Filtering of High-Dimensional Data for Sequential Classification

Marzieh Ajirak,† Yuhao Liu,◇ and Petar M. Djurić†
†Department of Electrical and Computer Engineering, Stony Brook University, NY, USA
◇Capital One, DC, USA

***Abstract*—In many science and engineering problems, we observe high-dimensional data acquired sequentially. At each time instant, these data correspond to one of a predefined number of classes. The sequence of classes follows a certain pattern, with the transition probabilities of the classes being unknown. Our hypothesized generative model of the observed data involves two latent processes. The first is a root process representing the sequence of classes, while the second is a low-dimensional process generated as a Markovian process, depending on the current class and the previous value of the low-dimensional process. The observed high-dimensional process is generated from the low-dimensional state process. Our objective is to infer the posterior distributions of the classes as they evolve over time based on the observed data and the adopted model. To achieve this, we propose a method for estimating the latent processes. We demonstrate the effectiveness of our approach on synthesized data.**

***Index Terms*—deep state-space models, Gaussian processes, discrete latent processes, particle filtering, preferential attachment prior**

I. Introduction

In science and engineering, high-dimensional time series are commonly modeled using state-space models, assuming the state processes are low-dimensional. Furthermore, the dynamics of the state process are influenced by an unobserved discrete latent process that corresponds to different classes of the state process. It is often of interest to find the sequence of classes, i.e., the discrete latent process, from the observed time series. Such models are valuable for capturing sudden shifts in system behavior that occur randomly over time.

One approach to estimating the dynamics of discrete latent variables that correspond to different classes is by using multiple switching models. For these types of problems, one can leverage methods from signal processing, such as those based on Markovian switching systems, also known as jump Markov systems [6], [7], [27].

An important domain where these models are applied is neuroscience [1], [28]. There, researchers often analyze high-dimensional time series and seek inference methods that can effectively address the problem of identifying changes in behavioral patterns and/or neural dynamics [2]. The focal point revolves around modeling the spatiotemporal dynamics of neural population activity while facilitating flexible inference. This is frequently achieved by incorporating lower-dimensional nonlinear latent factors and structures [1], [25].

This is close to regime switching which is an important area of work in econometrics [15]. In these models the statistical properties of time series data, such as mean, variance, or autocorrelation structure, change over time according to different regimes or states. These regimes could represent different economic conditions, market environments, policy regimes, or other underlying factors that influence the behavior of the time series. The used models should capture nonlinearities, structural breaks, and time-varying dynamics in economic and financial time series data. They are employed to model stock returns, interest rates, exchange rates, economic growth, and other macroeconomic variables [11], [13], [26].

In our paper, we consider regimes that represent *ordinal* classes [1], where the sequence of classes follow an unknown pattern. For clarity, ordinal classes refer to categories or groups that possess a natural order or ranking. Thus, in our paper, when discussing switching between classes, such transitions refer specifically to moving to a 'neighboring' class only. Ordinal classes are of importance in machine learning because of many applications where outcomes are described by ordered categories (e.g., in healthcare [3], natural language processing [18], and economics [14]). In the remainder of the paper, we will mostly omit the adjective "ordinal" before "classes," although our discussion exclusively concerns ordinal classes.

We adopt a generative model of the data, where classes representing a system form a sequence following the Yule-Simon process law once the system enters a particular regime (class). When the system decides to transition to a new class, the next class is selected following a model similar to the Polya-urn model. Specifically, the probability of selecting the new regime is proportional to the total time the system has historically spent in that regime. Given the selected class at time t, the system generates a low-dimensional latent process $\mathbf{x}_t$ that depends on the class. The process value $\mathbf{x}_t$ also depends on its previous value, $\mathbf{x}_{t-1}$. Once $\mathbf{x}_t$ is generated, a high-dimensional vector $\mathbf{y}_t$ is obtained from $\mathbf{x}_t$. The function that maps $\mathbf{x}_t$ onto $\mathbf{y}_t$ is independent of the system's regime. In our study, we assume a lack of knowledge regarding the functions generating the latent and observed processes, as well as the parameters of the model governing the class sequence.

Our inference of the unknowns is formulated within the Bayesian framework. To estimate the unknown functions, we use an approximation of Gaussian processes based on

This work was supported by the National Science Foundation under Award 2212506.

