A Latent Manifold Markovian Dynamics Gaussian Process

Sotirios P. Chatzis and Dimitrios Kosmopoulos

***Abstract*—In this paper, we propose a Gaussian process model for analysis of nonlinear time series. Formulation of our model is based on the consideration that the observed data are functions of latent variables, with the associated mapping between observations and latent representations modeled through Gaussian process priors. In addition, to capture the temporal dynamics in the modeled data, we assume that subsequent latent representations depend on each other on the basis of a hidden Markov prior imposed over them. Derivation of our model is performed by marginalizing out the model parameters in closed form by using Gaussian process priors for observation mappings, and appropriate stick-breaking priors for the latent variable (Markovian) dynamics. This way, we eventually obtain a nonparametric Bayesian model for dynamical systems that accounts for uncertainty in the modeled data. We provide efficient inference algorithms for our model on the basis of a truncated variational Bayesian approximation. We demonstrate the efficacy of our approach considering a number of applications dealing with real-world data, and compare it to related state-of-the-art approaches.**

***Index Terms*—Gaussian process, stick-breaking process, Markovian dynamics, latent manifold, variational Bayes.**

I. INTRODUCTION

There is a wide variety of generative models used to perform analysis of nonlinear time series [1]. Approaches based on hidden Markov models (HMMs) and linear dynamical systems (LDS) are quite ubiquitous in the current literature due to their simplicity, efficiency, and generally satisfactory performance in many applications. More expressive models, such as switching linear dynamical systems (SLDS) and nonlinear dynamical systems (NLDS), have also been proposed; however, these approaches are faced with difficulties in terms of their learning and inference algorithms, due to the entailed large number of parameters that must be estimated, and the hence needed large amounts of training data [1].

Recently, a nonparametric Bayesian approach designed to resolve these issues of NLDS, namely the Gaussian process dynamical model (GPDM), was introduced in [2]. This approach is fully defined by a set of low-dimensional representations of the observed data, with both the observation and dynamical processes learned by means of Gaussian process (GP) regression [3]. This GP-based formulation of the model gives rise to a nonparametric Bayesian nature, which removes the need to select a large number of parameters associated with function approximators, while retaining the power of nonlinear dynamics and observation. GPDM is essentially an extension of the GP latent variable model (GPLVM) of [4], which models the joint distribution of the observed data and their representation in a low-dimensional latent space through a GP prior. GPDM extends GPLVM by augmenting it with a model of temporal dynamics captured through imposition of a dedicated GP prior. This way, GPDM allows for not only obtaining predictions about future data, but also regularizing the latent space to allow for more effective modeling of temporal dynamics.

Despite the merits of GPDM, a significant drawback of this model consists in the need of its inference algorithm to obtain maximum a posteriori (MAP) estimates of its parameters through type-II maximum-likelihood [2] (performed by means of scaled conjugate gradient descent, SCG [5]). This formulation poses a significant bottleneck to GPDM, due to both the entailed high computational costs, as well as the possibility of obtaining bad estimates due to the algorithm getting stuck to poor local maxima in cases of limited training datasets. In addition, to increase computational efficiency, GPDM imposes an oversimplistic spherical Gaussian prior over its model of temporal dynamics, which probably undermines its data modeling capacity. Finally, the use of GP priors to describe the temporal dynamics between the latent variables of the model leads to significant computational overheads, as it gives rise to calculations that entail inverting very large *gram* matrices [3].

To resolve these issues, in this paper we propose a flexible generative model for modeling sequential data by means of nonparametric component densities. Formulation of our proposed model is based on the assumption that, when modeling sequential data, each observation in a given sequence is related to a vector in a latent space, and is generated through a latent nonlinear function that maps the latent space to the space of observations. We use a GP prior to infer this unknown mapping function from the data in a flexible manner.

In addition, the latent vectors that generate the observed sequential data are assumed to possess strong temporal interdependencies; to capture these dependencies, we assume that these latent variables are generated from a hidden Markov model in the manifold of latent variables. Specifically, we assume a latent space HMM with infinite hidden states, and use flexible stick-breaking priors [6], [7] to infer its hidden state dynamics; this formulation allows us to automatically determine the required number of states in this latent manifold HMM in a data-driven fashion. We dub our approach the latent manifold hidden Markov Gaussian process (LM^{2}GP) model. Inference for our model is performed by means of an efficient truncated variational Bayesian algorithm [8], [9]. We evaluate the efficacy of our approach in several applications, dealing with sequential data clustering, classification, and generation. A high-level illustration of the conceptual configuration of our

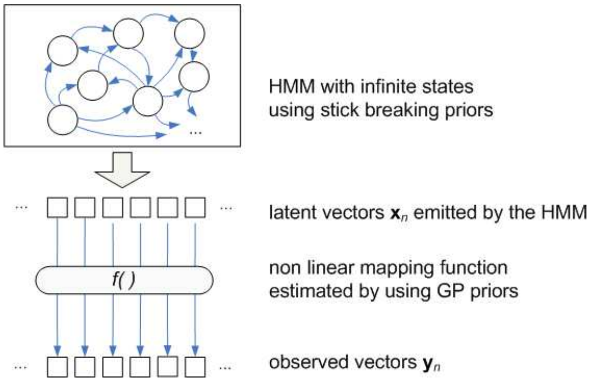
HMM with infinite states
using stick breaking priors
latent vectors $\mathbf{x}_n$ emitted by the HMM
f( )
non linear mapping function
estimated by using GP priors
observed vectors $\mathbf{y}_n$

Figure 1. Intuitive illustration of the generative construction of our model.

model is provided in Fig. 1.

The remainder of this paper is organized as follows: In Section II, we provide a brief presentation of the theoretical background of the proposed method. Initially, we present the Dirichlet process and its function as a prior in nonparametric Bayesian models; further, we provide a brief summary of the GPDM model. In Section III, we introduce the proposed LM^{2}GP model, and derive efficient model inference algorithms based on the variational Bayesian framework. In Section IV, we conduct the experimental evaluation of our proposed model, considering a number of applications dealing with several real-world datasets, and we compare its performance to state-of-the-art related approaches. Finally, in the last section of this paper, we conclude and summarize our work.

II. PRELIMINARIES

A. The Dirichlet process

Dirichlet process models were first introduced by Ferguson [7]. A DP is characterized by a base distribution G_0 and a positive scalar α, usually referred to as the innovation parameter, and is denoted as $\mathrm{DP}(\alpha, G_0)$. Essentially, a DP is a distribution placed over a distribution. Let us suppose we randomly draw a sample distribution G from a DP, and, subsequently, we independently draw M random variables $\{\Theta^*_m\}_{m=1}^M$ from G:

$$G|\alpha, G_0 \sim \mathrm{DP}(\alpha, G_0) \tag{1}$$

$$\Theta^*_m|G \sim G, \quad m = 1, \dots M \tag{2}$$

Integrating out G, the joint distribution of the variables $\{\Theta^*_m\}_{m=1}^M$ can be shown to exhibit a clustering effect. Specifically, given the first $M-1$ samples of G, $\{\Theta^*_m\}_{m=1}^{M-1}$, it can be shown that a new sample Θ^*_M is either (a) drawn from the base distribution G_0 with probability $\frac{\alpha}{\alpha+M-1}$, or (b) is selected from the existing draws, according to a multinomial allocation, with probabilities proportional to the number of the previous draws with the same allocation [10]. Let $\{\Theta_c\}_{c=1}^C$ be the set of distinct values taken by the variables $\{\Theta^*_m\}_{m=1}^{M-1}$. Denoting as ν_c^{M-1} the number of values in $\{\Theta^*_m\}_{m=1}^{M-1}$ that equal to Θ_c, the distribution of Θ^*_M given $\{\Theta^*_m\}_{m=1}^{M-1}$ can be shown to be of the form [10]

$$\begin{aligned} p(\Theta^*_M|\{\Theta^*_m\}_{m=1}^{M-1}, \alpha, G_0) = &\frac{\alpha}{\alpha+M-1}G_0 \\ &+ \sum_{c=1}^{C} \frac{\nu_c^{M-1}}{\alpha+M-1}\delta_{\Theta_c} \end{aligned} \tag{3}$$

where δ_{Θ_c} denotes the distribution concentrated at a single point Θ_c. These results illustrate two key properties of the DP scheme. First, the innovation parameter α plays a key-role in determining the number of distinct parameter values. A larger α induces a higher tendency of drawing new parameters from the base distribution G_0; indeed, as $\alpha \rightarrow \infty$ we get $G \rightarrow G_0$. Conversely, as $\alpha \rightarrow 0$ all $\{\Theta^*_m\}_{m=1}^M$ tend to cluster to a single random variable. Second, the more often a parameter is shared, the more likely it will be shared in the future.

A characterization of the (unconditional) distribution of the random variable G drawn from a DP, $\mathrm{DP}(\alpha, G_0)$, is provided by the stick-breaking construction of Sethuraman [6]. Consider two infinite collections of independent random variables $\boldsymbol{v} = [v_c]_{c=1}^\infty$, $\{\Theta_c\}_{c=1}^\infty$, where the v_c are drawn from a Beta distribution, and the Θ_c are independently drawn from the base distribution G_0. The stick-breaking representation of G is then given by

$$G = \sum_{c=1}^{\infty} \varpi_c(\boldsymbol{v})\delta_{\Theta_c} \tag{4}$$

where

$$p(v_c) = \mathrm{Beta}(1, \alpha) \tag{5}$$

$$\varpi_c(\boldsymbol{v}) = v_c \prod_{j=1}^{c-1}(1 - v_j) \quad \in [0, 1] \tag{6}$$

and

$$\sum_{c=1}^{\infty} \varpi_c(\boldsymbol{v}) = 1 \tag{7}$$

Under the stick-breaking representation of the DP, the atoms Θ_c, drawn independently from the base distribution G_0, can be seen as the parameters of the component distributions of a mixture model comprising an unbounded number of component densities, with mixing proportions $\varpi_c(\boldsymbol{v})$.

