Markov Processes on Curves

Lawrence K. Saul and Mazin G. Rahim
AT&T Labs - Research
180 Park Avenue, E-171
Florham Park, NJ 07932

Abstract

We study the classification problem that arises when two variables—one continuous ($\boldsymbol{x}$), one discrete (s)—evolve jointly in time. We suppose that the vector $\boldsymbol{x}$ traces out a smooth multidimensional curve, to each point of which the variable s attaches a discrete label. The trace of s thus partitions the curve into different segments whose boundaries occur where s changes value. We consider how to learn the mapping between the trace of $\boldsymbol{x}$ and the trace of s from examples of segmented curves. Our approach is to model the conditional random process that generates segments of constant s along the curve of $\boldsymbol{x}$. We suppose that the variable s evolves stochastically as a function of the arc length traversed by $\boldsymbol{x}$. Since arc length does not depend on the rate at which a curve is traversed, this gives rise to a family of Markov processes whose predictions are invariant to nonlinear warpings (or reparameterizations) of time. We show how to estimate the parameters of these models—known as Markov processes on curves (MPCs)—from labeled and unlabeled data. We then apply these models to two problems in automatic speech recognition, where $\boldsymbol{x}$ are acoustic feature trajectories and s are phonetic alignments.

1 Introduction

The automatic segmentation of continuous trajectories poses a challenging problem in machine learning. The problem arises whenever a multidimensional trajectory $\{\boldsymbol{x}(t)|t \in [0,\tau]\}$ must be mapped into a sequence of discrete labels $s_1 s_2 \ldots s_n$. A segmentation performs this mapping by specifying consecutive time intervals such that $s(t) = s_k$ for $t \in [t_{k-1}, t_k]$, thereby attaching the labels s_k to contiguous arcs along the trajectory. The learning problem is to discover such a mapping from labeled or unlabeled examples.

In this paper, we study this problem, paying special attention to the fact that curves have intrinsic geometric properties that do not depend on the rate at which they are traversed (do Carmo, 1976). Such properties include, for example, the total arc length and the maximum distance between any two points on the curve. Given a multidimensional trajectory $\{\boldsymbol{x}(t)|t \in [0,\tau]\}$, these properties are invariant to reparameterizations $t \rightarrow f(t)$, where $f(t)$ is any monotonic function that maps the interval $[0,\tau]$ into itself. Put another way, the intrinsic geometric properties of the curve are invariant to *nonlinear warpings of time*.

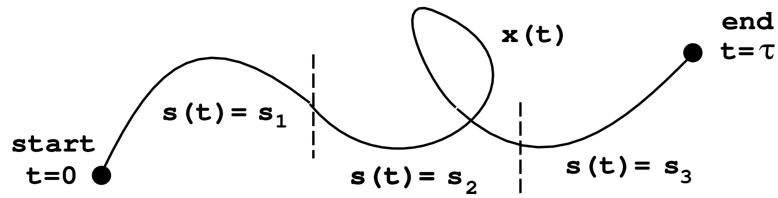

Figure 1: Two variables—one continuous ($\boldsymbol{x}$), one discrete (s)—evolve jointly in time. The trace of s partitions the curve of $\boldsymbol{x}$ into different segments whose boundaries occur where s changes value. Markov processes on curves model the conditional distribution, $\Pr[s|\boldsymbol{x}]$.

The study of curves requires some simple notions from differential geometry. As a matter of terminology, we refer to particular parameterizations of curves as trajectories. We regard two trajectories $\boldsymbol{x}_1(t)$ and $\boldsymbol{x}_2(t)$ as equivalent to the same curve if there exists a monotonically increasing function f for which $\boldsymbol{x}_1(t) = \boldsymbol{x}_2(f(t))$. (To be precise, we mean the same *oriented* curve: the direction of traversal matters.) Here, as in what follows, we adopt the convention of using $\boldsymbol{x}(t)$ to denote an entire trajectory as opposed to constantly writing out $\{\boldsymbol{x}(t)|t \in [0, \tau]\}$. Where necessary to refer to the value of $\boldsymbol{x}(t)$ at a particular moment in time, we use a different index, such as $\boldsymbol{x}(t_1)$.

Let us now return to the problem of automatic segmentation. Consider two variables—one continuous ($\boldsymbol{x}$), one discrete (s)—that evolve jointly in time. We assume that the vector $\boldsymbol{x}$ traces out a smooth multidimensional curve, to each point of which the variable s attaches a discrete label. Note that the trace of s yields a partition of the curve into different components; in particular, the boundaries of these components occur at the points where s changes value. We refer to such partitions as segmentations and to the regions of constant s as segments; see figure 1.

Our goal in this paper is to learn a probabilistic mapping between trajectories $\boldsymbol{x}(t)$ and segmentations $s(t)$ from labeled or unlabeled examples. Consider the random process that generates segments of constant s along the curve traced out by $\boldsymbol{x}$. Given a trajectory $\boldsymbol{x}(t)$, let $\Pr[s(t)\,|\,\boldsymbol{x}(t)]$ denote the conditional probability distribution over possible segmentations. Suppose that for any two equivalent trajectories $\boldsymbol{x}(t)$ and $\boldsymbol{x}(f(t))$, we have the identity:

$$\Pr[s(t)\,|\,\boldsymbol{x}(t)] = \Pr[s(f(t))\,|\,\boldsymbol{x}(f(t))]. \tag{1}$$

Eq. (1) captures a fundamental invariance—namely, that the probability that the curve is segmented in a particular way is independent of the rate at which it is traversed. In this paper, we study Markov processes with this property. We call them *Markov processes on curves* (MPCs) because for these processes it is unambiguous to write $\Pr[s\,|\,\boldsymbol{x}]$ without providing explicit parameterizations for the trajectories, $\boldsymbol{x}(t)$ or $s(t)$. The distinguishing feature of MPCs is that the variable s evolves as a function of the *arc length* traversed along $\boldsymbol{x}$, a quantity that is manifestly invariant to nonlinear warpings of time.

Invariances and symmetries play an important role in statistical pattern recognition because they encode prior knowledge about the problem domain (Duda & Hart, 1973). Most work on invariances has focused on spatial symmetries in vision. In optical character recognition, for example, researchers have improved the accuracy of automatically trained classifiers by incorporating invariances to translations, rotations, and changes of scale (Simard, LeCun, & Denker, 1993). This paper focuses on an invariance associated with pattern recognition in dynamical systems. Invariance to nonlinear warpings of time arises naturally in problems involving the segmentation of continuous trajectories. For example, in pen-based handwriting recognition, this invariance captures the notion that the shape of a letter does not depend on the rate at which it is penned. Likewise, in automatic speech recognition (Rabiner & Juang, 1993), this invariance can be used to model (approximately) the effects of speaking rate. Thus, in addition to being mathematically interesting in its own right, the principled handling of this invariance has important consequences for real-world applications of machine learning.

The main contributions of this paper are: (i) to postulate eq. (1) as a useful invariance for problems in statistical pattern recognition; (ii) to introduce MPCs as a family of probabilistic models that capture this invariance; (iii) to derive learning algorithms for MPCs based on the principle of maximum likelihood estimation; and (iv) to compare the performance of MPCs for automatic speech recognition versus that of hidden Markov models (Rabiner & Juang, 1993). In terms of previous work, our motivation most closely resembles that of Tishby (1990), who several years ago proposed a dynamical system approach to speech processing.