[1]In the remainder of the paper, we interchangeably use the terms "class," "regime," or "discrete latent variable".

random features. This model includes four sets of unknowns: 1) parameters associated with the class generation model, 2) linear parameters of the Gaussian process approximations, 3) unknown variances of the noises, and 4) the state and the discrete latent processes. We assume knowledge of the dimension of the state process, the number of classes, and the availability of training data for each class. (The relaxation of these assumptions is left for future work.) The proposed solution is based on integrating out the linear parameters of the models and the noise variances, and on particle filters (PFs) for estimating the state process $\mathbf{x}_t$.

We note that PFs have received significant attention in the literature regarding the inference of regime switching. Some recent contributions include [12], [20]. In [12], a PF algorithm for general regime-switching systems was introduced, which incorporates the model index as an unknown variable in the system. The model index is jointly estimated with the time-varying parameters of the system. Unlike existing approaches, the algorithm allows for a diverse set of candidate models by appropriately selecting the model index proposal distribution. In [20], the authors propose a new differentiable PF for regime-switching state-space models, where a set of unknown candidate dynamic and measurement models is learned and tracked. Other literature that addresses this problem with PFs includes [4], [9], [23], [24], [29].

The contributions of our paper include the following: we propose a fully Bayesian solution for the sequential classification of high-dimensional time series. Our approach involves estimating unknown functions in the state equation under model uncertainty and unknown functions in the observation equation. The assumptions about the generative model of the data are minimal.

The rest of the paper is organized as follows: In the next section, we present the generative model of our data and the problem we aim to solve. Section III elaborates on the proposed solution. We present numerical results demonstrating the performance of the proposed method in Section IV. Finally, Section V offers concluding remarks.

II. The generative model and the problem formulation

Let z_t be a latent process representing a sequence of classes, where $z_t \in \mathcal{Z} = \{c_1, c_2, c_3, ..., c_K\}$, where K represents the number of different classes. The classes are of ordinal nature, e.g., c_1 is the "worst" class, c_2 is better than c_1, and so on. The class z_0 is drawn from a prior probability mass function, $p(z)$, i.e.,

$$z_0 \sim p(z_0), \tag{1}$$

where $p(z_0)$ is known. When z_t equals c_k, in the subsequent time step, it may either retain the same class or transition to an adjacent class, namely c_{k-1} or c_{k+1}, where k ranges from 2 to $K-1$. In the special cases where z_t is equal to c_1, z_{t+1} is restricted to either c_1 or c_2. Similarly, if z_t is equal to c_K, its value at z_{t+1} can only be either c_{K-1} or c_K.

Let $z_t = c_k$. The probability that the class process does not change its value at $t+1$, is given by

$$p(z_{t+1} = z_t) = \frac{n_k}{\rho_k + n_k}, \tag{2}$$

where n_k is the number of consecutive instants of the process holding class c_k since the last change, and $\rho_k > 0$ is a parameter of the class c_k. The probability that the process changes the class at $t+1$ is

$$p(z_{t+1} \neq z_t) = \frac{\rho_k}{\rho_k + n_k}. \tag{3}$$

The duration of the class process keeping the same value follows a Yule-Simon distribution.

Let $z_t = c_k$, $k = 2, 3, \ldots, K-1$. After the system "decides" to change its value c_k at $t+1$, it must make another decision: selecting the next class. This selection is done according to

$$z_{t+1} = \begin{cases} c_{k-1}, & \text{with } P = \frac{\rho_{k-1}+N_{t,k-1}}{\rho_{k-1}+N_{t,k-1}+\rho_{k+1}N_{t,k+1}} \\ c_{k+1}, & \text{with } P = \frac{\rho_{k+1}+N_{t,k+1}}{\rho_{k+1}+N_{t,k-1}+\rho_{k+1}N_{t,k+1}} \end{cases}, \tag{4}$$

where $N_{t,k-1}$ and $N_{t,k+1}$ are the total numbers of samples of the class process with values c_{k-1} and c_{k+1} up to time instant t, respectively.