B. The Gaussian process dynamical model

1) Gaussian process models: Let us begin with a brief description of GP regression. Consider an observation space $\mathcal{X}$; a Gaussian process $f(\boldsymbol{x})$, $\boldsymbol{x} \in \mathcal{X}$, is defined as *a collection of random variables, any finite number of which have a joint Gaussian distribution* [3]. We typically use the notation

$$f(\boldsymbol{x}) \sim \mathrm{GP}(m(\boldsymbol{x}), k(\boldsymbol{x}, \boldsymbol{x})) \tag{8}$$

where $m(\boldsymbol{x})$ is the mean function of the process, and $k(\boldsymbol{x}, \boldsymbol{x}')$ is the covariance function of the process. Usually, for simplicity, and without any loss of generality, the mean of the process is taken to be zero, $m(\boldsymbol{x}) = 0$, although this is not necessary. Concerning selection of the covariance function, a large variety of *kernel* functions $k(\boldsymbol{x}, \boldsymbol{x}')$ might be employed, depending on the application considered [3]. Eventually, the real process $f(\boldsymbol{x})$ drawn from a GP with mean zero and kernel function $k(\boldsymbol{x}, \boldsymbol{x}')$ follows a Gaussian distribution with

$$p(f|\boldsymbol{x}) = \mathcal{N}(f|0, k(\boldsymbol{x}, \boldsymbol{x})). \tag{9}$$

Let us suppose a set of independent and identically distributed (i.i.d.) samples $\mathcal{D} = \{(\boldsymbol{x}_i, y_i)|i = 1, ..., N\}$, with the d-dimensional variables $\boldsymbol{x}_i$ being the observations related to a modeled phenomenon, and the scalars y_i being the associated target values. The goal of a regression model is, given a new observation $\boldsymbol{x}_*$, to predict the corresponding target value y_*, based on the information contained in the training set $\mathcal{D}$. The basic notion behind Gaussian process regression consists in the assumption that the observable (training) target values y in a considered regression problem can be expressed as the superposition of a Gaussian process over the input space $\mathcal{X}$, $f(\boldsymbol{x})$, and an independent white Gaussian noise

$$y = f(\boldsymbol{x}) + \epsilon \tag{10}$$

where $f(\boldsymbol{x})$ is given by (8), and

$$p(\epsilon) = \mathcal{N}(\epsilon|0, \sigma^2) \tag{11}$$

Under this regard, the joint normality of the training target values $Y = [y_i]_{i=1}^N$ and some unknown target value y_*, approximated by the value f_* of the postulated Gaussian process evaluated at the observation point $\boldsymbol{x}_*$, is a Gaussian of the form [3]

$$\mathcal{N}\left(\begin{bmatrix} Y \\ f_* \end{bmatrix} |\mathbf{0}, \begin{bmatrix} \boldsymbol{K}(X,X) + \sigma^2 \boldsymbol{I}_N & \boldsymbol{k}(\boldsymbol{x}_*) \\ \boldsymbol{k}(\boldsymbol{x}_*)^T & k(\boldsymbol{x}_*, \boldsymbol{x}_*) \end{bmatrix} \right) \tag{12}$$

where

$$\boldsymbol{k}(\boldsymbol{x}_*) \triangleq [k(\boldsymbol{x}_1, \boldsymbol{x}_*), \dots, k(\boldsymbol{x}_N, \boldsymbol{x}_*)]^T \tag{13}$$

$X = [\boldsymbol{x}_i]_{i=1}^N$, $\boldsymbol{I}_N$ is the $N \times N$ identity matrix, and $\boldsymbol{K}$ is the matrix of the covariances between the N training data points *(gram matrix),* i.e.

$$\boldsymbol{K}(X,X) \triangleq \begin{bmatrix} k(\boldsymbol{x}_1, \boldsymbol{x}_1) & k(\boldsymbol{x}_1, \boldsymbol{x}_2) \dots & k(\boldsymbol{x}_1, \boldsymbol{x}_N) \\ k(\boldsymbol{x}_2, \boldsymbol{x}_1) & k(\boldsymbol{x}_2, \boldsymbol{x}_2) \dots & k(\boldsymbol{x}_2, \boldsymbol{x}_N) \\ \vdots & \vdots & \vdots \\ k(\boldsymbol{x}_N, \boldsymbol{x}_1) & k(\boldsymbol{x}_N, \boldsymbol{x}_2) \dots & k(\boldsymbol{x}_N, \boldsymbol{x}_N) \end{bmatrix} \tag{14}$$

From (12), and conditioning on the available training samples, we can derive the expression of the model predictive distribution, yielding

$$p(f_*|\boldsymbol{x}_*, \mathcal{D}) = \mathcal{N}(f_*|\mu_*, \sigma_*^2) \tag{15}$$

$$\mu_* = \boldsymbol{k}(\boldsymbol{x}_*)^T (\boldsymbol{K}(X,X) + \sigma^2 \boldsymbol{I}_N)^{-1} Y \tag{16}$$

$$\sigma_*^2 = \sigma^2 - \boldsymbol{k}(\boldsymbol{x}_*)^T \left(\boldsymbol{K}(X,X) + \sigma^2 \boldsymbol{I}_N \right)^{-1} \boldsymbol{k}(\boldsymbol{x}_*) + k(\boldsymbol{x}_*, \boldsymbol{x}_*) \tag{17}$$

2) Gaussian process latent variable models: Building upon the GP model, the Gaussian process latent variable model (GPLVM) is essentially a GP the input variables $\boldsymbol{x}$ of which are considered latent variables rather than observed ones. Specifically, GPLVM considers that the $\boldsymbol{y} \in \mathbb{R}^D$ are observed multidimensional variables, while the latent vectors $\boldsymbol{x}$ are variables lying in some lower-dimensional manifold that generate the observed variables $\boldsymbol{y}$ through an unknown latent mapping f, modeled through a GP. This way, we have

$$p(\boldsymbol{y}|\boldsymbol{x}) = \prod_{d=1}^{D} \mathcal{N}(y_d|0, k_d(\boldsymbol{x}, \boldsymbol{x})) \tag{18}$$

which, considering a set of observations $Y = [\boldsymbol{y}_n]_{n=1}^N$, obtains

$$p(Y|X) = \prod_{d=1}^{D} \mathcal{N}(Y_d|\mathbf{0}, K_d(X,X)) \tag{19}$$

where $k_d(\cdot, \cdot)$ is the kernel of the model for the dth observed dimension, $X = [\boldsymbol{x}_n]_{n=1}^N$, $Y_d = [y_{nd}]_{n=1}^N$, and $K_d(X,X)$ is the gram matrix (with inputs X) corresponding to the dth dimension of the modeled data. GPLVM learns the values of the latent variables $\boldsymbol{x}$ corresponding to the observed data $\boldsymbol{y}$ through maximization of the model marginal likelihood, i.e. in a MAP fashion. As shown in [4], GPLVM can be viewed as a GP-based, nonparametric Bayesian extension of probabilistic principal component analysis (PPCA) [11], [12], a popular method for latent manifold modeling of high-dimensional data.

3) Dynamic Gaussian process latent variable models: Inspired from GPLVM, GPDM performs modeling of sequential data by considering a GP prior as in GPLVM, and introducing an additional model of temporal interdependencies between successive latent vectors $\boldsymbol{x}$. Specifically, considering a sequence of observed data $Y = [\boldsymbol{y}_n]_{n=1}^N$, where $\boldsymbol{y}_n = [y_{nd}]_{d=1}^D$, with corresponding latent manifold projections $X = [\boldsymbol{x}_n]_{n=1}^N$, $\boldsymbol{x}_n \in \mathbb{R}^Q$, GPDM models the dependencies between the latent and observed data in the form (19), while also considering a GP-based model of the interdependencies between the latent vectors $\boldsymbol{x}_n$. In detail, this latter model initially postulates

$$p(X) = p(\boldsymbol{x}_1) \int \prod_{n=2}^{N} p(\boldsymbol{x}_n|\boldsymbol{x}_{n-1}; \boldsymbol{A}) p(\boldsymbol{A}) \mathrm{d}\boldsymbol{A} \tag{20}$$

which, assuming

$$p(\boldsymbol{x}_n|\boldsymbol{x}_{n-1}; \boldsymbol{A}) = \mathcal{N}(\boldsymbol{x}_n|\boldsymbol{A}\boldsymbol{x}_{n-1}, \sigma^2) \tag{21}$$

and a simplistic isotropic Gaussian prior on the columns of the parameters matrix $\boldsymbol{A}$, yields a GP prior of the form

$$\begin{aligned} p(X) = & \frac{p(\boldsymbol{x}_1)}{\sqrt{(2\pi)^{Q(N-1)} |\boldsymbol{\Lambda}(X,X)|^Q}} \\ & \times \exp\left(-\frac{1}{2} \mathrm{tr}(\boldsymbol{\Lambda}(X,X)^{-1} X_{2:N} X_{2:N}^T) \right) \end{aligned} \tag{22}$$

where $\boldsymbol{\Lambda}(X,X)$ is the *gram* matrix:

$$\boldsymbol{\Lambda}(X,X) \triangleq \begin{bmatrix} k(\boldsymbol{x}_1, \boldsymbol{x}_1) & k(\boldsymbol{x}_1, \boldsymbol{x}_2) \dots & k(\boldsymbol{x}_1, \boldsymbol{x}_{N-1}) \\ k(\boldsymbol{x}_2, \boldsymbol{x}_1) & k(\boldsymbol{x}_2, \boldsymbol{x}_2) \dots & k(\boldsymbol{x}_2, \boldsymbol{x}_{N-1}) \\ \vdots & \vdots & \vdots \\ k(\boldsymbol{x}_{N-1}, \boldsymbol{x}_1) & k(\boldsymbol{x}_{N-1}, \boldsymbol{x}_2) \dots & k(\boldsymbol{x}_{N-1}, \boldsymbol{x}_{N-1}) \end{bmatrix} \tag{23}$$

Estimation of the set of (dynamically interdependent) latent variables X of the GPDM is performed by means of type-II maximum-likelihood, similar to GPLVM.

As we observe, GPDM utilizes a rather simplistic modeling approach for the dependencies between the latent variables $\boldsymbol{x}$, which is based on a linear model of dependencies, as in

(21), combined with a rather implausible spherical prior over $\boldsymbol{A}$. This model formulation is clearly restrictive, and limits the level of complexity of the temporal structure that GPDM can capture. In addition, the requirement of MAP estimation of the latent variables $\boldsymbol{x} \in \mathbb{R}^Q$ results in high computational requirements. This is aggravated even more by the form of the prior (22), which gives rise to computations entailing inverting the gram matrix $\boldsymbol{\Lambda}(X, X)$, in addition to computing the inverse of the matrices $\boldsymbol{K}_d(X, X)$. Our proposed model is designed to ameliorate these issues, as we shall discuss next.

III. PROPOSED APPROACH

A. Model definition

Let us consider a sequence of observations with strong temporal interdependencies $Y = [\boldsymbol{y}_n]_{n=1}^N$. Similar to GPLVM, we assume a latent lower-dimensional manifold that generates the observation space, and a smooth, nonlinear mapping from the latent space to the observations space. To infer this nonlinear mapping, we postulate a GP prior, similar to GPLVM. In addition though, we also postulate a model capable of capturing the temporal dynamics in the modeled data. For this purpose, we prefer to work in the lower-dimensional manifold of the latent vectors generating our observations. Specifically, we consider a generative, nonparametric Bayesian model for these latent coordinates: we assume they are generated from a hidden Markov model comprising infinite (latent) states. To make formulation of this latent manifold infinite-state HMM possible, we impose suitable stick-breaking priors over its state-transition dynamics, similar to the work of [13].

