], [9] that for large K, the density function for $\boldsymbol{x}$ is approximately

$$f(\boldsymbol{x}) \simeq (2\pi)^{-K/2} \exp\{-\tfrac{1}{2}\,\delta(\boldsymbol{x}; \mathbf{a})\} \tag{8}$$

where

$$\delta(\boldsymbol{x}; \mathbf{a}) = r_a(0)\, r(0) + 2 \sum_{i=1}^{p} r_a(i)\, r(i), \tag{9}$$

$$\mathbf{a}' = [1, a_1, a_2, \cdots, a_p]$$

$$r_a(i) \triangleq \sum_{n=0}^{p-i} a_n a_{n+i} \quad \text{with} \quad a_0 = 1 \tag{10}$$

and

$$r(i) \triangleq \sum_{n=0}^{k-i-1} x_n x_{n+i}. \tag{11}$$

Note that the assumption $E\{e_i^2\} = 1$ implies that the observation vector $\boldsymbol{x}$ has already been properly scaled. (In LPC analysis terminology, this is equivalent to normalization by the square root of the *average* residual energy; i.e., if a frame of speech samples has σ^2 as the average LPC residual energy per sample, then x_i is the ith speech sample value in the frame divided by σ.) To use unscaled speech samples as the observations, a corresponding density function can be easily derived [1], [2]. Also note that r_a's are the autocorrelation of the autoregressive coefficients and r's are the autocorrelation of the (normalized) observation samples. Maximum likelihood estimation of the autoregressive coefficients from $\boldsymbol{x}$ requires minimization of $\delta(\boldsymbol{x}; \mathbf{a})$, a procedure equivalent to the autocorrelation method in linear prediction analysis.

We use (8) as the basis density function, since the approximation does not pose any difficulty. Each basis density is thus defined by an autoregression vector $\boldsymbol{a}$ or equivalently an autocorrelation vector $\boldsymbol{r}_a = [r_a(0)\ r_a(1) \cdots r_a(p)]'$. Using appropriate subscripts, we have

$$b_{jm}(\boldsymbol{x}) = (2\pi)^{-K/2} \exp\left\{-\tfrac{1}{2}\, \delta(\boldsymbol{x}; \mathbf{a}_{jm})\right\}, \tag{12}$$

where $\mathbf{a}_{jm}$ is clearly the parameter vector defining the density for the mth mixture component in state j.

B. Reestimation Transformation

We do not intend to provide a full development of the reestimation algorithm here. Instead, we refer to [2], [4], [5], and [7] for the algorithm and its convergence proof. For self-completeness, however, we summarize the algorithm as follows.

For a given observation $\boldsymbol{O}$, the reestimation algorithm starts with an initial guess of the model λ. A transformation that maps the parameter space into itself is then obtained based upon λ. The transformation leads to new model $\bar{\lambda}$ which has $f(\boldsymbol{O}|\bar{\lambda}) > f(\boldsymbol{O}|\lambda)$ unless λ is a fixed point of the transformation. The procedure is iterated after replacing the old model with the new model, and stops when a fixed point, which corresponds to a critical point of $f(\boldsymbol{O}|\lambda)$, is reached. The procedure guarantees an increase in likelihood after each iteration and will converge to a local optimum, when proper densities, such as those outlined in [7] or those under current consideration, are used.

Most of the theoretical results presented here follows the treatment in [2] and [7] and we shall present them without extensive explanation. The estimation is based upon multiple observation sequences. We denote these sequences as $\boldsymbol{O}^{(l)}$, $l = 1, 2, \cdots, L$ where L is the total number of sequences. Each sequence $\boldsymbol{O}^{(l)} = (\boldsymbol{o}_1^{(l)}, \boldsymbol{o}_2^{(l)}, \cdots, \boldsymbol{o}_{T^{(l)}}^{(l)})$ is of duration $T^{(l)}$. We also use $S^{(l)}$ to designate the unobservable stochastic state sequence corresponding to the observation $\boldsymbol{O}^{(l)}$; $S^{(l)} = (s_0^{(l)}, s_1^{(l)}, \cdots, s_{T^{(l)}}^{(l)})$. The new model, after transformation, is denoted as $\bar{\lambda} = (\bar{\pi}, \bar{A}, \bar{B})$ where $\bar{A} = [\bar{a}_{ij}]$, $i, j = 1, 2, \cdots, N$, and $\bar{B}$ consists of $[\bar{\mathbf{a}}_{jm}]$ and $\bar{C} = [\bar{c}_{jm}]$, $j = 1, 2, \cdots, N$, $m = 1, 2, \cdots, M$ for the GAM case. (We shall skip discussion on $\bar{\pi}$ as it is of no interest in the current situation.) For the PGAM case based upon (6), $\bar{C}$ is not part of the parameter set and hence not subject to estimation.

The transformation of the transition probability, a_{ij}, is straightforward and is applicable for both GAM and PGAM cases. In particular,

$$\bar{a}_{ij} = \frac{\sum_{l=1}^{L} \sum_{t=1}^{T^{(l)}} f(\boldsymbol{O}^{(l)}, s_{t-1}^{(l)} = i, s_t^{(l)} = j|\lambda)/f(\boldsymbol{O}^{(l)}|\lambda)}{\sum_{l=1}^{L} \sum_{t=1}^{T^{(l)}} f(\boldsymbol{O}^{(l)}, s_{t-1}^{(l)} = i|\lambda)/f(\boldsymbol{O}^{(l)}|\lambda)}. \tag{13}$$

We use $f(\cdot)$ to denote generically the density or likelihood function without ambiguity. The quantity $f(\boldsymbol{O}^{(l)}, s_{t-1}^{(l)} = i, s_t^{(l)} = j|\lambda)$ is the probability density evaluated for the case of observing $\boldsymbol{O}^{(l)}$ with a transition from state i at time $t - 1$ to state j at time t, given the current model parameter set λ. Equation (13) has an intuitive interpretation of simply being the fractional transition count from state i to state j averaged over all sequences.