Given the drawn class z_t, we generate a low-dimensional vector process $\mathbf{x}_t$ according to

$$\mathbf{x}_t = f_{z_t}(\mathbf{x}_{t-1}) + \boldsymbol{\eta}_{z_t,t}, \tag{5}$$

where $\mathbf{x}_t \in \mathbb{R}^{d_x}$; $f_{z_t}(\cdot) : \mathbb{R}^{d_x} \to \mathbb{R}^{d_x}$ is a *class-dependent* function that maps the input vector $\mathbf{x}_{t-1}$ to a vector of the same size; the symbol $\boldsymbol{\eta}_{z_t,t} \in \mathbb{R}^{d_x}$ is a zero-mean Gaussian perturbation, or more specifically $\boldsymbol{\eta}_{z_t,t} \overset{i.i.d.}{\sim} \mathcal{N}(\mathbf{0}, \boldsymbol{\Sigma}_{\eta_{z_t}})$, where $\boldsymbol{\Sigma}_{\eta_{z_t}}$ is a diagonal matrix with corresponding variances along its diagonal. Once $\mathbf{x}_t$ is obtained, we generate another vector $\mathbf{y}_t$ by

$$\mathbf{y}_t = h(\mathbf{x}_t) + \boldsymbol{\nu}_t, \tag{6}$$

where $\mathbf{y}_t \in \mathbb{R}^{d_y}$, and $d_y >> d_x$; The function $h(\cdot) : \mathbb{R}^{d_x} \to \mathbb{R}^{d_y}$ is shared by all the classes, and it takes the vector $\mathbf{x}_t$ as input and produces a vector of much larger size; the vector $\boldsymbol{\nu}_t$ stands for noise, and $\boldsymbol{\nu}_t \overset{i.i.d.}{\sim} \mathcal{N}(\mathbf{0}, \boldsymbol{\Sigma}_\nu)$, where $\boldsymbol{\Sigma}_\nu$ is a diagonal matrix with corresponding variances along its diagonal.

Given the above model, the objective is to estimate the sequence of classes z_t from the observed sequence of vectors $\mathbf{y}_t$. The functions $f_k(\cdot), \forall k$ and $h(\cdot)$, along with the variances $\sigma^2_{\eta,k}$ and σ^2_ν, are unknown. We assume that we have training data from each class for learning the functions $f_k(\cdot)$ and $h(\cdot)$.

III. The proposed solution for sequential classification

In [22] and [21], the challenge of estimating latent processes is addressed as defined in the preceding section, specifically within a *single-class* context. This present work aims to extend the methodology introduced in [22] and [21] to accommodate multi-class scenarios. We will outline the approach for modeling and estimation of the unknown functions $f_k(\cdot)$ and $h(\cdot)$, describe the training process, and present our solution for multi-class applications.

A. Modeling the functions

We employ Gaussian processes (GPs) [30] to model the functions specified in (5) and (6). In this context, an unknown function f is conceptualized as a stochastic entity, sampled from a GP denoted as $f \sim \mathcal{GP}(m, \kappa)$, where m represents the mean function, and κ denotes the kernel function. Consequently, the function values $\mathbf{f}$ exhibit Gaussian distribution, expressed as:

$$p(\mathbf{f}|X) = \mathcal{N}(\mathbf{f}|\mathbf{m}, \mathbf{K}), \tag{7}$$

where the size of $\mathbf{f}$ aligns with the dimensions of the input-output data used for function learning. The covariance matrix $\mathbf{K}$ is formulated from the inputs of the function.

Scaling up GPs with a large number of input-output pairs poses challenges, particularly when inverting the covariance matrix $\mathbf{K}$ becomes computationally expensive. To address this, we turn to approximations leveraging the concept of sparsity. One common approach to address this approximation is by constructing GPs with features derived from a feature space [16].

In particular, GPs employing a shift-invariant kernel can undergo approximation within a feature space [16]. This approach allows computations to bypass matrix decompositions, relying solely on matrix multiplications. We explain this on the function $h(\cdot)$ from (6). This is a vector function with d_y outputs, and we model it by

$$h(\mathbf{x}_t) = \boldsymbol{\Theta}^\top \boldsymbol{\phi}(\mathbf{x}_t) \tag{8}$$

where $h(\cdot) : \mathbb{R}^{d_x} \to \mathbb{R}^{d_y}$, $\boldsymbol{\Theta} \in \mathbb{R}^{2J \times d_y}$, and $\boldsymbol{\phi} \in \mathbb{R}^{2J \times 1}$ is a feature vector defined by [16]

$$\boldsymbol{\phi}(\mathbf{x}_t) = \frac{1}{\sqrt{J}}[\sin(\mathbf{x}_t^\top \boldsymbol{\omega}^1), \cos(\mathbf{x}_t \boldsymbol{\omega}^1), ..., \sin(\mathbf{x}_t \boldsymbol{\omega}^J), \cos(\mathbf{x}_t \boldsymbol{\omega}^J)]^\top, \tag{9}$$

with $\boldsymbol{\Omega} = \{\boldsymbol{\omega}^1, \boldsymbol{\omega}^2, \ldots, \boldsymbol{\omega}^J\}$ representing a collection of samples randomly extracted from the power spectral density of the GP kernel.