In particular, formulation of our model commences by considering a smooth nonlinear (generative) mapping of the latent vectors $\boldsymbol{x}_n$ to the observed ones $\boldsymbol{y}_n$, described by means of a GP, with the addition of some additive white noise. Given these assumptions, we yield a likelihood function of the form

$$p(\boldsymbol{y}_n|\boldsymbol{x}_n) = \prod_{d=1}^{D} \mathcal{N}(y_{nd}|f_d(\boldsymbol{x}_n), \beta^{-1}) \tag{24}$$

where β is the white noise precision, and $f_d(\boldsymbol{x}_n)$ is modeled by means of a GP prior, such that

$$p(\boldsymbol{f}_d|X) = \mathcal{N}(\boldsymbol{f}_d|\boldsymbol{0}, \boldsymbol{K}_d(X, X)) \tag{25}$$

where $\boldsymbol{f}_d$ is the vector of the $f_d(\boldsymbol{x}_n)$ $\forall n$, i.e., $\boldsymbol{f}_d \triangleq [f_d(\boldsymbol{x}_n)]_{n=1}^N$, and $\boldsymbol{K}_d(X, X)$ is the *gram matrix* pertaining to the dth dimension of the observations space, with kernel hyperparameters set ψ_d.

Subsequently, we consider that the latent manifold projections $\boldsymbol{x}$ are generated by an HMM comprising infinite states. Let us introduce the set of variables $Z = \{z_{nc}\}_{n,c=1}^{N,\infty}$, with $z_{nc} = 1$ if $\boldsymbol{x}_n$ is considered to be generated from the cth HMM model state, $z_{nc} = 0$ otherwise. Then, our introduced generative model for the latent vectors $\boldsymbol{x}_n$ comprises the assumptions [13]

$$p(\boldsymbol{x}_n|z_{nc} = 1) = \mathcal{N}(\boldsymbol{x}_n|\boldsymbol{\mu}_c, \boldsymbol{R}_c^{-1}) \tag{26}$$

$$p(z_{nj} = 1|z_{n-1,i} = 1; \boldsymbol{v}_i^{\varpi}) = \varpi_j(\boldsymbol{v}_i^{\varpi}),\ n > 1 \tag{27}$$

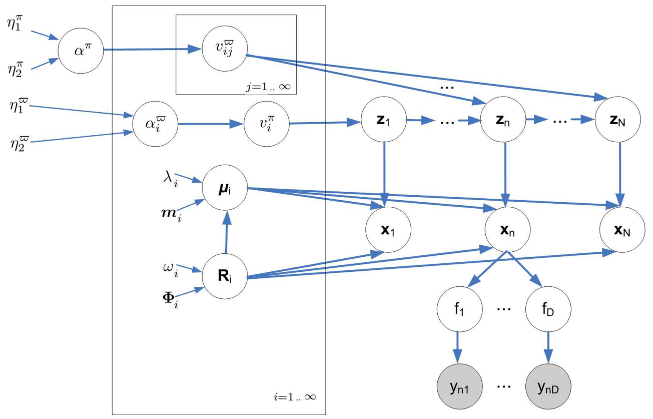

Figure 2. Plate diagram representation of the LM^{2}GP model. The arrows represent conditional dependencies. The plates indicate independent copies for states i, j.

$$\varpi_j(\boldsymbol{v}_i^{\varpi}) = v_{ij}^{\varpi} \prod_{k=1}^{j-1} (1 - v_{ik}^{\varpi}) \in [0, 1] \tag{28}$$

$$p(z_{1i} = 1|; \boldsymbol{v}^{\pi}) = \pi_i(\boldsymbol{v}^{\pi}) \tag{29}$$

$$\pi_i(\boldsymbol{v}^{\pi}) = v_i^{\pi} \prod_{j=1}^{i-1} (1 - v_j^{\pi}) \in [0, 1] \tag{30}$$

with

$$\sum_{j=1}^{\infty} \varpi_j(\boldsymbol{v}_i^{\varpi}) = 1\ \forall i \tag{31}$$

$$\sum_{j=1}^{\infty} \pi_j(\boldsymbol{v}^{\pi}) = 1 \tag{32}$$

For the stick-variables $\{\boldsymbol{v}_i^{\varpi}\}_{i=1}^{\infty}$, where $\boldsymbol{v}_i^{\varpi} = [v_{ij}^{\varpi}]_{j=1}^{\infty}$, and $\boldsymbol{v}^{\pi} = [v_i^{\pi}]_{i=1}^{\infty}$, we impose Beta priors [13] of the form

$$p(v_{ij}^{\varpi}) = \mathrm{Beta}(1, \alpha_i^{\varpi})\ \forall j \tag{33}$$

$$p(v_i^{\pi}) = \mathrm{Beta}(1, \alpha^{\pi})\ \forall i \tag{34}$$

In essence, this parameterization of the imposed Beta priors gives rise to a Dirichlet process-based prior construction for the transition dynamics of our model [6].

Finally, due to the effect of the innovation parameters on the number of effective latent Markov chain states, we also impose Gamma priors over them:

$$p(\alpha_i^{\varpi}) = \mathcal{G}(\alpha_i^{\varpi}|\eta_1^{\varpi}, \eta_2^{\varpi}) \tag{35}$$

$$p(\alpha^{\pi}) = \mathcal{G}(\alpha^{\pi}|\eta_1^{\pi}, \eta_2^{\pi}). \tag{36}$$

We also impose suitable conjugate priors over the mean and precision parameters of the (Markov chain-)state-conditional likelihoods of the latent vectors. Specifically, we choose Normal-Wishart priors, yielding

$$p(\boldsymbol{\mu}_c, \boldsymbol{R}_c) = \mathcal{NW}(\boldsymbol{\mu}_c, \boldsymbol{R}_c|\lambda_c, \boldsymbol{m}_c, \omega_c, \boldsymbol{\Phi}_c) \tag{37}$$

This completes the definition of our proposed LM^{2}GP model. A plate diagram of our model is provided in Fig. 2.

B. Inference algorithm

Inference for nonparametric models can be conducted under a Bayesian setting, typically by means of variational Bayes (e.g., [9]), or Markov Chain Monte Carlo techniques (e.g., [14]). Here, we prefer a variational Bayesian approach, due to its considerably better scalability in terms of computational costs, which becomes of major importance when having to deal with large data corpora.

Our variational Bayesian inference algorithm for the LM^{2}GP model comprises derivation of a family of variational posterior distributions $q(.)$ which approximate the true posterior distribution over the infinite sets Z, $\{\boldsymbol{\mu}_c, \boldsymbol{R}_c\}_{c=1}^{\infty}$, $\boldsymbol{v}^{\pi}$, $\{\boldsymbol{v}_i^{\varpi}\}_{i=1}^{\infty}$, and $\{\alpha_i^{\varpi}\}_{i=1}^{\infty}$, as well the parameters α^{π}, the latent mapping function values $\boldsymbol{f}_d = [f_d(\boldsymbol{x}_n)]_{n=1}^{N}$, and the latent mappings $X = [\boldsymbol{x}_n]_{n=1}^{N}$. Apparently, Bayesian inference is not tractable under this setting, since we are dealing with an infinite number of parameters.

For this reason, we employ a common strategy in the literature of Bayesian nonparametrics, formulated on the basis of a truncation of the stick-breaking process [9]. Specifically, we fix a value C and we let the variational posterior over the v_{ij}^{ϖ} and the v_i^{π} have the property $q(v_{iC}^{\varpi} = 1) = 1$, $\forall i = 1, \dots, C$, and $q(v_C^{\pi} = 1) = 1$. In other words, we set $\pi_c(\boldsymbol{v}^{\pi})$ and $\varpi_c(\boldsymbol{v}_i^{\varpi})$ equal to zero for $c > C$. Note that, under this setting, the treated LM^{2}GP model involves full stick-breaking priors; truncation is not imposed on the model itself, but only on the variational distribution to allow for tractable inference. Hence, the truncation level C is a variational parameter which can be freely set, and not part of the prior model specification.

Let $W \triangleq \{X, Z, \boldsymbol{v}^{\pi}, \alpha^{\pi}, \{\boldsymbol{f}_d\}_{d=1}^{D}, \{\boldsymbol{v}_c^{\varpi}, \alpha_c^{\varpi}, \boldsymbol{\mu}_c, \boldsymbol{R}_c\}_{c=1}^{C}\}$ be the set of all the parameters of the LM^{2}GP model over which a prior distribution has been imposed, and Ξ be the set of the hyperparameters of the model priors and kernel functions. Variational Bayesian inference introduces an arbitrary distribution $q(W)$ to approximate the actual posterior $p(W|\Xi, Y)$ which is computationally intractable, yielding [8]

$$\log p(Y) = \mathcal{L}(q) + \mathrm{KL}(q||p) \tag{38}$$

where

$$\mathcal{L}(q) = \int \mathrm{d}W q(W) \log \frac{p(Y, W|\Xi)}{q(W)} \tag{39}$$

and $\mathrm{KL}(q||p)$ stands for the Kullback-Leibler (KL) divergence between the (approximate) variational posterior, $q(W)$, and the actual posterior, $p(W|\Xi, Y)$. Since KL divergence is nonnegative, $\mathcal{L}(q)$ forms a strict lower bound of the log evidence, and would become exact if $q(W) = p(W|\Xi, Y)$. Hence, by maximizing this lower bound $\mathcal{L}(q)$ (variational free energy) so that it becomes as tight as possible, not only do we minimize the KL-divergence between the true and the variational posterior, but we also implicitly integrate out the unknowns W [8].