The organization of this paper is as follows. In section 2, we begin by reviewing some basic concepts from differential geometry. We then introduce MPCs as a family of continuous-time Markov processes that parameterize the conditional probability distribution, $\Pr[s \,|\, \boldsymbol{x}]$. The processes are derived from a set of differential equations that describe the pointwise evolution of s along the curve traced out by $\boldsymbol{x}$.

In section 3, we consider how to learn the parameters of MPCs in both supervised and unsupervised settings. These settings correspond to whether the learner has access to labeled or unlabeled examples. Labeled examples consist of trajectories $\boldsymbol{x}(t)$, along with their corresponding segmentations:

$$\{(\textsc{start}, 0) \rightarrow (s_1, t_1) \cdots\ (s_n, t_n) \rightarrow (\textsc{end}, \tau)\}. \tag{2}$$

The ordered pairs in eq. (2) indicate that $s(t)$ takes the value s_k between times t_{k-1} and t_k; the START and END states are used to mark endpoints. Unlabeled examples consist only of the trajectories $\boldsymbol{x}(t)$ and the boundary values:

$$\{(\textsc{start}, 0)\ \longrightarrow\ (\textsc{end}, \tau)\}. \tag{3}$$

Eq. (3) specifies only that the Markov process starts at time $t = 0$ and terminates at some later time τ. In this case, the learner must infer its own target values for $s(t)$ in order to update its parameter estimates. We view both types of learning as instances of maximum likelihood estimation and describe an EM algorithm for the more general case of unlabeled examples.

In section 4, we describe some simple extensions of MPCs that significantly increase their modeling power. We also compare MPCs to other probabilistic models of trajectory segmentation, such as hidden Markov models. We argue that MPCs are distinguished by two special properties: the mathematical invariance to nonlinear warpings of time, and the parameterization of a *segmentation* model $\Pr[s|\boldsymbol{x}]$, as opposed to a *synthesis* model $\Pr[\boldsymbol{x}|s]$.

Finally, in section 5, we apply MPCs to the problem of automatic speech recognition. In this setting, we identify the curves $\boldsymbol{x}$ with acoustic feature trajectories and the segmentations s with phonetic labelings and alignments. We present experimental results on two tasks—recognizing New Jersey town names and connected alpha-digits. On these tasks, we find that MPCs generally match or exceed the performance of comparably trained hidden Markov models. We conclude in section 6 by posing several open questions for future research.

2 Markov processes on curves

Markov processes on curves are based fundamentally on the notion of *arc length*. After reviewing how to compute arc lengths along curves, we show how they can be used to define random processes that capture the invariance of eq. (1).

2.1 Arc length

Let $g(\boldsymbol{x})$ define a $D \times D$ matrix for each point $\boldsymbol{x} \in \mathcal{R}^D$; in other words, to each point $\boldsymbol{x}$, we associate a particular $D \times D$ matrix $g(\boldsymbol{x})$. If $g(\boldsymbol{x})$ is non-negative definite for all $\boldsymbol{x}$, then we can use it as a *metric* to compute distances along curves. In particular, consider two nearby points $\boldsymbol{x}$ and $\boldsymbol{x} + \boldsymbol{dx}$ separated by the infinitesimal vector $\boldsymbol{dx}$. We define the squared distance between these two points as:

$$d\ell^2 \;=\; \boldsymbol{dx}^T g(\boldsymbol{x})\; \boldsymbol{dx}. \tag{4}$$

Arc length along a curve is the non-decreasing function computed by integrating these local distances. Thus, for the trajectory $\boldsymbol{x}(t)$, the arc length between the points $\boldsymbol{x}(t_1)$ and $\boldsymbol{x}(t_2)$ is given by:

$$\ell = \int_{t_1}^{t_2} dt\;\left[\dot{\boldsymbol{x}}^T g(\boldsymbol{x}(t))\,\dot{\boldsymbol{x}}\right]^{\frac{1}{2}}, \tag{5}$$

where $\dot{\boldsymbol{x}} = \frac{d}{dt}[\boldsymbol{x}(t)]$ denotes the time derivative of $\boldsymbol{x}$. Note that the arc length between two points is invariant under reparameterizations of the trajectory, $\boldsymbol{x}(t) \rightarrow \boldsymbol{x}(f(t))$, where $f(t)$ is any smooth monotonic function of time that maps the interval $[t_1, t_2]$ into itself.

In the special case where $g(\boldsymbol{x})$ is the identity matrix for all $\boldsymbol{x}$, eq. (5) reduces to the standard definition of arc length in Euclidean space. More generally, however, eq. (4) defines a non-Euclidean metric for computing arc lengths. Thus, for example, if the metric $g(\boldsymbol{x})$ varies as a function of $\boldsymbol{x}$, then eq. (5) can assign different arc lengths to the trajectories $\boldsymbol{x}(t)$ and $\boldsymbol{x}(t) + \boldsymbol{x}_0$, where $\boldsymbol{x}_0$ is a constant displacement.

2.2 States and lifelengths

The problem of segmentation is to map a trajectory $\boldsymbol{x}(t)$ into a sequence of discrete labels $s_1 s_2 \ldots s_n$. If these labels are attached to contiguous arcs along the curve of $\boldsymbol{x}$, then we can describe this sequence by a piecewise constant function of time, $s(t)$, as in figure 1. We refer to the possible values of s as *states*. In what follows, we introduce a family of random processes that evolve s as a function of the arc length traversed along the curve traced out by $\boldsymbol{x}$. These random processes are based on a simple premise—namely, that *the probability of remaining in a particular state decays exponentially with the cumulative arc length traversed in that state.* The signature of a state is the particular way in which it computes arc length.

To formalize this idea, we associate with each state i the following quantities: (i) a metric $g_i(\boldsymbol{x})$ that can be used to compute arc lengths, as in eq. (5); (ii) a decay parameter λ_i that measures the probability per unit arc length that s makes a transition from state i to some other state; and (iii) a set of transition probabilities a_{ij}, where a_{ij} represents the probability that—having decayed out of state i—the variable s makes a transition to state j. Thus, a_{ij} defines a stochastic transition matrix with zero elements along the diagonal and rows that sum to one: $a_{ii} = 0$ and $\sum_j a_{ij} = 1$. (We do not consider the possibility of self-transitions, as they give rise to artificial boundaries in otherwise indistinguishable segmentations.) Note that all these quantities—the metric $g_i(\boldsymbol{x})$, the decay parameter λ_i, and the transition probabilities a_{ij}—depend explicitly on the state i with which they are associated.

Together, these quantities can be used define a Markov process along the curve traced out by $\boldsymbol{x}$. In particular, let $p_i(t)$ denote the probability that s is in state i at time t, based on its history up to that point in time. A Markov process is defined by the set of differential equations:

$$\frac{dp_i}{dt} = -\lambda_i p_i \left[\dot{\boldsymbol{x}}^T g_i(\boldsymbol{x})\, \dot{\boldsymbol{x}}\right]^{\frac{1}{2}} + \sum_{j \neq i} \lambda_j p_j a_{ji} \left[\dot{\boldsymbol{x}}^T g_j(\boldsymbol{x})\, \dot{\boldsymbol{x}}\right]^{\frac{1}{2}}. \tag{6}$$

The right hand side of eq. (6) consists of two competing terms. The first term computes the probability that s decays out of state i; the second computes the probability that s decays into state i. Both probabilities are proportional to measures of arc length, and combining them gives the overall change in probability that occurs in the time interval $[t, t + dt]$. The process is Markovian because the evolution of p_i depends only on quantities available at time t; thus the future is independent of the past given the present.