The mixture weight transformation, c_{jm}, as needed in the GAM case, is also relatively straightforward. However, we need to introduce another set of stochastic processes, $H^{(l)}$, $l = 1, 2, \cdots, L$, to designate the mixture component; $H^{(l)} = (h_1^{(l)}, h_2^{(l)}, \cdots, h_{T^{(l)}}^{(l)})$. For convenience, we shall call $H^{(l)}$ a branch sequence. Each $h_t^{(l)} \in \{1, 2, \cdots, M\}$ is a random variable and represents an event in which the observation $\boldsymbol{o}_t^{(l)}$ is drawn from mixture component (branch) $h_t^{(l)}$. The quantity $f(\boldsymbol{O}^{(l)}, s_t^{(l)} = j, h_t^{(l)} = m|\lambda)$ is then the probability density evaluated for the case of observing $\boldsymbol{O}^{(l)}$ with $\boldsymbol{o}_t^{(l)}$ being drawn from the mixture component defined by $b_{jm}(\cdot)$, of state j at time t, given λ. The mixture weight transformation can now be stated as

$$\bar{c}_{jm} = \frac{\sum_{l=1}^{L} \sum_{t=1}^{T^{(l)}} f(\boldsymbol{O}^{(l)}, s_t^{(l)} = j, h_t^{(l)} = m|\lambda)/f(\boldsymbol{O}^{(l)}|\lambda)}{\sum_{l=1}^{L} \sum_{t=1}^{T^{(l)}} f(\boldsymbol{O}^{(l)}, s_t^{(l)} = j|\lambda)/f(\boldsymbol{O}^{(l)}|\lambda)}. \tag{14}$$

It should be noted that the stochastic constraint of (3) is automatically satisfied in the reestimation transformation.

The transformation of autoregression coefficient vectors, $\bar{\mathbf{a}}_{jm}$, for each component involves the observation autocorrelation and the linear prediction minimization procedure mentioned in Section II-A. Denoting the autocorrelation coefficients of observation vector $o_t^{(l)}$ by $r_t^{(l)}(\cdot)$, i.e.,

$$o_t^{(l)} = [o_{t,0}^{(l)} \quad o_{t,1}^{(l)} \cdots o_{t,K-1}^{(l)}]'$$

and

$$r_t^{(l)}(i) = \sum_{j=0}^{K-1-i} o_{t,j}^{(l)} o_{t,(j+i)}^{(l)}, \tag{15}$$

we then have a new set of autocorrelation parameters $\bar{r}_{jm}$ for each mixture component m in each state j

$$\bar{r}_{jm}(i) = \frac{\sum_{l=1}^{L} \sum_{t=1}^{T^{(l)}} f(O^{(l)}, s_t^{(l)} = j, h_t^{(l)} = m|\lambda) \cdot r_t^{(l)}(i)/f(O^{(l)}|\lambda)}{\sum_{l=1}^{L} \sum_{t=1}^{T^{(l)}} f(O^{(l)}, s_t^{(l)} = j, h_t^{(l)} = m|\lambda)/f(O^{(l)}|\lambda)} \tag{16}$$

for $i = 0, 1, 2, \cdots, p, j = 1, 2, \cdots, N$, and $m = 1, 2, \cdots, M$. It is seen that $\bar{r}_{jm}(i)$ is a normalized accumulative autocorrelation with each summation term properly weighted by its probability of occurrence, From $\bar{r}_{jm}(i)$, $i = 0, 1, 2, \cdots, p$, one can solve a set of normal equations and obtain the corresponding autoregressive coefficient vector [10] which is exactly the required $\bar{\mathbf{a}}_{jm}$ vector in the Gaussian autoregressive density for the mth mixture component in the jth state. The new autocorrelation vectors of the autoregression coefficients can then be easily calculated using (10).

Equation (16) is explicit for the GAM case. For the PGAM case, where a nearest neighbor partitioning is involved, the reestimation transformation of the autocorrelation coefficients slightly differs from (16). Particularly, for PGAM,

$$\bar{r}_{jm}(i) = \frac{\sum_{l=1}^{L} \sum_{t=1}^{T^{(l)}} f(O^{(l)}, s_t^{(l)} = j|\lambda) \cdot r_t^{(l)}(i) \cdot q_{jm}(o_t^{(l)})/f(O^{(l)}|\lambda)}{\sum_{l=1}^{L} \sum_{t=1}^{T^{(l)}} f(O^{(l)}, s_t^{(l)} = j|\lambda)/f(O^{(l)}|\lambda)} \tag{17}$$

where $q_{jm}(o_t)$ is a nearest neighbor function defined as

$$q_{jm}(o_t) = \begin{cases} 1, & \text{if } b_{jm}(o_t) = \max\limits_{i=1,2,\cdots,M} b_{ji}(o_t) \\ 0, & \text{otherwise} \end{cases} \tag{18}$$

Therefore, in PGAM, only those autocorrelation vectors in the correct region are averaged, and each individual region is an independent entity. After the new set of autocorrelation vectors are obtained, the rest of the procedure remains the same as GAM.

It should be noted that the positive definiteness of the autocorrelation matrix is maintained in the reestimation transformation and thus a stable solution to the normal equations is guaranteed.