To facilitate variational Bayesian inference for our model, we assume that the posterior factorizes similar to the prior of our model (*mean-field* approximation) [15], [16]. This way, the variational free energy $\mathcal{L}(q)$ yields (ignoring constant terms):

$$\begin{aligned}
\mathcal{L}(q) =& \sum_{d=1}^{D} \int \int \mathrm{d}X \mathrm{d}\boldsymbol{f}_d q(X) q(\boldsymbol{f}_d) \Bigg[\log \frac{p(\boldsymbol{f}_d|X)}{q(\boldsymbol{f}_d)} \\
&\quad + \sum_{n=1}^{N} \log p(y_{nd}|f_d(\boldsymbol{x}_n), \beta^{-1}) \Bigg] \\
&+ \sum_{c=1}^{C} \int \int \mathrm{d}\boldsymbol{\mu}_c \mathrm{d}\boldsymbol{R}_c q(\boldsymbol{\mu}_c, \boldsymbol{R}_c) \\
&\quad \times \log \frac{p(\boldsymbol{\mu}_c, \boldsymbol{R}_c|\lambda_c, \boldsymbol{m}_c, \omega_c, \boldsymbol{\Phi}_c)}{q(\boldsymbol{\mu}_c, \boldsymbol{R}_c)} \\
&+ \int \mathrm{d}\alpha^{\pi} q(\alpha^{\pi}) \Bigg\{ \log \frac{p(\alpha^{\pi}|\eta_1^{\pi}, \eta_2^{\pi})}{q(\alpha^{\pi})} \\
&\quad + \sum_{c=1}^{C-1} \int \mathrm{d}v_c^{\pi} q(v_c^{\pi}) \log \frac{p(v_c^{\pi}|\alpha^{\pi})}{q(v_c^{\pi})} \Bigg\} \\
&+ \sum_{c=1}^{C-1} \int \mathrm{d}\alpha_c^{\varpi} q(\alpha_c^{\varpi}) \Bigg\{ \log \frac{p(\alpha_c^{\varpi}|\eta_1^{\varpi}, \eta_2^{\varpi})}{q(\alpha_c^{\varpi})} \\
&\quad + \sum_{c'=1}^{C-1} \int \mathrm{d}v_{cc'}^{\varpi} q(v_{cc'}^{\varpi}) \log \frac{p(v_{cc'}^{\varpi}|\alpha_c^{\varpi})}{q(v_{cc'}^{\varpi})} \Bigg\} \\
&+ \sum_{i=1}^{C} \sum_{j=1}^{C} \sum_{n=2}^{N} q(z_{nj} = 1|z_{n-1,i} = 1) \\
&\quad \times \int \mathrm{d}\boldsymbol{v}_i^{\varpi} q(\boldsymbol{v}_i^{\varpi}) \log \frac{p(z_{nj} = 1|z_{n-1,i} = 1; \boldsymbol{v}_i^{\varpi})}{q(z_{nj} = 1|z_{n-1,i} = 1)} \\
&+ \sum_{c=1}^{C} q(z_{1c} = 1) \int \mathrm{d}\boldsymbol{v}^{\pi} q(\boldsymbol{v}^{\pi}) \log \frac{p(z_{1c} = 1|\boldsymbol{v}^{\pi})}{q(z_{1c} = 1)} \\
&+ \sum_{c=1}^{C} \sum_{n=1}^{N} q(z_{nc} = 1) \int \int \int \mathrm{d}\boldsymbol{x}_n \mathrm{d}\boldsymbol{\mu}_c \mathrm{d}\boldsymbol{R}_c \\
&\quad \times q(\boldsymbol{x}_n) q(\boldsymbol{\mu}_c, \boldsymbol{R}_c) \log \frac{p(\boldsymbol{x}_n|\boldsymbol{\mu}_c, \boldsymbol{R}_c)}{q(\boldsymbol{x}_n)}
\end{aligned} \tag{40}$$

Derivation of the variational posterior distribution $q(W)$ involves maximization of the variational free energy $\mathcal{L}(q)$ over each one of the factors of $q(W)$ in turn, holding the others fixed, in an iterative manner [15]. On each iteration, apart from variational posterior updating, we also update the estimates of the model hyperparameters in Ξ, by maximization of the variational free energy $\mathcal{L}(q)$ over each one of them. By construction, this iterative, consecutive updating of the model is guaranteed to monotonically and maximally increase the free energy $\mathcal{L}(q)$ [17].

Variational posteriors. Let us denote as $\langle . \rangle_{q(\cdot)}$ the posterior expectation of a quantity, that is the quantity's mean value considering the entailed random variables follow the variational posteriors $q(\cdot)$. In the following, all these posterior means can be computed analytically, except for the expectations w.r.t. the posterior $q(\boldsymbol{x}_n)$ over the latent manifold representations of our modeled data. We shall elaborate on computation of these latter quantities later in this section.

From (40), we obtain the following variational (approximate) posteriors over the parameters of our model:

1. For the $q(\boldsymbol{v}_i^{\varpi})$, we have

$$q(v_{ij}^{\varpi}) = \text{Beta}(\tilde{\beta}_{ij}^{\varpi}, \hat{\beta}_{ij}^{\varpi}) \tag{41}$$

$$\tilde{\beta}_{ij}^{\varpi} = 1 + \sum_{n=2}^{N} q(z_{nj} = 1 | z_{n-1,i} = 1) \tag{42}$$

$$\hat{\beta}_{ij}^{\varpi} = \langle \alpha_i^{\varpi} \rangle_{q(\alpha_i^{\varpi})} + \sum_{\varrho=j+1}^{C} \sum_{n=2}^{N} q(z_{n\rho} = 1 | z_{n-1,i} = 1) \tag{43}$$

2. Similar, for the $q(\boldsymbol{v}^{\pi})$ we have

$$q(v_i^{\pi}) = \text{Beta}(\tilde{\beta}_i^{\pi}, \hat{\beta}_i^{\pi}) \tag{44}$$

$$\tilde{\beta}_i^{\pi} = 1 + q(z_{1i} = 1) \tag{45}$$

$$\hat{\beta}_i^{\pi} = \langle \alpha^{\pi} \rangle_{q(\alpha^{\pi})} + \sum_{\varrho=i+1}^{C} q(z_{1\rho} = 1) \tag{46}$$

3. For the $q(\alpha_i^{\varpi})$ we have

$$q(\alpha_i^{\varpi}) = \mathcal{G}(\alpha_i^{\varpi} | \tilde{\epsilon}_i^{\varpi}, \hat{\epsilon}_i^{\varpi}) \tag{47}$$

where

$$\tilde{\epsilon}_i^{\varpi} = \eta_1^{\varpi} + C - 1 \tag{48}$$

$$\hat{\epsilon}_i^{\varpi} = \eta_2^{\varpi} - \sum_{j=1}^{C-1} \left[\psi(\hat{\beta}_{ij}^{\varpi}) - \psi(\tilde{\beta}_{ij}^{\varpi} + \hat{\beta}_{ij}^{\varpi}) \right] \tag{49}$$

4. For the $q(\alpha^{\pi})$ we have

$$q(\alpha^{\pi}) = \mathcal{G}(\alpha^{\pi} | \tilde{\varepsilon}^{\pi}, \hat{\varepsilon}^{\pi}) \tag{50}$$

where

$$\tilde{\varepsilon}^{\pi} = \eta_1^{\pi} + C - 1 \tag{51}$$

$$\hat{\varepsilon}^{\pi} = \eta_2^{\pi} - \sum_{i=1}^{C-1} \left[\psi(\hat{\beta}_i^{\pi}) - \psi(\tilde{\beta}_i^{\pi} + \hat{\beta}_i^{\pi}) \right] \tag{52}$$

5. The posteriors $q(\boldsymbol{\mu}_c, \boldsymbol{R}_c)$ are approximated in the following form:

$$q(\boldsymbol{\mu}_c, \boldsymbol{R}_c) = \mathcal{NW}(\boldsymbol{\mu}_c, \boldsymbol{R}_c | \tilde{\lambda}_c, \tilde{\boldsymbol{m}}_c, \tilde{\omega}_c, \tilde{\boldsymbol{\Phi}}_c) \tag{53}$$

where we introduce the notation

$$\bar{\boldsymbol{x}}_c \triangleq \frac{\sum_{n=1}^{N} q(z_{nc} = 1) \langle \boldsymbol{x}_n \rangle_{q(\boldsymbol{x}_n)}}{\sum_{n=1}^{N} q(z_{nc} = 1)} \tag{54}$$

$$\boldsymbol{\Delta}_c \triangleq \sum_{n=1}^{N} q(z_{nc} = 1) \left(\langle \boldsymbol{x}_n \rangle_{q(\boldsymbol{x}_n)} - \bar{\boldsymbol{x}}_c \right) \left(\langle \boldsymbol{x}_n \rangle_{q(\boldsymbol{x}_n)} - \bar{\boldsymbol{x}}_c \right)^T \tag{55}$$

and, we have

$$\tilde{\omega}_c = \omega_c + \sum_{n=1}^{N} q(z_{nc} = 1) \tag{56}$$

$$\begin{aligned} \tilde{\boldsymbol{\Phi}}_c = & \frac{\lambda_c \sum_{n=1}^{N} q(z_{nc} = 1)}{\lambda_c + \sum_{n=1}^{N} q(z_{nc} = 1)} (\boldsymbol{m}_c - \bar{\boldsymbol{x}}_c)(\boldsymbol{m}_c - \bar{\boldsymbol{x}}_c)^T \\ & + \boldsymbol{\Phi}_c + \boldsymbol{\Delta}_c \end{aligned} \tag{57}$$

$$\tilde{\lambda}_c = \lambda_c + \sum_{n=1}^{N} q(z_{nc} = 1) \tag{58}$$

$$\tilde{\boldsymbol{m}}_c = \frac{\lambda_c \boldsymbol{m}_c + \bar{\boldsymbol{x}}_c \sum_{n=1}^{N} q(z_{nc} = 1)}{\lambda_c + \sum_{n=1}^{N} q(z_{nc} = 1)} \tag{59}$$

In the above expressions, $\langle \boldsymbol{x}_n \rangle_{q(\boldsymbol{x}_n)}$ is the (posterior) expectation of the latent variable $\boldsymbol{x}_n$ given its posterior $q(\boldsymbol{x}_n)$.

6. Regarding the posteriors over the latent functions $\boldsymbol{f}_d$, we have

$$q(\boldsymbol{f}_d) = \mathcal{N}(\boldsymbol{f}_d | \hat{\boldsymbol{\mu}}_d, \boldsymbol{\Sigma}_d) \tag{60}$$

where

$$\boldsymbol{\Sigma}_d = \left(\left\langle \boldsymbol{K}_d(X,X)^{-1} \right\rangle_{q(X)} + \beta \boldsymbol{I} \right)^{-1} \tag{61}$$

$$\hat{\boldsymbol{\mu}}_d = \beta \boldsymbol{\Sigma}_d \boldsymbol{I} Y_d \tag{62}$$

$Y_d \triangleq [y_{nd}]_{n=1}^{N}$, and $\left\langle \boldsymbol{K}_d(X,X)^{-1} \right\rangle_{q(X)}$ is the posterior mean of the inverse gram matrix $\boldsymbol{K}_d(X,X)^{-1}$ given the variational posteriors $q(\boldsymbol{x}_n)$ $\forall n$.

7. Similar, the posteriors over the indicator variables Z yield

$$q(Z) \propto \pi_{\delta_1}^{*} \prod_{n=1}^{N-1} \varpi_{\delta_n \delta_{n+1}}^{*} \prod_{n=1}^{N} p^{*}(\boldsymbol{x}_n | \boldsymbol{\mu}_{\delta_n}, \boldsymbol{R}_{\delta_n}) \tag{63}$$

where

$$\pi_c^{*} \triangleq \exp \left[\langle \log \pi_c(\boldsymbol{v}^{\pi}) \rangle_{q(\boldsymbol{v}^{\pi})} \right] \tag{64}$$

$$\varpi_{ij}^{*} \triangleq \exp \left[\langle \log \varpi_j(\boldsymbol{v}_i^{\varpi})_{q(\boldsymbol{v}_i^{\varpi})} \rangle \right] \tag{65}$$

$$p^{*}(\boldsymbol{x}_t | \boldsymbol{\mu}_i, \boldsymbol{R}_i) \triangleq \exp \left[\langle \log p(\boldsymbol{x}_t | \boldsymbol{\mu}_i, \boldsymbol{R}_i) \rangle_{q(\boldsymbol{\mu}_i, \boldsymbol{R}_i), q(\boldsymbol{x}_t)} \right] \tag{66}$$

and

$$\delta_n \triangleq \arg_c (z_{nc} = 1)$$

From (63), and comparing this expression to the corresponding expressions of standard HMMs [18], it is easy to observe that computation of the probabilities $q(z_{nj} = 1 | z_{n-1,i} = 1)$, and $q(z_{nc} = 1)$, which constitute the variational posterior $q(Z)$, can be easily performed by means of the *forward-backward algorithm* for simple HMMs trained by means of maximum-likelihood, exactly as described in [19]; specifically, it suffices to run *forward-backward* for a simple HMM model with its optimized values ("point"-estimates) of the Markov chain probabilities set equal to the posterior expected values π_c^{*}, ϖ_{ij}^{*} in (64)-(65), and its state-conditional likelihoods set equal to the expectations $p^{*}(\boldsymbol{x}_t | \boldsymbol{\mu}_i, \boldsymbol{R}_i)$.