Eq. (6) has certain properties of interest. First, note that summing both sides over i gives the identity $\sum_i dp_i/dt = 0$. This shows that p_i remains a normalized probability distribution: i.e., $\sum_i p_i = 1$ at all times. Second, suppose that we start in state i and do not allow return visits: i.e., $p_i = 1$ at $t = 0$ and $a_{ji} = 0$ for all j. In this case, the second term of eq. (6) vanishes, and we obtain a simple, one-dimensional linear differential equation for $p_i(t)$. It follows that the probability of remaining in state i decays exponentially (Papoulis, 1991) with the amount of arc length traversed by $\boldsymbol{x}$, where arc length is computed using the matrix $g_i(\boldsymbol{x})$. The decay parameter, λ_i, controls the typical amount of arc length traversed in state i; it may be viewed as an inverse lifetime or—to be more precise—an inverse *lifelength.* Finally, noting that arc length is a reparameterization-invariant quantity, we therefore observe that these dynamics capture the fundamental invariance of eq. (1).

2.3 Inference

Let a_{0i} denote the probability that the variable s makes an immediate transition from the START state—denoted by the zero index—to state i; put another way, this is the probability that the first segment belongs to state i. Given a trajectory $\boldsymbol{x}(t)$, the Markov process in eq. (6) gives rise to a conditional probability distribution over possible segmentations, $s(t)$. Consider the segmentation in which $s(t)$ takes the value s_k between times t_{k-1} and t_k, and let

$$\ell_k \;=\; \int_{t_{k-1}}^{t_k} dt\; \left[\dot{\boldsymbol{x}}^T g_{s_k}(\boldsymbol{x}(t))\,\dot{\boldsymbol{x}}\,\right]^{\frac{1}{2}} \tag{7}$$

denote the arc length traversed in state s_k. From eq. (6), we know that the probability of remaining in a particular state decays exponentially with this arc length. Thus, the conditional probability of this segmentation is given by (Papoulis, 1991):

$$\Pr[s|\boldsymbol{x}] = \left(\prod_{k=1}^{n} \lambda_{s_k} e^{-\lambda_{s_k}\ell_k}\right)\left(\prod_{k=0}^{n} a_{s_k s_{k+1}}\right), \tag{8}$$

where we have used s_0 and s_{n+1} to denote the START and END states of the Markov process. The first product in eq. (8) multiplies the probabilities that each segment traverses exactly its observed arc length. The second product multiplies the probabilities for transitions between states s_k and s_{k+1}. The leading factors of λ_{s_k} are included to normalize each state's duration model.

There are many important quantities that can be computed from the distribution, $\Pr[s|\boldsymbol{x}]$. Of particular interest is the most probable segmentation:

$$s^* \;=\; \arg\max_s \left\{\ln \Pr[s|\boldsymbol{x}]\right\}. \tag{9}$$

Given a particular trajectory $\boldsymbol{x}(t)$, eq. (9) calls for a maximization over all piecewise constant functions of time, $s(t)$. In practice, this maximization can be performed by discretizing the time axis and applying a dynamic programming (or forward-backward) procedure, analogous to the Viterbi decoder in HMMs (Viterbi, 1967). The resulting segmentations will be optimal at some finite temporal resolution, Δt. For example, let $\alpha_i(t)$ denote the log-likelihood of the most probable segmentation, ending in state i, of the subtrajectory up to time t. Starting from the initial condition $\alpha_i(0) = \ln[a_{0i}]$, we compute

$$\alpha_j(t+\Delta t) = \max_i \left\{\alpha_i(t) \;-\; \lambda_i \Delta t\left[\dot{\boldsymbol{x}}^T g_i(\boldsymbol{x})\,\dot{\boldsymbol{x}}\,\right]^{\frac{1}{2}} +\; \ln[\lambda_i a_{ij}](1-\delta_{ij})\right\}, \tag{10}$$

where δ_{ij} is the discrete delta function. Also, at each time step, let $\Psi_j(t+\Delta t)$ record the value of i that maximizes the right hand side of eq. (10). Suppose that the Markov process terminates at time τ. Enforcing the endpoint condition $s^*(\tau) = \text{END}$, we find the most likely segmentation by back-tracking:

$$s^*(t-\Delta t) = \Psi_{s^*(t)}(t). \tag{11}$$

These recursions yield a segmentation that is optimal at some finite temporal resolution Δt. Generally speaking, by choosing Δt to be sufficiently small, one can minimize the errors introduced by discretization. In practice, one would choose Δt to reflect the time scale beyond which it is not necessary to consider changes of state. For example, in pen-based handwriting recognition, Δt might be determined by the maximum pen velocity; in automatic speech recognition, by the sampling rate and frame rate.

Other types of inferences can be made from the distribution, eq. (8). For example, one can compute the marginal probability, $\Pr\left[s(\tau) = \text{END} \,|\, \boldsymbol{x}(t)\right]$ that the Markov process terminates at precisely the observed time. Similarly, one can compute the posterior probability, $\Pr[s(t_1) = i|\boldsymbol{x}(t), s(\tau) = \text{END}]$, that at an earlier time t_1, the variable s was in state i. These inferences are made by summing the probabilities in eq. (8) over all segmentations that terminate precisely at time τ. This sum is performed by discretizing the time axis and applying a forward-backward procedure similar to eqs. (10–11). These algorithms have the same form and computational complexity as their counterparts in hidden Markov models (Rabiner & Juang, 1993).

3 Learning from examples

The learning problem in MPCs is to estimate the parameters $\{\lambda_i, a_{ij}, g_i(\boldsymbol{x})\}$ in eq. (6) from examples of segmented (or non-segmented) curves. Our first step is to assume a convenient parameterization for the metrics, $g_i(\boldsymbol{x})$, that compute arc lengths. We then show how to fit these metrics, along with the parameters λ_i and a_{ij}, by maximum likelihood estimation.

3.1 Parameterizing the metric

A variety of parameterizations can be considered for the metrics, $g_i(\boldsymbol{x})$. The simplest possible form is a Euclidean metric, where $g_i(\boldsymbol{x})$ does not have any dependence on the point $\boldsymbol{x}$. Such a metric has the virtue of simplicity, but it is not very powerful in terms of what it can model. In this paper, we consider the more general form:

$$g_i(\boldsymbol{x}) = \Phi_i^2(\boldsymbol{x})\Sigma_i^{-1}, \tag{12}$$

where $\Phi_i(\boldsymbol{x})$ is a positive scalar-valued function of $\boldsymbol{x}$, and Σ_i is a positive-definite matrix with $|\Sigma_i| = 1$. Eq. (12) is a conformal transformation (Wald, 1984) of a Euclidean metric—that is, a non-Euclidean metric in which all the dependence on $\boldsymbol{x}$ is captured by a scalar prefactor. (A conformal transformation is one that locally preserves angles, but not distances.) Eq. (12) strikes one possible balance between the confines of Euclidean geometry and the full generality of Riemannian manifolds. The determinant constraint $|\Sigma_i| = 1$ is imposed to avoid the degenerate solution $g_i(\boldsymbol{x}) = 0$, in which every trajectory is assigned zero arc length. Note that we have defined the metric $g_i(\boldsymbol{x})$ in terms of the inverse of Σ_i; this turns out to simplify the parameter reestimation formula for Σ_i, given later in the section.