C. Forward Probabilities, Backward Probabilities, and Numerical Considerations

The computational complexity of the reestimation algorithm is significantly reduced by using the forward and backward likelihood (or probability) [4]. We shall drop the superscript for the sequence number for simplicity. The forward likelihood $\alpha_t(i)$ is defined as

$$\alpha_t(j) \triangleq f(o_1, o_2, \cdots, o_t, s_t = j|\lambda) = \sum_{i=1}^{N} \alpha_{t-1}(i)\, a_{ij} b_j(o_t), \tag{19}$$

with $\alpha_0(j) = \pi_j$, and the backward likelihood is

$$\beta_t(j) \triangleq f(o_{t+1}, o_{t+2}, \cdots, o_T|s_t = j, \lambda) = \sum_{i=1}^{N} \beta_{t+1}(i)\, \alpha_{ji} b_i(o_{t+1}). \tag{20}$$

with $\beta_T(j) = 1$ for all j.

The recursive nature of $\alpha_t(j)$ and $\beta_t(j)$ implies a trellis structure which is the key to computation reduction [11]. Many quantities required in the reestimation transformation can be easily expressed in terms of $\alpha_t(i)$ and $\beta_t(i)$. In particular,

$$f(O, s_t = i|\lambda) = \alpha_t(i)\, \beta_t(i), \tag{21}$$

$$f(O|\lambda) = \sum_{i=1}^{N} \alpha_t(i)\, \beta_t(i) = \sum_{i=1}^{N} \alpha_T(i), \tag{22}$$

$$f(O, s_{t-1} = i, s_t = j|\lambda) = \alpha_{t-1}(i)\, a_{ij} b_j(o_t)\, \beta_t(j), \tag{23}$$

and

$$f(O, s_t = j, h_t = m|\lambda) = \sum_{i=1}^{N} \alpha_{t-1}(i)\, a_{ij} c_{jm} b_{jm}(o_t)\, \beta_t(j). \tag{24}$$

These quantities are usually very small and a scaling scheme is required to prevent underflow problems. We use essentially the same scaling scheme as in [11] but write each term explicitly for ease of computer programming. Let

$$\phi_\tau = \sum_{i=1}^{N} \sum_{j=1}^{N} \alpha'_{\tau-1}(i)\, a_{ij} b_j(o_\tau), \tag{25}$$

and

$$\alpha'_t(i) = \frac{\alpha_t(i)}{\prod_{\tau=1}^{t} \phi_\tau}. \tag{26}$$

Similarly, we scale $\beta_t(i)$ by

$$\beta'_t(i) = \frac{\beta_t(i)}{\prod_{\tau=t+1}^{T} \phi_\tau}. \tag{27}$$

Clearly,

$$\phi_\tau = \sum_{j=1}^{N} \alpha_\tau(j) \Big/ \sum_{j=1}^{N} \alpha_{\tau-1}(j)$$

and hence

$$\prod_{\tau=1}^{T} \phi_\tau = \sum_{j=1}^{N} \alpha_T(j) = f(\boldsymbol{O}|\lambda) \tag{28}$$

since $\Sigma_{j=1}^{N} \alpha_0(j) = 1$. Then,

$$\alpha_t'(i)\,\beta_t'(i) = \frac{\alpha_t(i)\,\beta_t(i)}{\left(\prod_{\tau=1}^{t} \phi_\tau\right) \cdot \left(\prod_{\tau=t+1}^{T} \phi_\tau\right)} = \frac{f(\boldsymbol{O}, s_t = i|\lambda)}{f(\boldsymbol{O}|\lambda)}. \tag{29}$$

Similarly,

$$f(\boldsymbol{O}, s_{t-1} = i, s_t = j|\lambda)/f(\boldsymbol{O}|\lambda) = \alpha_{t-1}'(i)\, a_{ij} b_j(\boldsymbol{o}_t)\, \beta_t'(j)/\phi_t, \tag{30}$$

and

$$f(\boldsymbol{O}, s_t = j, h_t = m|\lambda)/f(\boldsymbol{O}|\lambda) = \sum_{i=1}^{N} \alpha_{t=1}'(i)\, a_{ij} c_{jm} b_{jm}(\boldsymbol{o}_t)\, \beta_t'(j)/\phi_t. \tag{31}$$

It is understood that (24) and (31) are explicit for the GAM case. For the PGAM case, incorporation of the nearest neighbor function q_{jm} as defined in (18) is necessary. This scaling scheme effectively prevents the underflow and overflow problems that often arise in the reestimation algorithm. Also, the scaling constant in (12) is usually omitted as it has no effect on the transformation equations.

III. Implementation of the Model Estimation Procedure

The procedure for estimating the HMM model parameters, outlined in Section II, has been shown to be very sensitive to initial estimates of several of the model parameters [12], [13]. A crucial prerequisite of the entire estimation procedure is a reliable and meaningful method for initialization. Obtaining such reliable initial estimates is often nontrivial since they are strongly affected by the prescribed Markov chain constraints. For left-to-right Markov models which have been shown to be extremely useful in speech modeling, a modified training procedure was developed in which very good initial estimates of model parameters were obtained via a segmental k-means procedure, and then the formal reestimation algorithm was used as a model refinement tool [13]. (A segmental k-means procedure is a k-means clustering procedure operated over properly selected segments of the training sequence.) A block diagram of this modified training procedure is given in Fig. 1. An initial model λ is assumed. This initial model can be chosen via a variety of procedures including random initial guesses of model parameters (subject to the required stochastic constraints). Based on the initial model, each of a set of L training sequences is segmented into the maximum likelihood state sequence (via a Viterbi decoding procedure [14]). For each state of the model, a k-means procedure, specifically the LPC vector quantizer design algorithm [17], clusters all the observation vectors within that state into a set of M clusters, based on a nearest neighbor classifier, using the distance measure of (9). Based on the vectors within each cluster, the initial estimates of the parameters of B are given as