8. Finally, regarding the latent variable posteriors $q(\boldsymbol{x}_n)$, optimization over $\mathcal{L}(q)$ yields

$$\begin{aligned} \log q(\boldsymbol{x}_n) = & \sum_{c=1}^{C} q(z_{nc} = 1) \langle \log p(\boldsymbol{x}_n | z_{nc} = 1) \rangle_{q(\boldsymbol{\mu}_c, \boldsymbol{R}_c)} \\ & + \sum_{d=1}^{D} \langle \log p(\boldsymbol{f}_d | \boldsymbol{0}, \boldsymbol{K}_d(X,X)) \rangle_{q(\boldsymbol{f}_d)} \end{aligned} \tag{67}$$

From (67), it becomes apparent that model configuration is not conjugate when it comes to the latent variables $\boldsymbol{x}_n$. As a consequence, the model does not yield a closed-form expression for the posteriors $q(\boldsymbol{x}_n)$. A repercussion of this fact is that the entailed posterior expectations w.r.t. $q(\boldsymbol{x}_n)$ cannot be computed in an analytical fashion, wherever they appear in the inference algorithm of our model (namely, the quantities $\left\langle \boldsymbol{K}_d(X,X)^{-1} \right\rangle_{q(\boldsymbol{x}_n)}$ and $\langle \boldsymbol{x}_n \rangle_{q(\boldsymbol{x}_n)}$). To resolve this issue, we can either resort to a deterministic approximation of the posteriors $q(\boldsymbol{x}_n)$, e.g., by application of Laplace approximation, or perform sampling by means of MCMC. Our investigations have shown that Laplace approximation does not perform very well for our model. For this reason, in this paper we opt for the latter alternative.

Specifically, for this purpose we draw samples from the variational posterior (67) using hybrid Monte Carlo (HMC) [20]. HMC provides an efficient method to draw samples from the variational posterior distribution $q(\boldsymbol{x}_n)$ by performing a physical simulation of an energy-conserving system to generate proposal moves. In detail, we add "kinetic energy" variables, p, for each latent dimension, while the expression of $-\log q(\boldsymbol{x}_n)$ is used to specify a potential energy over the latent variables. Each HMC step involves sampling p and carrying out a physics-based simulation using "leap-frog" discretization. The final state of the simulation is accepted or rejected based on the Metropolis-Hastings algorithm. HMC sampling of $\boldsymbol{x}_n$ requires computation of the gradient of $q(\boldsymbol{x}_n)$, which can be analytically performed in a straightforward manner.

Hyperparameter Selection. Regarding the values of the hyperparameters of the model priors, we set $\boldsymbol{m}_c = \boldsymbol{0}$, $\lambda_c = 1$, $\omega_c = 10$, $\boldsymbol{\Phi}_c = 100\boldsymbol{I}$. Regarding the hyperparameters of the kernel functions $k_d(\cdot,\cdot)$, we estimate them by performing maximization of the model variational free energy $\mathcal{L}(q)$ over each one of them. Computation of the variational free energy $\mathcal{L}(q)$ and its gradient requires derivation of the entailed posterior expectations in (40), which can be obtained analytically, except for the expectations w.r.t. the posteriors $q(\boldsymbol{x}_n)$. These latter quantities can be obtained by means of MCMC, utilizing the samples of $\boldsymbol{x}_n$, previously drawn by HMC sampling. Given the fact that none of the sought quantities yields closed-form estimators, to perform $\mathcal{L}(q)$ maximization we resort to the L-BFGS algorithm [21].

C. Predictive density

Let us now consider the problem of computing the probability of a given sequence of observations $Y^* = [\boldsymbol{y}_n^*]_{n=1}^{N_*}$ w.r.t. a trained LM^{2}GP model. This quantity is useful, e.g., in applying our model to (sequence-level) classification applications. For this purpose, we need to derive the predictive density of the model

$$p(Y^*|Y) = \int\int p(Y^*|X^*;Y,X)p(X^*|X,Y)p(X|Y)\mathrm{d}X\mathrm{d}X^* \tag{68}$$

To begin with, the expression of $p(Y^*|X^*;Y,X)$ is analogous to the predictive density expression of standard GP regression, yielding

$$p(Y^*|X^*;Y,X) = \prod_{n=1}^{N^*}\prod_{d=1}^{D}\mathcal{N}(y_{nd}^*|a_{nd}^*,(\sigma_{nd}^*)^2) \tag{69}$$

$$a_{nd}^* = \boldsymbol{k}_d(\boldsymbol{x}_n^*)^T\left(\boldsymbol{K}_d(X,X)+\beta^{-1}\right)^{-1}Y_d \tag{70}$$

$$(\sigma_{nd}^*)^2 = -\boldsymbol{k}_d(\boldsymbol{x}_n^*)^T\left(\boldsymbol{K}_d(X,X)+\beta^{-1}\right)^{-1}\boldsymbol{k}_d(\boldsymbol{x}_n^*) + k_d(\boldsymbol{x}_*,\boldsymbol{x}_*) \tag{71}$$

where

$$\boldsymbol{k}_d(\boldsymbol{x}_n^*) \triangleq [k_d(\boldsymbol{x}_1,\boldsymbol{x}_n^*),...,k_d(\boldsymbol{x}_N,\boldsymbol{x}_n^*)]^T \tag{72}$$

Now, regarding the computation of the entailed expectation of $p(Y^*|X^*;Y,X)$ w.r.t. the distributions $p(X|Y)$ and $p(X^*|X,Y)$, we proceed as follows: First, we substitute $p(X|Y)$ with its approximation (variational posterior) $q(X) = \prod_{n=1}^{N} q(\boldsymbol{x}_n)$. This way, the corresponding expectation can be approximated by making use of the samples of the latent variables $\boldsymbol{x}_n$ previously drawn through HMC. Finally, to obtain the expectations w.r.t. the density $p(X^*|X,Y)$, we resort to Gibbs sampling from the inferred latent-manifold infinite-state HMM. This obtains a set of samples of X^* from the latent manifold which can be used to approximate the sought (remaining) expectation w.r.t. $p(X^*|X,Y)$.

D. Relations to existing approaches

As we have already discussed, our model is related to the GPDM method of [2]. The main difference between our work and GPDM is that the latter uses a simplistic linear model of temporal dynamics combined with a spherical GP prior over the latent variables. In contrast, our approach employs a more flexible generative construction, which considers latent variable emission from a latent stick-breaking hidden Markov model. In addition, one can also observe that our method essentially reduces to a GPLVM model by setting the truncation threshold C equal to one, i.e. $C = 1$.

IV. EXPERIMENTS

Here, we experimentally evaluate our method in three scenarios: (i) Unsupervised segmentation (frame-level clustering) of sequential data; (ii) supervised segmentation (frame-level classification) of sequential data; and (iii) (whole) sequence classification.

In the context of our unsupervised sequence segmentation experiments, we estimate the assignment of the data points (frames) of each sequence to the discovered latent classes (model states) using $q(Z)$, and compare the estimated values to the available groundtruth sequence segmentation. To compute the optimal assignment of the discovered class labels to the available groundtruth class labels, we resort to the Munkres assignment algorithm. This set of experiments makes it thus possible for us to evaluate the quality of the latent temporal dynamics discovered by our model.

On the other hand, the considered supervised sequence segmentation experiments allow for us to additionally evaluate the quality of the drawn latent manifold representations of the modeled data. Specifically, in this set of experiments, the drawn latent manifold representations $\boldsymbol{x}_n$ are subsequently fed to a simple multiclass classifier (specifically, a multiclass linear support vector machine [22]), which is trained to discover the correct (frame-level) classes by being presented with the generated manifold representations $\boldsymbol{x}_n$. We expect that a good performance of our model under such an experimental setup would indicate high quality representational capabilities for both the generated latent manifold representations $\boldsymbol{x}_n$ and the temporal dynamics discovered by our model. Further, in the context of this set of experiments, we also explore whether our model could be used to obtain a deep learning architecture, by stacking multiple layers of models, each one fed with the latent state representations generated from the previous layer (and the first layer fed with the actual observations). We believe that, by stacking multiple layers of models, we might be capable of extracting more complex, higher-level temporal dynamics encoded in the final-layer latent representations $\boldsymbol{x}_n$, thus allowing for eventually obtaining increased supervised (frame-level) classification performance.

Finally, in the case of the sequence classification experiments, we train one model for each one of the considered classes, and evaluate our method by assigning each test sequence to the class the model of which yields the highest predictive probability for that sequence.

To obtain some comparative results, in our experiments, apart from our method, we also evaluate GPLVM [4] and GPDM models [2], as well as other state-of-the-art approaches relevant to each scenario. In our experiments, we make use of the large-scale linear SVM library of [23], and the HMM-SVM implementation of Thorsten Joachims. In all our experiments, we use RBF kernels for all the evaluated GP-based models. Finally, HMC is run for 100 iterations, with 25 leap-frog steps at each iteration, and with a step-length equal to $0.001\sqrt{N}$, where N is the number of modeled data points.

A. *Unsupervised sequence segmentation*

1) Workflow recognition: We first consider an unsupervised sequence segmentation (frame-level clustering) application. For this purpose, we use a public benchmark dataset involving action recognition of humans, namely the workflow recognition (WR) database [24]. Specifically, we use the first two workflows pertaining to car assembly (see [24] for more details). The frame-level *tasks to recognize* in an unsupervised fashion in these workflows are the following:

1) Worker 1 picks up part 1 from rack 1 (upper) and places it on the welding cell; mean duration is 8-10 sec.
2) Worker 1 and worker 2 pick part 2a from rack 2 and place it on the welding cell.
3) Worker 1 and worker 2 pick part 2b from rack 3 and place it on the welding cell.
4) Worker 2 picks up spare parts 3a, 3b from rack 4 and places them on the welding cell.
5) Worker 2 picks up spare part 4 from rack 1 and places it on the welding cell.
6) Worker 1 and worker 2 pick up part 5 from rack 5 and place it on the welding cell.