The form of the metric determines the nature of the learning problem in MPCs. For the choice in eq. (12), one must estimate the functions $\Phi_i(\boldsymbol{x})$, the matrices Σ_i, the decay parameters λ_i and the transition probabilities a_{ij}. In this section, we will consider the functions $\Phi_i(\boldsymbol{x})$ as fixed or pre-determined, leaving only the parameters Σ_i, λ_i, and a_{ij} to be estimated from training data. Later, in section 4.2, we will suggest a particular choice for the functions $\Phi_i(\boldsymbol{x})$ based on the relationship between MPCs and hidden Markov models.

3.2 Labeled examples

Suppose we are given examples of segmented trajectories, $\{\boldsymbol{x}_\alpha(t), s_\alpha(t)\}$, where the index α runs over the examples in the training set. As shorthand, let $I_{i\alpha}(t)$ denote the indicator function that selects out segments associated with state i:

$$I_{i\alpha}(t) \;=\; \begin{cases} 1 & \text{if } s_\alpha(t) = i, \\ 0 & \text{otherwise.} \end{cases} \tag{13}$$

Also, let $\ell_{i\alpha}$ denote the total arc length traversed by state i in the αth example:

$$\ell_{i\alpha} = \int dt\;\, I_{i\alpha}(t)\,\left[\dot{\boldsymbol{x}}_\alpha^T\, g_i(\boldsymbol{x}_\alpha)\,\dot{\boldsymbol{x}}_\alpha\right]^{\frac{1}{2}}. \tag{14}$$

In this paper we view learning as a problem in maximum likelihood estimation. Thus we seek the parameters that maximize the log-likelihood:

$$\sum_\alpha \ln \Pr[s_\alpha|\boldsymbol{x}_\alpha] \;=\; -\sum_{i\alpha} \lambda_i \ell_{i\alpha} + \sum_i n_i \ln \lambda_i + \sum_{ij} n_{ij} \ln a_{ij}, \tag{15}$$

where n_{ij} is the overall number of observed transitions from state i to state j, and $n_i = \sum_j n_{ij}$ is the number of visits to state i. Eq. (15) follows directly from the distribution over segmentations in eq. (8). Note that the first two terms measure the log-likelihood of observed segments in isolation, while the last term measures the log-likelihood of observed transitions.

Eq. (15) has a convenient form for maximum likelihood estimation. In particular, for fixed Σ_i, there are closed-form solutions for the optimal values of λ_i and a_{ij}; these are given by:

$$a_{ij} \;=\; n_{ij}/n_i, \tag{16}$$

$$\lambda_i^{-1} \;=\; \frac{1}{n_i}\sum_\alpha \ell_{i\alpha}. \tag{17}$$

These formulae are easy to interpret. The transition probabilities a_{ij} are determined by observed counts of transitions, while the decay parameters λ_i are determined by the mean arc lengths traversed in each state.

In general, we cannot find closed-form solutions for the maximum likelihood estimates of Σ_i. However, we can update these matrices in an iterative fashion that is guaranteed to increase the log-likelihood at each step. Denoting the updated matrices by $\tilde{\Sigma}_i$, we consider the iterative scheme (derived in the appendix):

$$\tilde{\Sigma}_i \;\leftarrow\; c_i \sum_\alpha \int dt\;\, I_{i\alpha}(t)\,\Phi_i(\boldsymbol{x}_\alpha(t))\,\frac{\dot{\boldsymbol{x}}_\alpha\,\dot{\boldsymbol{x}}_\alpha^T}{\left[\dot{\boldsymbol{x}}_\alpha^T \Sigma_i^{-1}\,\dot{\boldsymbol{x}}_\alpha\right]^{\frac{1}{2}}}, \tag{18}$$

where the constant c_i is determined by the determinant constraint $|\tilde{\Sigma}_i| = 1$. The reestimation formula for Σ_i involves a sum and integral over all segments assigned to the ith state of the MPC. In practice, the integral is evaluated numerically by discretizing the time axis. By taking gradients of eq. (15), one can show that the fixed points of this iterative procedure correspond to stationary points of the log-likelihood. A proof of monotonic convergence is given in the appendix.

In the case of labeled examples, the above procedures for maximum likelihood estimation can be invoked independently for each state i. One first iterates eq. (18) to estimate the matrix elements of Σ_i. These parameters are then used to compute the arc lengths, $\ell_{i\alpha}$, that appear in eq. (14). Given these arc lengths, the decay parameters and transition probabilities follow directly from eqs. (16–17). Thus the problem of learning given labeled examples is relatively straightforward.

3.3 Unlabeled examples

In an unsupervised setting, the learner does not have access to labeled examples; the only available information consists of the trajectories $\boldsymbol{x}_\alpha(t)$, as well as the fact that each process terminates at some time τ_α. The goal of unsupervised learning is to maximize the log-likelihood that for each trajectory $\boldsymbol{x}_\alpha(t)$, some probable segmentation can be found that terminates at precisely the observed time. The appropriate marginal probability is computed by summing $\Pr[s(t)|\boldsymbol{x}(t)]$ over allowed segmentations, as described at the end of section 2.3.

The maximization of this log-likelihood defines a problem in hidden variable density estimation. The hidden variables are the states of the Markov process. If these variables were known, the problem would reduce to the one considered in the previous section. To fill in these missing values, one can use the Expectation-Maximization (EM) algorithm (Baum, 1972; Dempster, Laird, & Rubin, 1976). Roughly speaking, the EM algorithm works by converting the maximization of the hidden variable problem into a weighted version of the problem where the segmentations, $s_\alpha(t)$, are known. The weights are determined by the posterior probabilities, $\Pr[s_\alpha(t)|\boldsymbol{x}_\alpha(t), s_\alpha(\tau_\alpha) = \text{END}]$, derived from the current parameter estimates.

We note that eqs. (10–11) suffice to implement an extremely useful approximation to the EM algorithm in MPCs. This approximation is to compute, based on the current parameter estimates, the optimal segmentation, $s^*_\alpha(t)$, for each trajectory in the training set; one then re-estimates the parameters of the Markov process by treating the inferred segmentations, $s^*_\alpha(t)$, as targets. This approximation reduces the problem of parameter estimation to the one considered in the previous section. It can be viewed as a winner-take-all approximation to the full EM algorithm, analogous to the Viterbi approximation for learning in hidden Markov models (Rabiner & Juang, 1993).

Essentially the same algorithm can also be applied to partially labeled examples. Partially labeled examples can take different forms. For example, the labels may specify the state sequences, but not the segment boundaries, such as:

$$\{(\text{START}, 0) \to (s_1, ?) \cdots\ (s_n, ?) \to (\text{END})\}. \tag{19}$$

Or they may specify the segment boundaries, but not the state sequences:

$$\{(\text{START}, 0) \to (?, t_1) \cdots\ (?, t_n) \to (\text{END})\}. \tag{20}$$

The ability to handle such examples is important for two reasons: first, because they provide more information than unlabeled examples, and second, because complete segmentations in the form of eq. (2) may not be available. For example, in the problem of automatic speech recognition, phonetic transcriptions in the form of eq. (19) are much easier to obtain than phonetic alignments. As before, we can view the learning problem for these examples as one in hidden variable density estimation. For these examples, partial knowledge of the state sequence and/or segment boundaries is incorporated into the EM algorithm simply by restricting the forward-backward procedures to allowed paths.