We have a similar approximation of all the class functions in (5), i.e.,

$$f_k(\mathbf{x}_{t-1}) = \boldsymbol{\Psi}_k^\top \boldsymbol{\varphi}(\mathbf{x}_{t-1}), \tag{10}$$

where $f_k(\mathbf{x}_{t-1}) : \mathbb{R}^{d_x} \to \mathbb{R}^{d_x}$, $\forall k$, $\boldsymbol{\Psi} \in \mathbb{R}^{2J \times d_x}$, and $\boldsymbol{\varphi}(\mathbf{x}_{t-1}) \in \mathbb{R}^{2J \times 1}$ is defined like $\boldsymbol{\phi}(\cdot)$ in (9).

In summary, we work with the following models of $\mathbf{x}_t$ and $\mathbf{y}_t$:

$$\mathbf{x}_t = \boldsymbol{\Psi}_{z_t}^\top \boldsymbol{\varphi}(\mathbf{x}_{t-1}) + \boldsymbol{\eta}_{z_t,t}, \tag{11}$$

$$\mathbf{y}_t = \boldsymbol{\Theta}^\top \boldsymbol{\phi}(\mathbf{x}_t) + \boldsymbol{\epsilon}_t. \tag{12}$$

Clearly, the number of parameters in (11) and (12) that need to be estimated is $d_x \times K \times 2J + d_y \times 2J$ plus $d_x K$ noise variances from (11) and d_y variances from (12).

B. Training the models

Our model involves several types of unknowns, including the sequence of classes, z_t, their respective parameters ρ_k for $k = 1, 2, \ldots, K$, the latent process $\mathbf{x}_t$, the parameters $\boldsymbol{\Theta}$ and $\boldsymbol{\Psi}_k$ for $k = 1, 2, \ldots, K$, the noise covariance matrix $\boldsymbol{\Sigma}_{\eta_k}$ for $k = 1, 2, \ldots, K$, and the covariance matrix $\boldsymbol{\Sigma}_\nu$. Our approach is Bayesian, and during the training phase, we obtain the joint posterior distributions of $\boldsymbol{\Psi}_k$ and $\boldsymbol{\Sigma}_{\eta_k}$, denoted as $p(\boldsymbol{\Psi}_k, \boldsymbol{\Sigma}_{\eta_k}|\mathcal{D}_k)$ for all classes k and $p(\boldsymbol{\Theta}, \boldsymbol{\Sigma}_\epsilon|\mathcal{D})$. Here, $\mathcal{D}_k$ represents the training data belonging to class k and $\mathcal{D}$ is the set of all training data. The posteriors derived from the training data serve as the initial priors for the subsequent filtering operation.

We estimate the unknowns of the model given by (11) and (12) via particle filtering and using analytical expressions. Particle filtering is employed to track the state process $\mathbf{x}_t$, and given $\mathbf{x}_t$, we update the posterior of $\boldsymbol{\Psi}_k$ and $\boldsymbol{\Sigma}_{\eta_k}$ as well as the posterior of $\boldsymbol{\Theta}$ and $\boldsymbol{\Sigma}_\epsilon$. The particle filter generates many possible trajectories of the state process, and we denote them by $\mathbf{x}_t^{(m)}$, $m = 1, 2, \ldots, M$. We note that each trajectory has its own joint posterior of the parameters. We update the parameters of the posteriors and estimate the state process, as described next.

a) Updating the posteriors of $(\boldsymbol{\Psi}_k, \boldsymbol{\Sigma}_{\eta_k})$ and $(\boldsymbol{\Theta}, \boldsymbol{\Sigma}_\epsilon)$: We explain the update of the posterior of $\boldsymbol{\theta}_i \in \mathbb{R}^{2J \times 1}$ (the $i-$th column of $\boldsymbol{\Theta}$) and $[\boldsymbol{\Sigma}_\epsilon]_{ii} = \sigma_{i\epsilon}^2$. We start with the prior

$$p(\boldsymbol{\theta}_i, \sigma_{i\epsilon}^2) \propto \frac{1}{\sigma_{i\epsilon}^{a_0+1}} e^{-\frac{1}{2\sigma_{i\epsilon}^2}\left(b_{i0} + (\boldsymbol{\theta}_i - \boldsymbol{\theta}_{i0})^\top \boldsymbol{\Sigma}_{i0}^{-1} (\boldsymbol{\theta} - \boldsymbol{\theta}_{i0})\right)}, \tag{13}$$