Feature extraction is performed as follows: To extract the spatiotemporal variations, we use pixel change history images to capture the motion history (see, e.g., [25]), and compute the complex Zernike moments $A_{00}, A_{11}, A_{20}, A_{22}, A_{31}, A_{33}, A_{40}, A_{42}, A_{44}, A_{51}, A_{53}, A_{55}, A_{60}, A_{62}, A_{64}, A_{66}$, for each of which we compute the norm and the angle. Additionally the center of gravity and the area of the found blobs are also used. In total, this feature extraction procedure results in *31-dimensional observation vectors.* Zernike moments are calculated in rectangular regions of interest of approximately 15K pixels in each image to limit the processing and allow real time feature extraction (performed at a rate of approximately 50-60 fps). In our experiments, we use a total of *40 sequences* representing full assembly cycles and containing at least one of the considered behaviors, with *each sequence being approximately 1K frames long*. Frame annotation has been performed manually.

Table I
UNSUPERVISED SEQUENCE SEGMENTATION: *Workflow Recognition*. ERROR RATES (%) FOR OPTIMAL LATENT MANIFOLD DIMENSIONALITY.

Model	Mean Performance	Error Variance
LM^{2}GP	40.5	0.037
GPDM	45.31	0.046
GPLVM	47.39	0.075
SB-HMM	43.17	0.057

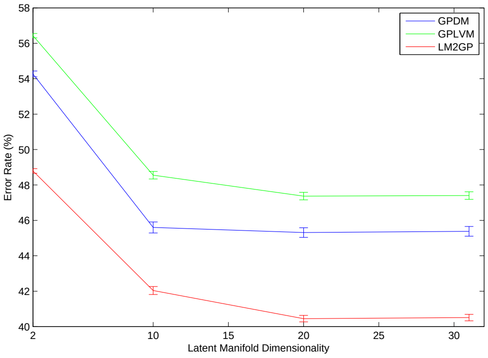

Figure 3. Unsupervised sequence segmentation: *Workflow Recognition*. Error rate fluctuation with latent manifold dimensionality.

In Table I, we illustrate the obtained error rates for optimal selection of the latent manifold dimensionality of our method, as well as the GPDM and GPLVM methods. Regarding the GPDM and GPLVM methods, clustering was performed by presenting the generated latent subspace representations to a stick-breaking HMM (SB-HMM) [13]. All these results are means and variances over 10 repetitions of the experiment. As a baseline, we also evaluate SB-HMMs presented with the original (observed) data. As we observe, our approach yields a clear advantage over the competition; we also observe that all the other latent manifold models yield inferior performance compared to the baseline SB-HMM. This result is a clear indication of the much improved capability of our approach to capture latent temporal dynamics in the modeled data.

In addition, in Fig. 3 we show how model performance changes as a function of the postulated latent manifold dimensionality, for the LM^{2}GP, GPDM, and GPLVM methods. We observe that model performance increases with manifold dimensionality in a consistent fashion. We also observe that our model benefits the most by an increase in latent manifold dimensionality. Further, we run the Student's-t test on all pairs of performances across the evaluated methods, to assess the statistical significance of the obtained differences; we obtain that all performance differences are deemed statistically significant.

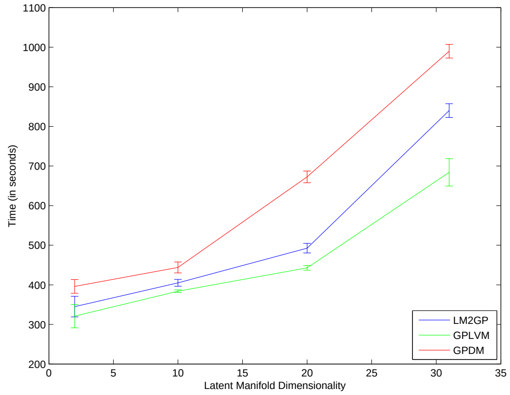

Figure 4. Workflow Recognition. Average execution time (per sequence) of inference algorithm.

Finally, in Fig. 4 we show an indicative empirical comparison of computational times, for processing one of the sequences in our dataset (means and error bars over all the used sequences). As we observe, GPDM requires the longest time between the compared methods; our method is faster than GPDM, while imposing a reasonable increase in computational costs compared to GPLVM. This difference from GPDM was expected, since our method involves one less Gram matrix compared to GPDM.

2) Honeybee dataset: Further, we perform a second unsupervised sequence segmentation experiment using the Honeybee dataset [26]; it contains video sequences of honeybees which communicate the location and distance to a food source through a dance that takes place within the hive. The dance can be decomposed into *three* different movement patterns that must be recognized by the evaluated algorithms: *waggle, right-turn,* and *left-turn*. During the waggle dance, the bee moves roughly in a straight line while rapidly shaking its body from left to right; the duration and orientation of this phase correspond to the distance and the orientation to the food source. At the endpoint of a waggle dance, the bee turns in a clockwise or counter-clockwise direction to form a turning dance.

Our dataset consists of *six video sequences with lengths 1058, 1125, 1054, 757, 609, and 814 frames, respectively*. The bees were visually tracked, and their locations and head angles were recorded. This resulted in obtaining *4-dimensional frame-level feature vectors* comprising the 2D location of the bee and the sine and cosine of its head angle. Once the sequence observations were obtained, the trajectories were preprocessed as in [27]. Specifically, the trajectory sequences were rotated so that the waggle dances had head angle measurements centered about zero radian. The sequences were then translated to center at (0, 0), and the 2D coordinates were scaled to the [-1,1] range. Aligning the waggle dances was possible by looking at the high frequency portions of the head angle measurements. Following the suggestion of [26], the data was smoothed using a Gaussian FIR pulse-shaping filter with 0.5dB bandwidth-symbol time.

Table II

UNSUPERVISED SEQUENCE SEGMENTATION: *Honeybee*. ERROR RATES (%) FOR OPTIMAL LATENT MANIFOLD DIMENSIONALITY.

Model	Mean Performance	Error Variance
LM^{2}GP	41.85	0.017
GPDM	48.26	0.018
GPLVM	47.16	0.026
SB-HMM	42.02	0.022

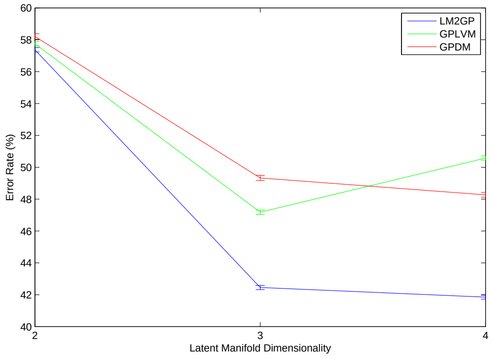

Figure 5. Unsupervised sequence segmentation: *Honeybee*. Error rate fluctuation with latent manifold dimensionality.

In our evaluations, we adopt the experimental setup of the previous experiment. In Fig. 5, we show how model performance changes as a function of the postulated latent manifold dimensionality, for the LM^{2}GP, GPDM, and GPLVM methods. In Table II, we provide the error rates of all the evaluated models for optimal latent manifold dimensionality (wherever applicable). These results are means and variances over 10 repetitions of the experiment. We again observe that both GPDM and GPLVM are outperformed by the baseline SB-HMM. Our method works better than all the alternatives. Finally, we run the Student's-t test on all pairs of performances across the evaluated methods, to assess the statistical significance of the obtained differences; we obtain that all performance differences are deemed statistically significant, except for the differences between the optimal performance of GPLVM and the optimal performance of GPDM (obtained for 3 and 4 latent features, respectively).

B. Supervised sequence segmentation

1) Workflow recognition: Here, we use the same dataset as in Section IV.A.1, but with the goal to perform supervised sequence segmentation using the obtained latent manifold representations of the modeled data. For this purpose, we first use our approach to obtain the latent manifold representations of the considered datasets. Subsequently, we use *half of the resulting representations (corresponding to all the modeled classes) for initial training* of a (frame-level) multiclass linear SVM classifier [22], and *keep the rest for testing* of the obtained classifier. Apart from our method, we also evaluate GPDM and GPLVM under the same experimental setup. In addition, as a baseline, we evaluate the HMM-SVM approach of [28] (presented with the *original* data, *not* the latent encodings).

Table III
SUPERVISED SEQUENCE SEGMENTATION: *Workflow Recognition*. ERROR RATES (%) FOR OPTIMAL LATENT MANIFOLD DIMENSIONALITY.

Model	Mean Performance	Error Variance
LM^{2}GP	27.72	0.020
GPDM	32.82	0.022
GPLVM	31.40	0.022
HMM-SVM	41.66	0.014

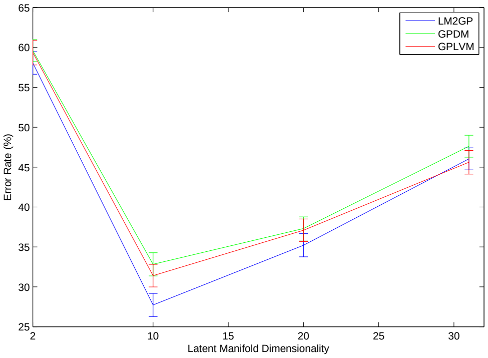

Figure 6. Supervised sequence segmentation: *Workflow Recognition*. Error rate fluctuation with latent manifold dimensionality.

In Table III, we provide the obtained classification error rates for the considered models. These results correspond to optimal latent manifold dimensionality selection for the LM^{2}GP, GPDM, and GPLVM methods, and constitute means and variances over 10 repetitions of the experiment. As we observe, our method outperforms the competition. We also observe that GPDM and GPLVM yield essentially identical results; this fact indicates that GPDM doesn't capture richer temporal dynamics in our data compared to GPLVM. Another significant finding from these results is that the state-of-the-art HMM-SVM approach yields a considerably inferior performance compared to the evaluated latent variable models. This result seems to indicate that extracting latent manifold representations of the modeled data is much more effective for classification purposes than directly using the observed data and assuming temporal dynamics in the observations space. In addition, in Fig. 6 we show how model performance changes with the number of latent features for the LM^{2}GP, GPDM, and GPLVM methods. We observe that all methods yield their optimal performance with 10 latent features.

Further, we turn to our deep learning scenario, examining how model performance changes if we add a second layer of models (with the same latent dimensionality), fed with the latent manifold representations generated by the original models. We present the output (latent manifold representations) of the second-layer models as input to a multiclass linear SVM classifier, and we evaluate the classification performance of the so-obtained classifier in the considered tasks.

In Fig. 7, we illustrate the classification error rates of the resulting models as a function of latent manifold dimensionality. As we observe, 2-layer deep learning architectures yield a significant improvement over the corresponding single-layer architectures when the latent manifold dimensionality is low; however, performance for high latent manifold dimensionality turns out to be similar in the cases of GPDM and our model, and substantially worse in the case of GPLVM. To our perception, this result indicates that our model encodes richer and more consistent temporal dynamics information in the obtained latent manifold representations, that can be combined in a hierarchical manner to extract even more complex temporal dynamics.