4 Observations

In this section, we present some extensions to MPCs and discuss how they relate to other probabilistic models for trajectory segmentation.

4.1 Extensions to MPCs

MPCs can accomodate more general measures of distance than the one presented in eq. (4). For example, let $\boldsymbol{u} = \dot{\boldsymbol{x}}/|\dot{\boldsymbol{x}}|$ denote the unit tangent vector along the curve of $\boldsymbol{x}$, where $|\dot{\boldsymbol{x}}| = (\dot{\boldsymbol{x}}^T\dot{\boldsymbol{x}})^{1/2}$. A simple extension of eq. (4) is to consider $d\ell^2 = \dot{\boldsymbol{x}}^T g(\boldsymbol{x},\boldsymbol{u})\,\dot{\boldsymbol{x}}$, where $g(\boldsymbol{x},\boldsymbol{u})$ depends not only on the point $\boldsymbol{x}$, but also on the tangent vector $\boldsymbol{u}$. This extension enables one to assign different distances to time-reversed trajectories, as opposed to the measure in eq. (5), which does not depend on whether the curve is traversed forwards or backwards. More generally, one may incorporate any of the vector invariants

$$\left[\frac{1}{|\dot{\boldsymbol{x}}|}\left(\frac{d}{dt}\right)\right]^n [\boldsymbol{x}(t)] \tag{21}$$

into the distance measure. These vectors characterize the local geometry at each point along the curve; in particular, eq. (21) gives the point $\boldsymbol{x}$ for $n=0$, the unit tangent vector $\boldsymbol{u}$ for $n=1$, the local curvature for $n=2$, etc. Incorporating higher-order derivatives in this way enables one to use fairly general distance measures in MPCs.

The invariance to nonlinear warpings of time can also be relaxed in MPCs. This is done by including time as a coordinate in its own right—i.e., by operating on the *spacetime* trajectories $\boldsymbol{z}(t) = \{\boldsymbol{x}(t), t\}$ and computing generalized arc lengths, $d\ell^2 = \dot{\boldsymbol{z}}^T G(\boldsymbol{x})\,\dot{\boldsymbol{z}}$, where $\dot{\boldsymbol{z}} = \{\dot{\boldsymbol{x}}, 1\}$ and $G(\boldsymbol{x})$ is a spacetime metric—a $(D+1)$-dimensional square matrix for each point $\boldsymbol{x}$. The effect of replacing $\dot{\boldsymbol{x}}$ by $\dot{\boldsymbol{z}}$ is to allow stationary portions of the trajectory to contribute to the integral $\ell = \int d\ell$. Mixing space and time coordinates in this way is an old idea from physics, originating in the theory of relativity, though in that context the metric is negative-definite (Wald, 1984). Note that this extension of MPCs can also be combined with the previous one—for instance, by incorporating both tangent vectors and timing information into the distance measure.

4.2 Relation to hidden Markov models and previous work

Hidden Markov models (HMMs), currently the most popular approach to trajectory segmentation, are also based on probabilistic methods. These models parameterize joint distributions of the form:

$$\Pr[s, \boldsymbol{x}] \;=\; \prod_t \Pr[s_t|s_{t-1}]\; \Pr[\boldsymbol{x}_t|s_t]. \tag{22}$$

There are several important differences between HMMs and MPCs (besides the trivial one that HMMs are formulated for discrete-time processes). First, the predictions of HMMs are not invariant to nonlinear warpings of time. For example, consider the pair of trajectories $\boldsymbol{x}_t$ and $\boldsymbol{y}_t$, where $\boldsymbol{y}_t$ is created by the doubling operation:

$$\boldsymbol{y}_t \;=\; \begin{cases} \boldsymbol{x}_{t/2} & \text{if } t \text{ even,} \\ \boldsymbol{y}_{t-1} & \text{if } t \text{ odd.} \end{cases} \tag{23}$$

Both trajectories trace out the same curve, but $\boldsymbol{y}_t$ does so at half the rate as $\boldsymbol{x}_t$. In general, HMMs will not assign these trajectories the same likelihood, nor are they guaranteed to infer equivalent segmentations. This is true even for HMMs with more sophisticated durational models (Rabiner & Juang, 1993). By contrast, these trajectories will be processed identically by MPCs based on eqs. (5-6).

The states in HMMs and MPCs are also weighted differently by their inference procedures. On one hand, in HMMs, the contribution of each state to the log-likelihood grows in proportion to its duration in time (i.e., to the number of observations attributed to that state). On the other hand, in MPCs, the contribution of each state grows in proportion to its arc length. Naturally, the weighting by arc length attaches a more important role to short-lived states with non-stationary trajectories. The consequences of this for automatic speech recognition are discussed in section 5.

HMMs and MPCs also differ in what they model. HMMs parameterize joint distributions of the form given by eq. (22). Thus, in HMMs, maximum likelihood parameter estimation is directed at learning a *synthesis* model, $\Pr[\boldsymbol{x}|s]$, while in MPCs, it is directed at learning a *segmentation* model, $\Pr[s|\boldsymbol{x}]$. The direction of conditioning on $\boldsymbol{x}$ is a crucial difference. In HMMs, one can generate artificial trajectories by sampling from the joint distribution $\Pr[s, \boldsymbol{x}]$; MPCs, on the other hand, do not provide a generative model of trajectories. The Markov assumption is also slightly different in HMMs and MPCs. HMMs observe the conditional independence $\Pr[s_{t+1}|s_t, \boldsymbol{x}_t] = \Pr[s_{t+1}|s_t]$, such that the state, s_{t+1}, is independent of the observation, x_t, given the previous state, s_t. By contrast, in MPCs the evolution of $p_i(t)$, as given by eq. (6), depends explicitly on the trajectory at time t—namely, through the arc length $[\dot{\boldsymbol{x}}^T g_i(\boldsymbol{x})\dot{\boldsymbol{x}}]^{1/2}$.

While MPCs do not provide a generative model of trajectories, we emphasize that they do provide a generative model of segmentations. In particular, one can generate a state sequence $s_0 s_1 s_2 \ldots s_{n+1}$, where s_0 is the START state and s_{n+1} is the END state, by sampling from the transition probabilities a_{ij}. (Here, the sequence length n is not fixed in advance, but determined by the sampling procedure.) Moreover, for each state s_k, one can generate an arc length ℓ_k by sampling from the exponential distribution, $P(\ell_k) = \lambda_{s_k} e^{-\lambda_{s_k}\ell_k}$. Together, these sampled values of s_k and ℓ_k define a segmentation that can be grafted onto any (sufficiently long) trajectory $\boldsymbol{x}(t)$. Importantly, this interpretation of MPCs allows them to be combined hierarchically with other generative models, such as language models in automatic speech recognition. For example, denoting words by $\mathcal{W}$, states by $\mathcal{S}$, and arc lengths by $\mathcal{L}$, one can write: $\Pr[\mathcal{W}, \mathcal{S}, \mathcal{L}] = \Pr[\mathcal{W}]\,\Pr[\mathcal{S}|\mathcal{W}]\,\Pr[\mathcal{L}|\mathcal{S}]$.