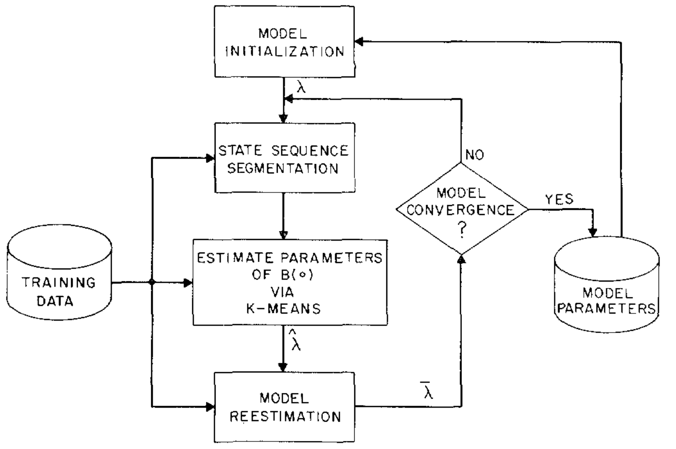


Fig. 1. Block diagram of model training procedure.

$$\hat{c}_{jm} = \frac{\text{number of vectors in cluster } m, \text{ state } j}{\text{number of vectors in state } j}$$

$\hat{\mathbf{a}}_{jm}$ = LPC vector corresponding to the centroid (of the normalized autocorrelation) of cluster m, state j.

The transition coefficients, a_{ij}, are not modified in the segmental k-means prcedure. The new model $\hat{\lambda} = (\pi, A, \hat{B})$ is used as the initial estimate for the model reestimation algorithm which leads to a new $\bar{\lambda}$. A check on model convergence, in which the new model, $\bar{\lambda}$, is compared to the previous model, λ, is used to determine if the model estimation has converged. The way in which convergence is determined is to generate, via Monte Carlo methods, a set of observation sequences, $\bar{\boldsymbol{O}}$, from the new model, $\bar{\lambda}$, and then to calculate the log likelihoods of $\bar{\boldsymbol{O}}$ given both the old and new models [15]. If the difference in the log likelihoods (normalized by the total number of observations in $\bar{\boldsymbol{O}}$) is sufficiently small, then model convergence is assumed. If no convergence is obtained, a new iteration of the training procedure is carried out with $\lambda = \bar{\lambda}$. Practically we have found that rapid convergence is obtained

In the following, we shall describe an example that demonstrates the effectiveness of the above estimation procedure, particularly for discrete speech utterances such as isolated digits. Our presentation focuses on left-to-right models [11], but the methodology can be easily extended to general cases.

A. An Example

To illustrate the use of the training procedure we used the following training data. For a vocabulary of 10 words

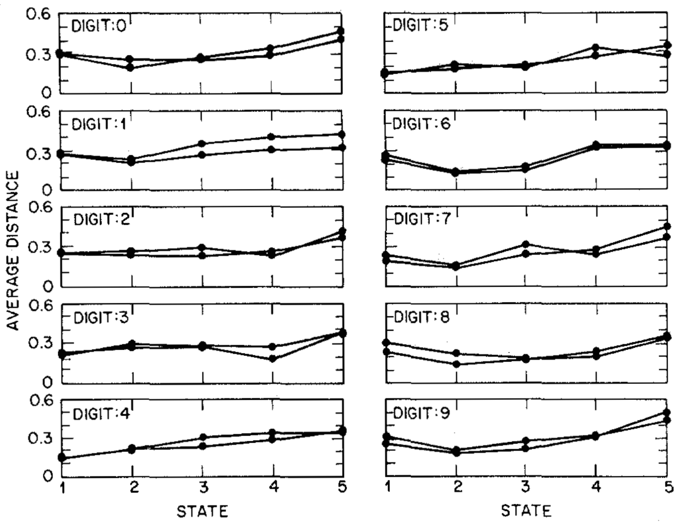


Fig. 2. Average LPC distance scores versus state number for each digit and model for the GAM mixture density models.

(the digits), the training procedure of Fig. 1 was used on two independent sets of training data with about 50 training sequences (from 50 different talkers) per set, where each training sequence consisted of about 40 frames of LPC vectors. Thus a total of about 2000 observations (frames of LPC vectors) were available for the segmental k-means algorithm. The initial model, for each training set, was a previous model based on a cepstral representation of the signal [13] so the initial state sequence segmentation was already a very good reflection of the time-varying statistical characteristics present in the sequences.

The target model was a 5 state left to right HMM [11] with a 5 term mixture autoregressive density in each state for each set of training sequences (i.e., two independent models were created for each word). The distance measure was the distance of (9). After model convergence the average distance of the state-segmented observation (training) sequence, in each state of each model, was monitored and a plot of these average distances, for the GAM case, is given in Fig. 2 for each of the 10 digits. The two curves, for each digit, represent results for the two models created from the training data. Although there is a fairly broad range of average distance scores (from a low of 0.14 to a high of 0.50), a general trend to having higher distance scores for the last state than for earlier states is seen. This is due to both the broader range of sounds that occur at the ends of isolated words than at other points within the word, and the fact that, in left-to-right models, the probability of staying in any state, except the last, is inherently an exponentially decreasing functio of the duration (recall that $a_{ii} < 1$ for $i \neq 5$ in the current 5 state model). It can also be seen from Fig. 2 that the average distances for the two independent models for each word are reasonably close for all states and for most of the words. This result can be attributed to the close linking, for isolated words, between states and specific sounds (phonemes, syllables, etc.) in the word.

The model reestimation procedure was used with the initial (unsegmented) training sequences and the model obtained from the segmental k-means procedure. Again we monitored the average model distance for the entire training set. The model reestimation procedure often converged rapidly and produced only a slight change in the average distance score from that provided by the model obtained from the segmental k-means algorithm. In other words, the segmental k-means algorithm was capable of providing a model with essentially the same likelihood as the one provided by the reestimation procedure for speech signals.