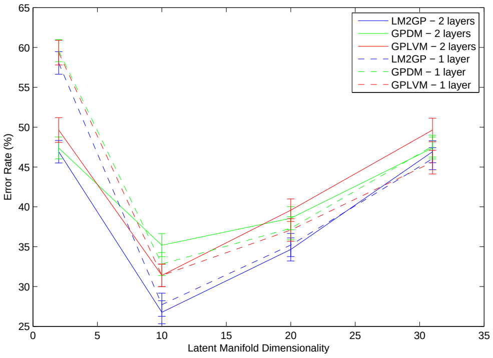

Figure 7. Supervised sequence segmentation: *Workflow Recognition*. Error rate fluctuation with latent manifold dimensionality for 2-layer deep learning architectures.

Finally, the Student's-t test on all pairs of performances across the evaluated methods obtains here some very interesting results:

(i) Our method yields statistically significant differences from GPDM and GPLVM for latent manifold dimensionality between 10 and 20. However, for more or less latent dimensions the obtained differences are not statistically significant.

(ii) GPDM and GPLVM are comparable, with no statistically significant differences, whatsoever.

(iii) Models with two layers have statistically significant performance differences w.r.t. similar one-layer models in cases of low latent manifold dimensionality. However, these differences quickly become insignificant as the number of latent dimensions increases.

2) Honeybee dataset: Further, we perform additional frame-level classification experiments using the Honeybee dataset described in Section IV.A.2. In our evaluations, *we adopt the experimental setup of the previous experiment.* First, we consider a shallow modeling scenario. In Table IV, we provide the obtained classification error rates for the considered models. These results correspond to optimal latent manifold dimensionality selection for the LM^{2}GP, GPDM, and GPLVM methods, and are means and variances over 10 repetitions of the experiment.

As we observe, our method outperforms the competition. We also observe that GPDM works better than GPLVM. Another significant finding from these results is that the state-of-the-art HMM-SVM approach yields again inferior performance compared to the evaluated latent variable models. This result corroborates our intuition that extracting latent manifold representations of the modeled data is much more effective for classification purposes than directly using the observed data and assuming temporal dynamics in the observations space. In

Table IV
SUPERVISED SEQUENCE SEGMENTATION: *Honeybee*. ERROR RATES (%) FOR OPTIMAL LATENT MANIFOLD DIMENSIONALITY.

Model	Mean Performance	Error Variance
LM^{2}GP	46.13	0.037
GPDM	47.17	0.041
GPLVM	47.69	0.043
HMM-SVM	51.07	0.021

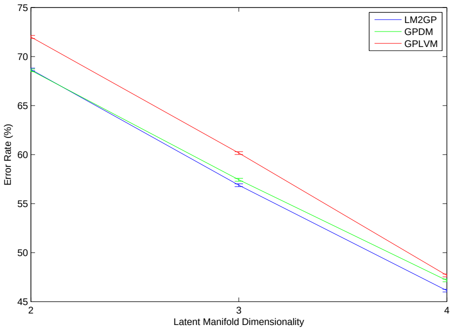

Figure 8. Supervised sequence segmentation: *Honeybee*. Error rate fluctuation with latent manifold dimensionality.

addition, in Fig. 8 we show how model performance changes with the number of latent features for the LM^{2}GP, GPDM, and GPLVM methods. We observe a consistent performance improvement with latent manifold dimensionality. This is not unexpected, since we are dealing with models trained on low-dimensional data using some thousands of training points.

Further, we turn to our deep learning scenario, examining how model performance changes if we add a second layer of models (with the same latent dimensionality), fed with the latent manifold representations generated by the original models. In Fig. 9, we illustrate the classification error rates of the resulting models as a function of latent manifold dimensionality. As we observe, 2-layer deep learning architectures yield improved performance for low latent manifold dimensionality (2-dimensional latent space); however, any performance gains quickly disappear as latent dimensionality increases, with GPLVM performance eventually becoming inferior to the shallow modeling scenario, similar to the previous experiment.

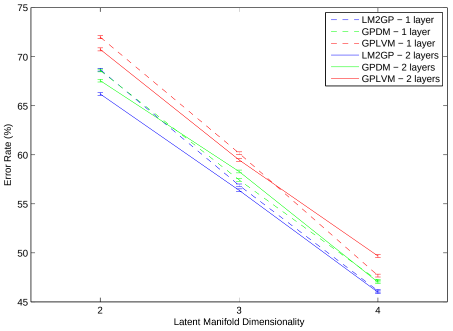

Figure 9. Supervised sequence segmentation: *Honeybee*. Error rate fluctuation with latent manifold dimensionality for 2-layer deep learning architectures.

Finally, the Student's-t test in this scenario finds that all pairs of differences are statistically significant, except for the case of 1-layer vs 2-layer model comparison, where the assumption of statistical significance is rejected for models with 4 latent dimensions.

C. Sequence-level classification

1) Learning music-to-dance mappings: Finally, we evaluate our method in sequence-level classification (classification of whole sequences). Initially, we consider the problem of learning music-to-dance mappings. In this experiment, the observed sequences presented to our model constitute the chroma features extracted from a collection of music clips. Chroma analysis [29] is an interesting and powerful representation for music audio in which the entire spectrum is projected onto 12 bins representing the 12 distinct semitones (or chroma) of the musical octave. Since, in music, notes exactly one octave apart are perceived as particularly similar, knowing the distribution of chroma even without the absolute frequency (i.e. the original octave) can give useful musical information about the audio, and may even reveal perceived musical similarity that is not apparent in the original spectra [30].

In our experiments, we *use a dataset of 600 music clips*; they are split into sets of 100 clips pertaining to each one of the dance classes: *Waltz, Tango, Foxtrot, Cha Cha, Quickstep,* and *Samba*[1]. We preprocess these clips as described above to obtain chroma features; this way, we eventually obtain sequences of *12-dimensional observations,* each one 35K-184K frames long. The *clips are further segmented into approximately 8K frames long subsequences* to form our dataset. *We use half our data for model training, and the rest for testing*. We train one model for each one of the 6 classes we want to recognize. Classification of our test sequences is conducted by computing the sequence predictive densities with respect to the model of each class, and assigning each sequence to the class the model of which yields the highest predictive density. Apart from our method, we also evaluate SB-HMMs [13], GPLVMs [4], and GPDMs [2] under the same experimental setup.

The obtained error rates for optimal latent manifold dimensionality (wherever applicable) are depicted in Table V. These results are means and variances over 10 repetitions of the experiment. As we observe, our approach works much better than the competition. We also observe that, in this experiment, GPDM does actually work better than GPLVM. In Fig. 10, we show how average model performance changes with latent manifold dimensionality. It becomes apparent from this graph that the modeled data contain a great deal of redundancy, since all models yield a performance deterioration for high latent space dimensionality. Finally, the Student's-t test finds that all performance differences are statistically significant.

2) Bimanual gesture recognition: Further, we perform evaluations considering the problem of bimanual gesture recognition. For this purpose, we experiment with the American

---

[1] Music clips were downloaded from: http://www.ballroomdancers.com/Music/Default.asp?Tab=2; we converted them to WAV format for our experiments.

Table V
SEQUENCE-LEVEL CLASSIFICATION: *Learning music-to-dance mappings*. ERROR RATES (%) FOR OPTIMAL LATENT MANIFOLD DIMENSIONALITY.

Model	SB-HMM	GPLVM	GPDM	LM^{2}GP
Error Rate	35.41	32.73	31.48	29.6
Error Variance	0.091	0.108	0.103	0.095

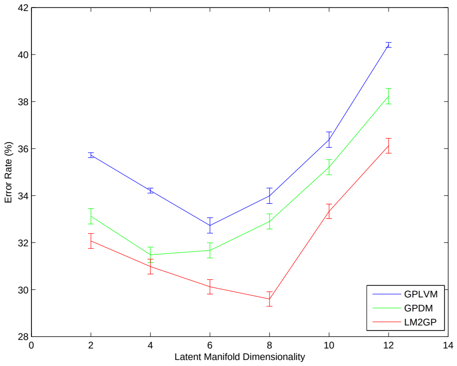

Figure 10. Sequence-level classification: *Learning music-to-dance mappings*. Error rate fluctuation with latent manifold dimensionality.

Sign Language gestures for the words: *against, aim, balloon, bandit, cake, chair, computer, concentrate, cross, deaf, explore, hunt, knife, relay, reverse,* and *role*. The used dataset was obtained from four different persons executing each one of these gestures and comprises *40 videos per gesture. 30 of these videos are used for training and the rest for model evaluation*. From this dataset, we extracted several features representing the relative position of the hands and the face in the images, as well as the shape of the respective skin regions, by means of the complex Zernike moments [31], as described in [18]. This way, each used video comprises *1K-4K frames of 12-dimensional feature vectors* used in our experiments.

For each one of these 16 gestures, we fitted one model to recognize it. The obtained error rates (means and variances over 10 experiment repetitions) for optimal latent manifold dimensionality (wherever applicable) are depicted in Table VI. As we observe, our approach works much better than the competition. Further, in Fig. 11 we show how average model performance changes with latent manifold dimensionality. We observe that all models yield a performance increase for moderate latent space dimensionality. Finally, the Student's-t test finds that all performance differences are statistically significant.

Table VI
SEQUENCE-LEVEL CLASSIFICATION: *Bimanual gesture recognition*. ERROR RATES (%) FOR OPTIMAL LATENT MANIFOLD DIMENSIONALITY.

Model	SB-HMM	GPLVM	GPDM	LM^{2}GP
Error Rate	11.44	13.81	6.76	4.97
Error Variance	0.38	0.42	0.46	0.41

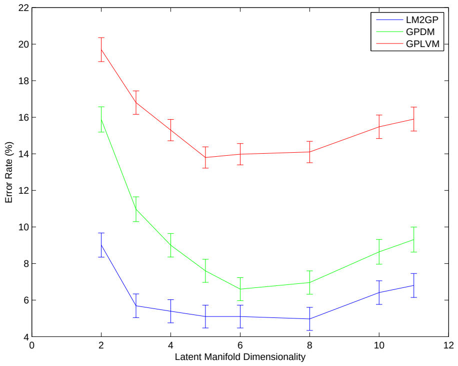

Figure 11. Sequence-level classification: *Bimanual gesture recognition*. Error rate fluctuation with latent manifold dimensionality.

V. CONCLUSIONS

In this paper, we proposed a method for sequential data modeling that leverages the strengths of Gaussian processes, allowing for more flexibly capturing temporal dynamics compared to existing nonparametric Bayesian approaches. Our method considers a latent manifold representation of the modeled data, and chooses to postulate a model of temporal dependencies on this latent manifold. Temporal dependencies in our model are captured through consideration of infinite-state Markovian dynamics, and imposition of stick-breaking priors over the entailed Markov chain probabilities. Inference for our model was performed by means of an efficient variational Bayesian algorithm.