Finally, we note that one can essentially realize HMMs as a special case of MPCs. This is done by computing arc lengths along spacetime trajectories $\boldsymbol{z}(t) = \{\boldsymbol{x}(t), t\}$, as described in section 4.1. In this setting, one can mimic the predictions of HMMs by setting the Σ_i matrices to have only one non-zero element (namely, the diagonal element for delta-time contributions to the arc length) and by defining the functions $\Phi_i(\boldsymbol{x})$ in terms of the HMM emission probabilities $\Pr(\boldsymbol{x}|i)$ as:

$$\Phi_i(\boldsymbol{x}) = -\ln\left[\frac{\Pr(\boldsymbol{x}|i)}{\sum_k \Pr(\boldsymbol{x}|k)}\right]. \tag{24}$$

This equation sets up a correspondence between the emission log-probabilities in HMMs and the arc lengths in MPCs. Ignoring the effects of transition probabilities (which are often negligible), an MPC initialized by eq. (24) and this singular choice of Σ_i will reproduce the segmentations of its "parent" HMM. This correspondence is important because it allows one to bootstrap an MPC from a previously trained HMM. (Also, despite many efforts, we have not found a more effective way to estimate the functions $\Phi_i(\boldsymbol{x})$.)

In terms of previous work, our motivation for MPCs resembles that of Tishby (1990), who several years ago proposed a dynamical systems approach to speech processing. Because MPCs exploit the notion that trajectories are continuous, they also bear some resemblance to so-called *segmental* HMMs (Ostendorf, Digalakis, & Kimball, 1996). MPCs nevertheless differ from segmental HMMs in two important respects: (i) the treatment of arc length—particularly, the estimation of a metric $g_i(\boldsymbol{x})$ for each hidden state of the Markov process, and (ii) the natural parameterization of a segmentation model $\Pr[s|\boldsymbol{x}]$, as opposed to a synthesis model, $\Pr[\boldsymbol{x}|s]$, that is even more complicated than the one in ordinary HMMs.

5 Automatic speech recognition

The Markov processes in this paper were conceived as models for automatic speech recognition (Rabiner & Juang, 1993). Speech recognizers take as input a sequence of feature vectors, each of which summarizes the acoustic properties of a short window of speech. Acoustic feature vectors typically have ten or more components, so that a particular sequence of feature vectors can be viewed as tracing out a multidimensional curve. The goal of a speech recognizer is to translate this curve into a sequence of words, or more generally, a sequence of sub-syllabic units known as *phonemes*. Denoting the feature vectors by $\boldsymbol{x}_t$ and the phonemes by s_t, we can view this problem as the discrete-time equivalent of the segmentation problem in MPCs.

5.1 Invariances of speech

Though HMMs have led to significant advances in automatic speech recognition, they are handicapped by certain weaknesses. One of these is the poor manner in which they model variations in speaking rate (Siegler & Stern, 1995). Typically, HMMs make more errors on fast speech than slow speech. A related effect, occurring at the phoneme level, is that consonants are confused more often than vowels. Generally speaking, consonants have short-lived, non-stationary acoustic signatures; vowels, just the opposite. Thus, at the phoneme level, we can view consonantal confusions as a consequence of locally fast speech.

It is tempting to imagine that HMMs make these mistakes because they do not incorporate an invariance to nonlinear warpings of time. While this oversimplifies the problem, it is clear that HMMs have systemic biases. In HMMs, the contribution of each state to the log-likelihood grows in proportion to its duration in time. Thus decoding procedures in HMMs are inherently biased to pay more attention to long-lived states than short-lived ones. In our view, this suggests one plausible explanation for the tendency of HMMs to confuse consonants more often than vowels.

MPCs are quite different from HMMs in how they weight the speech signal. In MPCs, the contribution of each state is determined by its arc length. The weighting by arc length attaches a more important role to short-lived but non-stationary phonemes, such as consonants. Of course, one can imagine heuristics in HMMs that achieve the same effect, such as dividing each state's contribution to the log-likelihood by its observed (or inferred) duration. Unlike such heuristics, however, the metrics $g_i(\boldsymbol{x})$ in MPCs are estimated from each state's training data; in other words, they are designed to reweight the speech signal in a way that reflects the statistics of acoustic trajectories.

Admittedly, it is oversimplistic to model the effects of speaking rate by an invariance to nonlinear warpings of time. The acoustic realization (i.e., spectral profile) of any phoneme does depend to some extent on the speaking rate, and certain phonemes are more likely to be stretched or shortened than others. Also, articulatory trajectories do not remain invariant to changes in the rate of speech; one observes both overshoot and undershoot of articulatory targets. Finally, an invariance to nonlinear warpings of time presupposes a certain separation of time scales: on one hand, there is the time scale at which input features are extracted from the speech signal (i.e., the frame rate); on the other, there is the time scale at which these features tend to vary. These time scales need to be well separated for MPCs to have a meaningful interpretation. Whether this is true obviously depends on the choice of input features.

Despite these caveats, we feel that MPCs provide a compelling alternative to traditional methods. While we have motivated MPCs by appealing to the intrinsic geometric properties of curves, we emphasize that for automatic speech recognition, it is important to relax the invariance to nonlinear warpings of time. This is done by computing arc lengths along spacetime trajectories, as described in section 4.1. This extension allows MPCs to incorporate both movement in acoustic feature space *and* duration in time as measures of phonemic evolution. Both of these measures are important for speech recognition.

5.2 Experiments

Both HMMs and MPCs were used to build connected speech recognizers. Training and test data came from speaker-independent databases of telephone speech. All data was digitized at the caller's local switch and transmitted in this form to the receiver. For feature extraction, input telephone signals (sampled at 8 kHz and band-limited between 100–3800 Hz) were pre-emphasized and blocked into 30ms frames with a frame shift of 10ms. Each frame was

Hamming windowed, autocorrelated, and processed by a linear predictive coding (LPC) cepstral analysis to produce a vector of 12 liftered cepstral coefficients (Rabiner & Juang, 1993). The feature vector was then augmented by its normalized log energy value, as well as temporal derivatives of first and second order. Overall, each frame of speech was described by 39 features. These features were used differently by HMMs and MPCs, as described below.

Recognizers were evaluated on two tasks. The first task was recognizing New Jersey town names (e.g., Hoboken). The training data for this task (Sachs et al, 1994) consisted of 12100 short phrases, spoken in the seven major dialects of American English. These phrases, ranging from two to four words in length, were selected to provide maximum phonetic coverage. The test data consisted of 2426 isolated utterances of 1219 New Jersey town names and was collected from nearly 100 speakers. Note that the training and test data for this task have non-overlapping vocabularies.

Baseline recognizers were built using 43 left-to-right continuous-density HMMs, each corresponding to a context-independent English phone. Phones were modeled by three-state HMMs, with the exception of background noise, which was modeled by a single state. State emission probabilities were computed by Gaussian mixture models with diagonal covariance matrices. Different sized models were trained using $M = 2$, 4, 8, 16, 32, and 64 mixture components per hidden state; for a particular model, the number of mixture components was the same across all states. Mixture model parameters were estimated by a Viterbi implementation of the Baum-Welch algorithm. Transition probabilities were assigned default values; in particular, all transitions allowed by the task grammar were assumed to be equally probable. (This assumption simplifies the forward-backward procedure in large state spaces.)