This result has the following significance even though its scope is limited by the fact that left-to-right models were used. First, speech signals, often considered as quasi-stationary processes, do manifest *sequentially* changing characteristics and each short term snapshot of the signal can be effectively analyzed by such methods as autoregression. The sequential nature of the characteristic changes of speech, however, can be adequately represented by simple segment registration (which cuts a speech utterance into a required number of segments and then registers each segment as a state). This segmental nature may not require sophisticated stochastic modeling particularly when the signal under analysis has rather well defined starting and ending characteristics like isolated digits. Second, the reestimation transformation outlined above does guarantee a local optimum and the use of the segmental k-means algorithm leads to a meaningful model reestimate of *speech signals*.

Although the results given in Fig. 2 are for the GAM representation, essentially identical results were obtained for the PGAM representation.

B. Incorporation of Model State Duration and Log Energy Estimates

For speech recognition applications, it is often desirable to incorporate state duration and energy information in the model. Based on the final segmentation of the training observation sequences into states, histograms of the average state duration and average log energy were measured as follows. For the state duration histograms, the length of time l spent in state j was determined for each training sequence and for each state. The state duration probability was defined as

$p_j(l/T)$ = probability of being in state j for exactly (l/T) of the sequence, where T is the number of observations in the sequence.

For each word, model, and state, the quantity $p_j(l/T)$ was estimated (via a simple counting procedure on the training data) for 25 values of l/T from 0 to 1.0. It is important to note that this is a particular implementation for the left-to-right model in which right-to-left paths are disallowed. For general model implementation, the length normalization by T is not necessary and l should be interpreted as the length of consecutive state segments.

In a similar manner, histograms of log energy for each state of each model were estimated from the training data in the following way. The log energy contour for each training sequence was normalized to a 0 dB peak and individual frame log energies were quantized into 3 dB wide bins (i.e., a 75 dB dynamic range). The log energy probability was then defined as

$w_j(\tilde{\epsilon})$ = probability of being in state j and having a quantized log energy value of $\tilde{\epsilon}$.

For each word, model, and state, the quantity $w_j(\tilde{\epsilon})$ was estimated via a simple counting procedure on the training data.

Calculation of the likelihood thus involves evaluation of the following equivalent distance (besides looking up the state transition probability matrix)

$$d(\boldsymbol{x}, l/T, \tilde{\epsilon}; \mathbf{a}_{jm}, p_j, w_j) = \hat{K}\left[\frac{1}{K}\delta(\boldsymbol{x}; \mathbf{a}_{jm}) - 1\right] - \frac{\gamma_D}{T}\log p_j(l/T) - \gamma_E \log w_j(\tilde{\epsilon}) \tag{32}$$

where $\hat{K}$ is the effective length of each data vector, l/T is the fraction of the word spent in state j accumulated up to the observation of $\boldsymbol{x}$, $\tilde{\epsilon}$ is the quantized log energy of the current frame, and γ_D and γ_E are experimentally determined positive scaling constants for the log probabilities of the duration and the log energy respectively.

The effective length $\hat{K}$ needs further explanation. Speech signal analysis is generally performed on frames of K samples, with consecutive frames being taken after a shift of K_s samples. A speech signal of L_T samples is hence segmented into L_T/K_s frames. These frames are overlapped if the analysis frame size K satisfies $K > K_s$. Then, $\boldsymbol{x}$ is used to represent K samples of the speech signal and $[\delta(\boldsymbol{x}; \mathbf{a}_{jm})/K] - 1$ becomes the well known likelihood ratio distortion measure. If we denote this quantity by $d_{LR}(\boldsymbol{x}; \mathbf{a}_{jm})$ (12) becomes

$$b_{jm}(\boldsymbol{x}) = (2\pi)^{-K/2} \exp(-K/2) \exp\left\{-\frac{K}{2} d_{LR}(\boldsymbol{x}; \mathbf{a}_{jm})\right\}. \tag{33}$$

It is therefore clear that $d_{LR}(\boldsymbol{x}, \mathbf{a}_{jm})$ can be regarded as the average cross entropy [16] *per sample* between the observed vector $\boldsymbol{x}$ and an autoregressive source characterized by $\mathbf{a}_{jm}$. Using $\{\boldsymbol{x}_i\}$ and $\{\mathbf{a}_i\}$, $i = 1, 2, \cdots, L_T/K_s$, to denote the L_T/K_s frames of speech data and the sequence of autoregressive sources for comparison respectively, we note that the cross entropy between $\{\boldsymbol{x}_i\}$ and $\{\mathbf{a}_i\}$, with a memoryless vector source assumption, is

$$K_s \sum_{i=1}^{L_T/K_s} d_{LR}(\boldsymbol{x}_i; \mathbf{a}_i) = K_s \sum_{i=1}^{L_T/K_s} \left[\frac{1}{K}\delta(\boldsymbol{x}_i; \mathbf{a}_i) - 1\right] \tag{34}$$

if the original time scale of the speech samples is to be maintained, independent of the analysis length K. This becomes more crucial when an underlying Markov chain, instead of a memoryless source, is assumed, since the transition structure contributes to the cross entropy. $\hat{K}$ in (32) thus allows adjustment on the relative cross entropy contributions between spectral parameters and the Markov chain. The relative rate of information in the model contributed by spectral, durational, and energy parameters, respectively, can therefore be adjusted by the three scaling factors, $\hat{K}$, γ_D, and γ_E.

IV. Isolated Digit Recognition Experiments

One direct application of the above modeling/estimation technique is in isolated digit recognition. A framework of such an application can be found in [11].