As we showed through experimental evaluations, our approach is suitable for unsupervised sequence segmentation (frame-level clustering), supervised sequence segmentation (frame-level classification), and whole sequence classification (sequence-level operation). We evaluated our approach in all these scenarios using real-world datasets, and observed that our method yields very competitive results, outperforming popular, recently proposed related approaches, e.g. GPDM and GPLVM.

Finally, we examined whether our method can be also employed to obtain a deep learning architecture, by stacking multiple (layers of) LM^{2}GP models, each one fed with the latent manifold representations generated from the previous layer (and the first one fed with the observable data); specifically, we experimented with two-layer architectures. As we observed, our method seems to yield much more significant gains in such a scenario than GPDM- or GPLVM-based models, especially for low-dimensional manifold assumptions.

Our future research endeavors in this line of research mainly focus on addressing two open questions: The first one concerns the possibility of sharing the precision matrices between "close" hidden states. A question that must be answered is what proximity criterion we could use for this purpose in the context of our model. Would, e.g., comparing the values of the latent vectors $\boldsymbol{x}_n$ generated from different states provide a valid proximity criterion? In such a case, what type of proximity measure would be more effective?

The second open question we want to investigate is the possibility of obtaining stacked denoising autoencoders [32] for sequential data modeling using our LM^{2}GP model as the main building block. Stacked denoising autoencoders are deep learning architectures that apart from extracting the most informative latent subspace representations of observed data are also capable of denoising the observed data, which are considered contaminated with noise at each phase of the model training algorithm. GPLVM models have already been used as building blocks for obtaining stacked denoising autoencoder architectures with great success [33]. However, existing formulations are not designed for capturing and exploiting temporal dependencies in the modeled data. In this paper, we investigated the utility of LM^{2}GP as the main building block for obtaining simple deep learning architectures, and obtained some promising results. How would our model perform in the context of a stacked denoising autoencoder framework? This is a question that remains to be addressed in our future work.

We shall publish source codes pertaining to our method at: http://www.cut.ac.cy/eecei/staff/sotirios.chatzis/?languageId=2.

APPENDIX

In this Appendix, we provide (for completeness sake) the expressions of the derivatives of the variational free energy $\mathcal{L}(q)$ over the kernel hyperparameters, required for hyperparameter optimization by means of L-BFGS, as well as the expressions of the derivatives of $q(\boldsymbol{x}_n)$ w.r.t. the latent vectors $\boldsymbol{x}_n$, required for HMC sampling from $q(\boldsymbol{x}_n)$.

Regarding the derivatives of $\mathcal{L}(q)$ w.r.t. the kernel hyperparameters, say $\boldsymbol{\varphi}_d$, $d = 1, \dots, D$, we have

$$\begin{aligned}\frac{\partial \mathcal{L}(q)}{\partial \boldsymbol{\varphi}_d} = &-\frac{1}{2}\left(\hat{\boldsymbol{\mu}}_d\hat{\boldsymbol{\mu}}_d^T + \boldsymbol{\Sigma}_d\right)\left\langle \frac{\partial \boldsymbol{K}_d(X,X)^{-1}}{\partial \boldsymbol{\varphi}_d}\right\rangle_{q(X)} \\ &+ \frac{1}{2}\boldsymbol{K}_d(X,X)\left\langle \frac{\partial \boldsymbol{K}_d(X,X)^{-1}}{\partial \boldsymbol{\varphi}_d}\right\rangle_{q(X)}\end{aligned} \tag{73}$$

Similar, the derivatives of $q(\boldsymbol{x}_n)$ w.r.t. the latent vectors $\boldsymbol{x}_n$ yield

$$\begin{aligned}\frac{\partial \mathrm{log} q(\boldsymbol{x}_n)}{\partial \boldsymbol{x}_n} = &\sum_c q(z_{nc}=1)\left[\tilde{\omega}_c\tilde{\boldsymbol{\Phi}}_c\tilde{\boldsymbol{m}}_c - \boldsymbol{x}_n^T\tilde{\omega}_c\tilde{\boldsymbol{\Phi}}_c\right] \\ &- \frac{1}{2}\sum_d\left(\hat{\boldsymbol{\mu}}_d\hat{\boldsymbol{\mu}}_d^T + \boldsymbol{\Sigma}_d\right)\frac{\partial \boldsymbol{K}_d(X,X)^{-1}}{\partial \boldsymbol{x}_n} \\ &+ \frac{1}{2}\sum_d \boldsymbol{K}_d(X,X)\frac{\partial \boldsymbol{K}_d(X,X)^{-1}}{\partial \boldsymbol{x}_n}\end{aligned} \tag{74}$$

REFERENCES

[1] C. M. Bishop, *Neural Networks for Pattern Recognition*. Clarendon Press, 1995.

[2] J. M. Wang, D. J. Fleet, and A. Hertzmann, "Gaussian process dynamical models for human motion," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 30, no. 2, pp. 283–298, 2008.

[3] C. E. Rasmussen and C. K. I. Williams, *Gaussian Processes for Machine Learning*. MIT Press, 2006.

[4] N. Lawrence, "Probabilistic non-linear principal component analysis with Gaussian process latent variable models," *J. Machine Learning Research*, vol. 6, pp. 1783–1816, 2005.

[5] M. F. Moller, "A scaled conjugate gradient algorithm for fast supervised learning," *Neural Networks*, vol. 6, pp. 525–533, 1993.

[6] J. Sethuraman, "A constructive definition of the Dirichlet prior," *Statistica Sinica*, vol. 2, pp. 639–650, 1994.

[7] T. Ferguson, "A Bayesian analysis of some nonparametric problems," *The Annals of Statistics*, vol. 1, pp. 209–230, 1973.

[8] M. Jordan, Z. Ghahramani, T. Jaakkola, and L. Saul, "An introduction to variational methods for graphical models," in *Learning in Graphical Models*, M. Jordan, Ed. Dordrecht: Kluwer, 1998, pp. 105–162.

[9] D. M. Blei and M. I. Jordan, "Variational inference for Dirichlet process mixtures," *Bayesian Analysis*, vol. 1, no. 1, pp. 121–144, 2006.

[10] D. Blackwell and J. MacQueen, "Ferguson distributions via Pólya urn schemes," *The Annals of Statistics*, vol. 1, no. 2, pp. 353–355, 1973.

[11] M. Tipping and C. Bishop, "Mixtures of probabilistic principal component analyzers," *Neural Computation*, vol. 11, no. 2, pp. 443–482, 1999.

[12] J. Zhao and Q. Jiang, "Probabilistic PCA for t distributions," *Neurocomputing*, vol. 69, no. 16-18, pp. 2217–2226, Oct. 2006.

[13] J. Paisley and L. Carin, "Hidden Markov models with stick-breaking priors," *Signal Processing, IEEE Transactions on*, vol. 57, no. 10, pp. 3905–3917, 2009.

[14] Y. Qi, J. W. Paisley, and L. Carin, "Music analysis using hidden Markov mixture models," *IEEE Transactions on Signal Processing*, vol. 55, no. 11, pp. 5209–5224, 2007.

[15] D. Chandler, *Introduction to Modern Statistical Mechanics*. New York: Oxford University Press, 1987.

[16] D. Blei and M. Jordan, "Variational methods for the Dirichlet process," in *21st Int. Conf. Machine Learning*, New York, NY, USA, July 2004, pp. 12–19.

[17] P. Muller and F. Quintana, "Nonparametric Bayesian data analysis," *Statist. Sci.*, vol. 19, no. 1, pp. 95–110, 2004.

[18] S. P. Chatzis, D. I. Kosmopoulos, and T. A. Varvarigou, "Robust sequential data modeling using an outlier tolerant hidden Markov model," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 31, no. 9, pp. 1657–1669, 2009.

[19] L. Rabiner, "A tutorial on hidden Markov models and selected applications in speech recognition," *Proceedings of the IEEE*, vol. 77, pp. 245–255, 1989.

[20] R. M. Neal, "Probabilistic inference using Markov chain Monte Carlo methods," University of Toronto, Tech. Rep., 1993.

[21] D. Liu and J. Nocedal, "On the limited memory method for large scale optimization," *Mathematical Programming B*, vol. 45, no. 3, pp. 503–528, 1989.

[22] V. N. Vapnik, *Statistical Learning Theory*. New York: Wiley, 1998.

[23] R.-E. Fan, K.-W. Chang, C.-J. Hsieh, X.-R. Wang, and C.-J. Lin, "LIBLINEAR: A library for large linear classification," *Journal of Machine Learning Research*, vol. 9, pp. 1871–1874, 2008.

[24] A. Voulodimos, D. Kosmopoulos, G. Vasileiou, E. Sardis, V. Anagnostopoulos, C. Lalos, A. Doulamis, and T. Varvarigou, "A threefold dataset for activity and workflow recognition in complex industrial environments," *MultiMedia, IEEE*, vol. 19, pp. 42 –52, July-Sept. 2012.

[25] D. Kosmopoulos and S. Chatzis, "Robust visual behavior recognition," *Signal Processing Magazine, IEEE*, vol. 27, no. 5, pp. 34 –45, sept. 2010.

[26] S. M. Oh, J. M. Rehg, T. Balch, and F. Dellaert, "Learning and inferring motion patterns using parametric segmental switching linear dynamic systems," *International Journal of Computer Vision*, vol. 77, no. 1-3, pp. 103–124, 2008.

[27] E. B. Fox, E. B. Sudderth, M. I. Jordan, and A. S. Willsky, "Nonparametric Bayesian learning of switching linear dynamical systems," in *Proc. Neural Information Processing Systems*, 2009.

[28] Y. Altun, I. Tsochantaridis, and T. Hofmann, "Hidden Markov support vector machines," in *Proc. ICML*, 2004.

[29] M. A. Bartsch and G. H. Wakefield, "To catch a chorus: Using chroma-based representations for audio thumbnailing," in *IEEE Workshop on the Applications of Signal Processing to Audio and Acoustics*, 2001, pp. 15–18.

[30] H. Jensen, M. G. Christensen, D. Ellis, and S. H. Jensen, "A tempo-insensitive distance measure for cover song identification based on chroma features," in *Proc. ICASSP-08*, 2008, pp. 2209–2212.

[31] R. Mukundan and K. R. Ramakrishnan, *Moment Functions in Image Analysis: Theory and Applications*. World Scientific, 1998.

[32] P. Vincent, H. Larochelle, I. Lajoie, Y. Bengio, and P.-A. Manzagol, "Stacked denoising autoencoders: Learning useful representations in a deep network with a local denoising criterion," *Journal of Machine Learning Research*, vol. 11, pp. 3371–3408, 2010.

[33] J. Snoek, R. P. Adams, and H. Larochelle, "Nonparametric guidance of autoencoder representations using label information," *Journal of Machine Learning Research*, vol. 13, pp. 2567–2588, 2012.