MPC recognizers were built using the same overall grammar. The HMM segmentations were used to provide labeled examples for the supervised learning procedures in section 3.2. Each hidden state in the MPCs was assigned a metric $g_i(\boldsymbol{x}) = \Sigma_i^{-1}\Phi_i^2(\boldsymbol{x})$. The functions $\Phi_i(\boldsymbol{x})$ were initialized (and fixed) by the state emission probabilities of the HMMs, as in eq. (24), using the 39 dimensional cepstral feature vectors. The matrices Σ_i were estimated by iterating eq. (18) until convergence. Convergence typically occurred within six iterations, over the course of which the matrices Σ_i changed significantly from their initial values. We computed arc lengths along the 14 dimensional spacetime trajectories through cepstra, log-energy, and time. Thus each Σ_i was a 14×14 symmetric matrix applied to tangent vectors consisting of delta-cepstra, delta-log-energy, and delta-time. Note that these MPCs made use of both extensions discussed in section 4.1. Curiously, our best results for MPCs were obtained by setting $\lambda_i = 1$, as opposed to estimating the values of these decay parameters from training data. We suspect this was due to the highly irregular (i.e., non-exponential) distribution of arc lengths in the state representing silence and background noise. As in the HMMs, transition probabilities were assigned default values.

Table 1 shows the results of these experiments comparing MPCs to HMMs. The error rates in these experiments measure the percentage of town names in the test set that were incorrectly recognized. The horizontal axis shows the number of mixture components per hidden state in the HMMs; the emission probabilities from these HMMs were also used to compute the MPC metrics from eqs. (12) and (24). Both recognizers used the same recursive search algorithm to determine the best path through the allowable lattice of states. Beam

mixture components	HMM error rate (%)	MPC error rate (%)
2	22.3	20.9
4	18.9	17.5
8	16.5	15.1
16	14.6	13.3
32	13.5	12.3
64	11.7	11.4

Table 1: Test set error rates on the task of recognizing New Jersey town names versus the number of mixture components per hidden state.

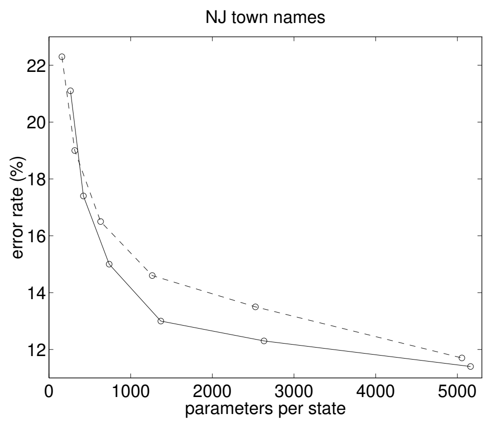

Figure 2: Test set error rates for HMMs (dashed) and MPCs (solid) on New Jersey town names versus the number of parameters per hidden state.

widths were chosen so that corresponding recognizers activated roughly equal numbers of arcs. For various model sizes (as measured by the number of mixture components), we found the MPCs to yield consistently lower error rates than the HMMs. The graph in figure 2 plots these error rates versus the number of modeling parameters per hidden state. This graph shows that the MPCs are not outperforming the HMMs merely because they have extra modeling parameters (i.e., the Σ_i matrices).

The second task in our experiments involved the recognition of connected alpha-digits (e.g., N Z 3 V J 4 E 3 U 2). The training and test data consisted of 14622 and 7255 utterances, respectively. Recognizers were built from 285 subword HMMs/MPCs, each corresponding to a context-dependent English phone. Each subword model had three states, with the exception of the model for background noise, which only had a single state. The recognizers were trained and evaluated in the same way as the previous task, except that we measured word error rates instead of phrase error rates. The results, shown in table 2 and figure 3, follow a similar pattern as before, with the MPCs outperforming the HMMs.

mixture components	HMM error rate (%)	MPC error rate (%)
2	12.5	10.0
4	10.7	8.8
8	10.0	8.2

Table 2: Test set word error rates on the task of recognizing connected alpha-digits versus the number of mixture components per hidden state.

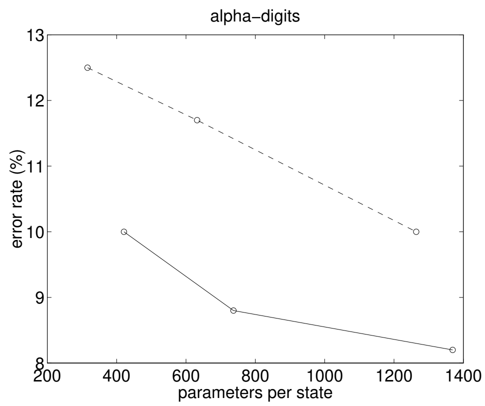

Figure 3: Test set word error rates for HMMs (dashed) and MPCs (solid) on connected alpha-digits versus the number of parameters per hidden state.

6 Discussion

The experimental results in the previous section demonstrate the viability of MPCs for automatic speech recognition. Nevertheless, several issues require further attention. One important issue is the problem of feature selection—namely, how to extract meaningful trajectories from the speech signal. In this work, we used the same cepstral features for both MPCs and HMMs; this was done to facilitate a side-by-side comparison. It is doubtful, however, that cepstral trajectories (which are not particularly smooth) provide the most meaningful type of input to MPCs. Intuitively, one suspects that other measurements (e.g., pitch contours, formant trajectories, articulatory features) would provide smoother, more informative trajectories than cepstra; unfortunately, these types of features are not as straightforward to compute. Further work in this area is needed.

Another important issue for MPCs is learning—namely, how to parameterize and estimate the metrics $g_i(\boldsymbol{x})$ from trajectories $\boldsymbol{x}(t)$. We stress that the learning problem in MPCs has many more degrees of freedom than the corresponding one in HMMs. In particular, whereas in HMMs one must learn a distribution $\Pr(\boldsymbol{x}|i)$ for each hidden state, in MPCs one must learn a metric $g_i(\boldsymbol{x})$. The former is a *scalar*-valued function over the acoustic feature space; the latter, a *matrix*-valued function. It is fair to say that we do not understand how to parameterize metrics nearly as well as probability distributions. Certainly, MPCs have the potential to exploit more sophisticated metrics than the one studied in this paper. Moreover, it is somewhat unsatisfactory that the metric in eq. (12) relies on a trained HMM for its initialization. Finally, we note that the form of eq. (12) has an unfortunate drawback: the denominator requires a sum over all the states in the MPC. This did not present a major computational penalty for the applications we considered, but it would be prohibitively expensive for problems with larger state spaces. Thus, better solutions for the metric are needed to scale MPCs up to larger problems in ASR.

On a final note, we emphasize that the issues of feature selection and parameter estimation in MPCs are not independent. The cepstral front end in today's speech recognizers is extremely well matched to the HMM back end; indeed, one might argue that over the last decade of research, each has been systematically honed to compensate for the other's failings. It seems likely that future progress in automatic speech recognition will require concerted efforts at both ends. Thus we hope that besides providing an alternative to HMMs, MPCs also encourage a fresh look at the signal processing performed by the front end.

A Reestimation formula

In this appendix we derive the reestimation formula, eq. (18) and show that it leads to monotonic increases in the log-likelihood, eq. (15). Recall that in MPCs, the probability of remaining in a state decays exponentially as a function of the arc length. It follows that maximizing the log-likelihood in each state is equivalent to minimizing its arc length. For the choice of metric in eqs. (12) and (24), the learning problem reduces to optimizing the matrices Σ_i. For simplicity, consider the arc length of a single trajectory under this metric:

$$\ell(\Sigma) = \int_0^\tau dt \ \left[\dot{\boldsymbol{x}}^T \Sigma^{-1} \dot{\boldsymbol{x}}\right]^{\frac{1}{2}} \Phi(\boldsymbol{x}(t)). \tag{25}$$

Here we have written the arc length $\ell(\Sigma)$ explicitly as a function of the matrix Σ, and we have suppressed the state index for notational convenience.