To evaluate the performance of the HMM recognizer with the GAM and PGAM mixture densities, a series of experiments was run using a database of isolated digits recorded over standard dialed-up telephone lines. Four sets of spoken digits were used. These consisted of the following:

DIG 1—100 talkers (50 male, 50 female), 1 replication of each digit by each talker [18]. The nominal bandwidth of these recordings was 100–3200 Hz.

DIG 2—Same 100 talkers and recording conditions as DIG 1; recordings made several weeks after those of DIG 1.

DIG 3—100 new talkers (50 male, 50 female), 1 averaged occurrence of each digit by each talker obtained from averaging a pair of robust tokens of the digit [19], [20]. The transmission conditions (i.e., analog front end, filter cutoff frequencies, etc.) differed slightly from those used in recording the DIG1 and DIG2 databases.

DIG 4—A second group of 100 new talkers (50 male, 50 female), 20 recordings of each digit by each talker [21]. A random sampling of 1 of the recordings of each digit by each talker was used. The transmission conditions differed substantially from those used in recording the other databases. The nominal bandwidth of these recordings was 200–3200 Hz.

Each of the 4 sets of digits contained 1000 digits. For training the models, only the digits in set DIG 1 or DIG 4 were used; for testing and evaluating the performance of the recognizer, each of the 4 sets of digits were used.

A. The HMM Recognizer

For both training and testing the recognizer, the front end analysis conditions consisted of the following:

sampling rate—6.67 kHz
analysis method—LPC autocorrelation method, Hamming window
preemphasis—first order $(1 - 0.95\, z^{-1})$
analysis frame size—$K = 300$ samples (45 ms)
analysis shift size—$K_s = \hat{K} = 100$ samples (15 ms)
LPC order—$p = 8$.

For training the recognizer, for each digit in the training set, the 100 versions were first clustered into 2 sets (of about 50 versions each), and for each set a left-to-right

TABLE I
COMPARISON OF PERFORMANCE OF THE HMM RECOGNIZER AS A FUNCTION OF THE TRAINING SET, MODEL TYPE, YD, AND YE

Training Set	Model Type	γ_D	γ_E	Average Digit Error Rate (%)				
				DIG1	DIG2	DIG3	DIG4	Test Set Average
DIG1	GAM	0.0	0.0	1.2	3.3	5.8	9.2	6.1
DIG1	GAM	0.0	3.0	0.9	2.3	4.3	6.0	4.2
DIG1	GAM	10.0	1.0	0.3	2.1	3.7	5.2	3.67
DIG1	GAM	10.0	3.0	0.3	1.8	3.4	4.1	3.1
DIG1	GAM	10.0	10.0	0.4	1.2	3.5	4.6	3.1
DIG4	GAM	0.0	0.0	7.1	7.4	4.1	1.7	6.2
DIG4	GAM	0.0	3.0	4.9	3.9	3.1	0.8	3.98
DIG4	GAM	10.0	1.0	5.7	4.3	2.7	0.5	4.23
DIG4	GAM	10.0	3.0	4.9	3.5	2.9	0.5	3.7
DIG4	GAM	10.0	10.0	3.9	2.9	3.7	0.8	3.5
DIG1	PGAM	0.0	0.0	1.1	4.1	5.3	8.3	5.9
DIG1	PGAM	0.0	3.0	1.0	2.8	3.8	6.4	4.67
DIG1	PGAM	10.0	1.0	0.7	2.1	3.4	6.0	3.83
DIG1	PGAM	10.0	3.0	0.7	2.0	3.1	5.9	3.67
DIG1	PGAM	10.0	10.0	0.5	1.7	3.6	5.0	3.43
DIG4	PGAM	0.0	0.0	7.0	7.3	4.0	1.6	6.1
DIG4	PGAM	0.0	3.0	5.3	4.9	2.9	1.4	4.33
DIG4	PGAM	10.0	1.0	5.3	4.2	3.2	1.1	4.23
DIG4	PGAM	10.0	3.0	4.9	3.4	2.3	0.8	3.53
DIG4	PGAM	10.0	10.0	3.4	2.9	3.7	1.0	3.33
DIG1	CEP/DC	10.0	–	0.1	0.7	2.8	4.2	2.57
DIG4	CEP/DC	10.0	–	2.5	1.7	2.1	0.8	2.1

HMM was designed using the procedure given in Section III. Thus, the output of the training procedure was a set of 2 HMM's per digit. Each HMM had $N = 5$ states, with $M = 5$ mixture densities per state for both the GAM and PGAM models.

For testing the recognizer, each of the 20 models (2 models $\times$ 10 digits) was scored using a Viterbi alignment procedure to give the optimal state sequence alignment of the unknown observation sequence to the input model. The distance measure of (32) was used to score the optimal alignment path.

B. Experimental Results

Several experiments were run using different training sets, model types (GAM, PGAM), and choices for γ_D and γ_E, the scaling constants in the distance measure, and the results of these experiments are given in Table I. Based on preliminary experimentation it was found that a value of $\gamma_D = 10.0$ (with $\gamma_E = 0$) gave the best performance when duration (without energy) was incorporated into the distance measure. As such the results in Table I primarily show the effects of different values of γ_E when γ_D was either 0 (no duration used) or 10.0 (the best duration scaling value).

The results given in Table I show the following:

1) Incorporation of duration and energy into the distance measure improve recognizer performance.

2) Performance of the two model types (GAM and PGAM) was almost identical across all test conditions.

3) Performance with both training sets (DIG1 and DIG4) was almost identical.

To gain perspective into how the performance of these recognizer compares to that of previous HMM recognizer, the bottom two lines of Table I summarize results from previous work with a cepstral model using diagonal covariance matrices (CEP/DC) [13]. It can be seen that the best GAM performance is about 1 percent worse than the CEP/DC model performance, and the best PGAM performance is about 1.2 percent worse than the CEP/DC model performance. (The CEP/DC recognizer performance is equivalent to the best performance of a current DTW recognizer [13].)

C. Computational Complexity

The computation required in the recognizer, using a Viterbi decoding algorithm, is

$$C_V = N \cdot T \cdot (p + 1)\, M \cdot V \qquad \text{multiplication/addition operations}$$

for a V word vocabulary, average word length T frames, N states per model, M mixtures per state, and pth order LPC representation.

A standard DTW recognizer requires

$$C_{DTW} = Q \cdot V \cdot \frac{T^2}{3} (p + 1) \qquad \text{multiplication/addition operations}$$

for a Q template per word system. The ratio of computation is thus

$$\text{RATIO} = \frac{C_V}{C_{DTW}} = \frac{NM}{QT/3}$$

which for $N = M = 5$, $Q = 12$, $T = 40$, gives

$$\text{RATIO} = \frac{5}{32} = \frac{1}{6.4}$$

i.e., a 6.4 to 1 reduction in computation is achieved for the LPC HMM recognizer over the standard DTW recognizer.

The computational advantage of LPC HMM over CEP/DC is mainly in the distortion computation. LPC HMM requires computation of a dot product as in (9), while CEP/DC requires computation of a weighted Euclidean distance. Practically speaking, a reduction in computation of from 2 to 4 can be achieved.

V. DISCUSSION AND SUMMARY

We have shown that the ubiquitous LPC analysis technique can be consistently welded into the general hidden Markov model methodology. The resultant autoregressive HMM's are powerful for modeling time-varying signal sources. For short speech signals such as discrete digit utterances, however, such an extensive stochastic modeling may not be necessary. A simple segment registration procedure may be adequate in coping with the sequentially changing characteristics of speech. We have also demonstrated how the two procedures, namely the segmental k-means and the probabilistic Markov chain, can be successfully combined. Since the segmental k-means procedure gives good initial model estimates, the models resulting from reestimation always worked well in practice.

For speaker independent, isolated digit recognition, the results given above show that both the GAM and PGAM models give comparably good performance in the *current* framework, although not as good as the more conventional continuous Gaussian mixture density models based upon cepstral representations of the signal [13]. The small degradation, we believe, can be attributed to the less reliable covariance estimate. [Note that the basis density function of (12) has an implicit covariance matrix (of the speech data vector, not representations derived therefrom) which is assumed to satisfy the autoregression condition of (7).] The assumption that the observation vector (speech frame) $\boldsymbol{x}$ is Gaussian autoregressive may be adequate in stationary spectral analysis of $\boldsymbol{x}$. It, however, may not be an accurate one when dealing with multiple observations that are produced by a number of different speakers and registered in the same state in the procedure. It is generally observed, particularly for the last state of the model, that the (equivalent) distances from each vector to its corresponding centroid are fairly diverse and are inconsistent with the theoretically predicted chi-square distribution. Such a discrepancy cannot be resolved by increasing the number of mixture terms although the average distance is expected to decrease in doing so. We have tried models with a higher number (more than 5, especially for the last state) of mixture terms based upon the same training data. The likelihood in the modeling procedure was significantly increased, but no performance improvement in recognition was obtained. Analysis on the error patterns showed that the unreliable covariance estimates lead to more increase in likelihood (during recognition) for incorrect words than for the correct one, thus resulting in recognition performance degradation. The effect is by no means a large one, as witnessed from the recognition results, but it does account for about a 1 percent increase in error rate over the standard mixture density model in which the vectors, often spectral representations, in each state are modelled with simple Gaussian mixture distributions characterized by mean vectors and covariance matrices (of the vector representation), estimates of which are normally reliable.

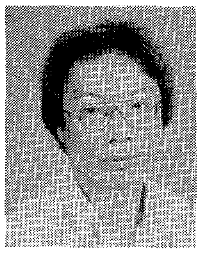
Based on the above discussion, it is believed that the GAM and PGAM HMM models would perhaps have more applicability to speaker dependent digit recognizers and to talker recognition systems because, in such cases, there would be considerably less variability of the LPC vectors in each state and, therefore, the bias due to large variability would be greatly reduced or eliminated. As yet we have not verified this conjecture, but work is in progress along these lines.

We should point out again that these experimental results are related to a specific isolated word recognizer that employs highly constrained left-to-right models. The methodology presented here, nevertheless, is a general, versatile one and its effectiveness in other applications deserves careful, though straightforward, evaluation.

In summary we have presented a consistent framework for combining mixture density LPC models with hidden Markov models. We have shown how to train these models using a segmental k-means classifier. Recognition results, on a speaker independent vocabulary of isolated digits, were good, but somewhat inferior to those based on mixture densities of LPC derived parameter vectors. Reasons for this small degradation in performance were given along with suggestions as to areas in which the LPC mixture models would be more effective.