Our goal is to minimize $\ell(\Sigma)$, subject to the determinant constraint $|\Sigma| = 1$. Note that the matrix elements of Σ^{-1} appear nonlinearly in the right hand side of eq. (25); thus it is not possible to compute their optimal values in closed form. As an alternative, we consider the auxiliary function:

$$Q(\Psi, \Sigma) = \int_0^\tau dt \ \left\{\frac{\dot{\boldsymbol{x}}^T \Psi^{-1} \dot{\boldsymbol{x}}}{\left[\dot{\boldsymbol{x}}^T \Sigma^{-1} \dot{\boldsymbol{x}}\right]^{\frac{1}{2}}} + \left[\dot{\boldsymbol{x}}^T \Sigma^{-1} \dot{\boldsymbol{x}}\right]^{\frac{1}{2}}\right\} \frac{\Phi\left(\boldsymbol{x}(t)\right)}{2}, \tag{26}$$

where Ψ is a $D \times D$ positive-definite matrix like Σ. It follows directly from the definition in eq. (26) that $\ell(\Sigma) = Q(\Sigma, \Sigma)$. Somewhat less trivially, we observe that $Q(\Psi, \Psi) \leq Q(\Psi, \Sigma)$ for all positive definite matrices Ψ and Σ. This inequality follows from the concavity of the square root function, as illustrated in figure 4.

Consider the value of Ψ which minimizes $Q(\Psi, \Sigma)$, subject to the determinant constraint $|\Psi| = 1$. We denote this value by $\tilde{\Sigma} = \min_{|\Psi|=1} Q(\Psi, \Sigma)$. Because the matrix elements of Ψ^{-1} appear *linearly* in $Q(\Psi, \Sigma)$, this minimization can be performed analytically, using Lagrange multipliers to enforce the determinant constraint. The calculation has essentially the same form as the maximum likelihood estimate of the covariance matrix for a multivariate normal density (Anderson, 1958). In particular, the arc length is minimized by computing the

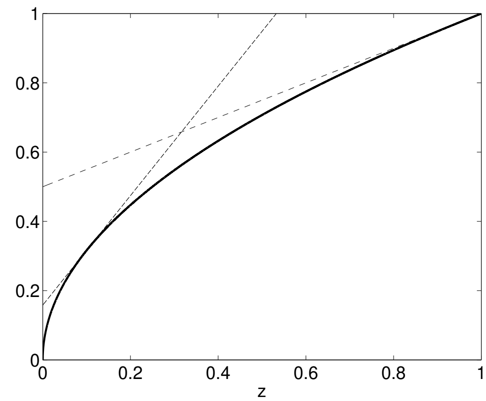

Figure 4: The square root function is concave and upper bounded by $\sqrt{z} \leq \frac{1}{2}[z\xi^{-1/2} + \xi^{1/2}]$ for all $\xi \geq 0$. The bounding tangents are shown for $\xi = \frac{1}{10}$ and $\xi = 1$.

correlation matrix of the tangent vector $\dot{\boldsymbol{x}}$, as distributed along the trajectory $\boldsymbol{x}(t)$, or:

$$\tilde{\Sigma} \;\propto\; \int_0^\tau dt\, \frac{\dot{\boldsymbol{x}}\dot{\boldsymbol{x}}^T}{[\dot{\boldsymbol{x}}^T\Sigma^{-1}\dot{\boldsymbol{x}}]^{\frac{1}{2}}}\,\Phi(\boldsymbol{x}(t)), \tag{27}$$

where the constant of proportionality is determined by the constraint $|\tilde{\Sigma}| = 1$. To minimize $\ell(\Sigma)$ with respect to Σ, we now consider the iterative procedure where at each step we replace Σ by $\tilde{\Sigma}$. We observe that:

$$\begin{aligned}
\ell(\tilde{\Sigma}) &= Q(\tilde{\Sigma}, \tilde{\Sigma}) \\
&\leq Q(\tilde{\Sigma}, \Sigma) \text{ due to concavity} \\
&\leq Q(\Sigma, \Sigma) \text{ since } \tilde{\Sigma} = \min_\Psi Q(\Psi, \Sigma) \\
&= \ell(\Sigma),
\end{aligned}$$

with equality generally holding only when $\tilde{\Sigma} = \Sigma$. In other words, this iterative procedure converges monotonically to a local minimum of the arc length, $\ell(\Sigma)$. Extending this procedure to combined arc lengths over multiple trajectories, we obtain eq. (18).

Acknowledgements

The authors thank F. Pereira and the anonymous reviewers for many helpful comments about the presentation of these ideas.

References

[1] Anderson, T. W. (1958). *An Introduction to Multivariate Statistical Analysis*. New York, NY: John Wiley.

[2] Baum, L. (1972). An inequality and associated maximization technique in statistical estimation for probabilistic functions of a markov process. In O. Shisha, editor, *Inequalities*, 3:1–8. New York, NY: Academic Press.

[3] Dempster, A., Laird, N., & Rubin, D.(1977). Maximum likelihood from incomplete data via the em algorithm. *Journal of the Royal Statistical Society B*, 39:1–38.

[4] DoCarmo, M. P. (1976). *Differential Geometry of Curves and Surfaces.* Englewood Cliffs, NJ: Prentice Hall.

[5] Duda, R. O., & Hart, P. E. (1973). *Pattern Classification and Scene Analysis.* New York, NY: Wiley.

[6] Ostendorf, M., Digalakis, V., & Kimball, O. (1996). From HMMs to segment models: a unified view of stochastic modeling for speech recognition. *IEEE Transactions on Acoustics, Speech and Signal Processing*, 4:360–378.

[7] Papoulis, A. (1991). *Probability, Random Variables, and Stochastic Processes.* Boston, MA: McGraw-Hill.

[8] Rabiner, L. R., & Juang, B. H. (1993). *Fundamentals of Speech Recognition.* Englewood Cliffs, NJ: Prentice Hall.

[9] Sachs, R., Tikijian, M., & Roskos, E. (1994). United States English subword speech data. *AT&T unpublished report.*

[10] Siegler, M. A., & Stern, R. M. (1995). On the effects of speech rate in large vocabulary speech recognition systems. In *Proceedings of the 1995 IEEE International Conference on Acoustics, Speech, and Signal Processing*, 612–615.

[11] Simard, P. Y., LeCun, Y., & Denker, J. (1993). Efficient pattern recognition using a new transformation distance. In *Advances in Neural Information Processing Systems* 5:50–58. San Mateo, CA: Morgan Kauffman.

[12] Tishby, N. (1990). A dynamical system approach to speech processing. In *Proceedings of the 1990 IEEE International Conference on Acoustics, Speech, and Signal Processing*, 365–368.

[13] Viterbi, A. J. (1967). Error bounds for convolutional codes and an asyptotically optimal decoding algorithm. *IEEE Transactions on Information Theory*, 13:260–269.

[14] Wald, R. M. (1984). *General Relativity.* Chicago, IL: University of Chicago Press.