References

[1] A. B. Poritz, "Linear predictive hidden Markov models and the speech signal," *Proc. ICASSP '82*, pp. 1291-1294, Paris, France, May 1982.
[2] B. H. Juang, "On hidden Markov model and dynamic time warping for speech recognition—A unified view," *AT&T BLTJ*, vol. 63, no. 7, pp. 1213-1243, Sept. 1984.
[3] L. E. Baum and J. A. Eagon, "An inequality with applications to statistical prediction for functions of Markov processes and to a model for ecology," *Bull. Amer. Math. Soc.*, vol. 73, pp. 360-363, 1963.
[4] L. E. Baum, T. Petrie, G. Soules, and N. Weiss, "A minimization technique occurring in the statistical analysis of probabilistic functions of Markov chains," *Ann. Math. Stat.*, vol. 41, pp. 164-171, 1970.
[5] L. R. Liporace, "Maximum likelihood estimation for multivariate observations of Markov sources," *IEEE Trans. Info. Theory*, vol. IT-28, pp. 729-734, Sept. 1982.
[6] K. Fan, "Les fonction définies—Positives et les fonctions complètement monotones," *Mèmonial des Sciences Math.*, vol. CXIV, 1950.
[7] B. H. Juang, S. E. Levinson, and M. M. Sondhi, "Maximum likelihood estimation for multivariate mixture observations of Markov chains," to be published.
[8] H. W. Sorenson and D. L. Alspach, "Recursive Baysian estimation using Gaussian sums," *Automatica*, vol. 7, pp. 465-479, 1971.
[9] F. Itakura, "Speech analysis and synthesis systems based on statistical method," doctoral dissertation, Dept. Eng., Nagoya Univ., Nagoya, Japan, 1972.
[10] J. D. Markel and A. H. Gray, Jr., *Linear Prediction of Speech*. New York: Springer-Verlag, 1976.
[11] S. E. Levinson, L. R. Rabiner, and . M. Sondhi, "An introduction to the application of the theory of probabilistic functions of a Markov process to automatic speech recognition," *B.S.T.J.*, vol. 62, no. 4, part 1, pp. 1035-1074, Apr. 1983.
[12] L. R. Rabiner, B. H. Juang, S. E. Levinson, and M. M. Sondhi, "Some properties of continuous hidden Markov model representations," *AT&T Tech. J.*, vol. 64, part 1, pp. 1251-1270, July-Aug. 1985.
[13] ——, "Recognition of isolated digits using hidden Markov models with continuous mixture densities," *AT&T Tech. J.*, vol. 64, part 1, pp. 1211-1234, July-Aug. 1985.
[14] G. D. Forney Jr., "The Viterbi algorithm," *Proc. IEEE*, vol. 61, pp. 268-278, Mar. 1973.
[15] B. H. Juang and L. R. Rabiner, "A probabilistic distance measure for hidden Markov models," *AT&T Bell Lab. Tech. J.*, 1985.
[16] R. M. Gray, A. H. Gray, Jr., G. Rebolleds, and J. E. Shore, "Rate distortion speech coding with a minimum discrimination information distortion measure," *IEEE Trans. Info. Theory*, vol. IT-27, no. 6, pp. 708-721, Nov. 1981.
[17] A. Buzo, A. H. Gray, Jr., R. M. Gray, and J. D. Markel, "Speech coding based upon vector quantization," *IEEE Trans. Acoust., Speech, Signal Processing*, vol. ASSP-28, pp. 562-574, Oct. 1980.
[18] L. R. Rabiner, S. E. Levinson, A. E. Rosenberg, and J. G. Wilpon, "Speaker independent recognition of isolated words using clustering techniques," *IEEE Trans. Acoust., Speech, Signal Processing*, vol. ASSP-27, no. 4, pp. 336-349, Aug. 1979.
[19] L. R. Rabiner and J. G. Wilpon, "A simplified robust training procedure for speaker trained, isolated word recognition systems," *J. Acoust. Soc. Amer.*, vol. 68, no. 5, pp. 1271-1276, Nov. 1980.
[20] J. G. Wilpon, L. R. Rabiner, and A. Bergh, "Speaker independent isolated word recognition using a 129-word airline vocabulary," *J. Acoust. Soc. Amer.*, vol. 72, no. 2, pp. 390-396, Aug. 1982.
[21] A. E. Rosenberg, K. L. Shipley, and D. E. Bock, "A speech data base facility using a computer controlled cassette tape deck," *J. Acoust. Soc. Amer.*, suppl. 7, vol. 72, p. 580, Fall 1982.
[22] M. O. Dunham and R. M. Gray, "An algorithm for the design of labeled-transition finite-state vector quantizers," *IEEE Trans. Comm.*, vol. COM-33, no. 1, Jan. 1985.



Biing-Hwang Juang (S'79–M'81) was born in 1951. He received the B.Sc. degree in electrical engineering from the National Taiwan University, Taipei, in 1973 and the M.Sc. and Ph.D. degrees in electrical and computer engineering from the University of California, Santa Barbara, in 1979 and 1981, respectively.

In 1978, he joined the Speech Communications Research Laboratory, Santa Barbara, and was involved with research work on vocal tract modeling. In 1979, he became affiliated with Signal Technology, Inc., Santa Barbara, where his research work was in the areas of speech coding and speech interference suppression. Since 1982, he has been with AT&T Bell Laboratories, Murray Hill, NJ. His current research interests include speech recognition, coding and stochastic processes.

Lawrence R. Rabiner (S'62–M'67–SM'75–F'75) was born in Brooklyn, New York, on September 28, 1943. He received the S.B. and S.M. degrees simultaneously in June 1964, and the Ph.D. degree in electrical engineering in June 1967, all from the Massachusetts Institute of Technology, Cambridge.

From 1962 through 1964 he participated in the cooperative plan in electrical engineering at Bell Laboratories, Whippany, and Murray Hill, New Jersey. He worked on digital circuitry, military communications problems, and problems in binaural hearing. Currently he is engaged in research on speech recognition and digital signal processing techniques at Bell Laboratories, Murray Hill. He is coauthor of the books *Theory and Application of Digital Signal Processing* (Prentice-Hall, 1975), *Digital Processing of Speech Signals* (Prentice-Hall, 1978) and *Multirate Digital Signal Processing* (Prentice-Hall, 1983).

Dr. Rabiner is a member of Eta Kappa Nu, Sigma Xi, Tau Beta Pi, the National Academy of Engineering, and a Fellow of the Acoustical Society of America.