where a_{i0}, b_{i0}, $\boldsymbol{\theta}_{i0}$, and $\boldsymbol{\Sigma}_{i0}$ are parameters of the prior, and where $a_{i0} > 2J$ and $b_{i0} > 0$. In (13), we recognize the multivariate normal–inverted Gamma distribution. At time instant t this distribution is still normal–inverted Gamma and with parameters $a_{it}, b_{it}^{(m)}, \boldsymbol{\theta}_{it}^{(m)^\top}$, and $\boldsymbol{\Sigma}_{it}^{(m)}$. These parameters are recursively obtained by

$$a_{it} = a_{i,t-1} + 1, \tag{14}$$

$$\begin{aligned} b_{it}^{(m)} &= b_{i,t-1}^{(m)} + y_{it}^2 \\ &\quad + \boldsymbol{\theta}_{i,t-1}^{(m)^\top} \boldsymbol{\Upsilon}_{i,t-1}^{(m)^{-1}} \boldsymbol{\theta}_{i,t-1}^{(m)} \\ &\quad - \boldsymbol{\theta}_{it}^{(m)^\top} \boldsymbol{\Upsilon}_{i,t}^{(m)^{-1}} \boldsymbol{\theta}_{it}^{(m)}, \end{aligned} \tag{15}$$

$$\boldsymbol{\theta}_{it}^{(m)} = \boldsymbol{\Upsilon}_{it}^{(m)} \left(\boldsymbol{\Upsilon}_{i,t-1}^{(m)^{-1}} \boldsymbol{\theta}_{i,t-1}^{(m)} + \boldsymbol{\phi}_t^{y^{(m)}} y_{it} \right), \tag{16}$$

$$\boldsymbol{\Upsilon}_{it}^{(m)} = \left(\boldsymbol{\Upsilon}_{i,t-1}^{(m)^{-1}} + \boldsymbol{\phi}_t^{(m)} \boldsymbol{\phi}_t^{(m)^\top} \right)^{-1}, \tag{17}$$

where the superscript $^{(m)}$ suggests that all the variables are associated with the mth particle stream of the state process. The same equations hold for all $\boldsymbol{\theta}_i$ and $\sigma_{i\epsilon}^2$, $i = 1, 2, \ldots, d_y$. Analogous equations also hold for $\boldsymbol{\psi}_{ik}^\top$ and $\sigma_{ik\eta}^2$, $i = 1, 2, \ldots, d_x, k = 1, 2, \ldots, K$.

b) Estimating the latent process $\mathbf{x}_t$: We employ particle filtering for estimating $\mathbf{x}_t$ [5], [8], [10]. In summary, particle filters approximate probability density functions (pdfs) using discrete random measures. The pdf's support is defined by a set of particles, where each particle is assigned a weight. For example, at time $t-1$, the posterior pdf $p(\mathbf{x}_{t-1}|\mathbf{y}_{1:t-1})$ is approximated by

$$p^M(\mathbf{x}_{t-1}|\mathbf{y}_{1:t-1}) = \frac{1}{M} \sum_{m=1}^{M} \delta(\mathbf{x}_{t-1} - \mathbf{x}_{t-1}^{(m)}), \tag{18}$$

where $\mathbf{x}_{t-1}^{(m)}$ denotes the $m-$th particle (sample) of $\mathbf{x}_{t-1}$, $\delta(\cdot)$ is the Dirac delta function, and M is the number of particles. The approximating random measure $p^M(\mathbf{x}_t|\mathbf{y}_{1:t})$ can be obtained from $p^M(\mathbf{x}_{t-1}|\mathbf{y}_{1:t-1})$ by

(a) Generating particles $\mathbf{x}_t^{(m)}$ according to

$$\mathbf{x}_t^{(m)} \sim p(\mathbf{x}_t|\mathbf{x}_{t-1}^{(m)}). \tag{19}$$

(b) Computing the weights of the particles $\mathbf{x}_t^{(m)}$ by

$$w_t^{(m)} \propto p(\mathbf{y}_t|\mathbf{x}_t^{(m)}), \tag{20}$$

where

$$\sum_{m=1}^{M} w_t^{(m)} = 1. \tag{21}$$

At this point, $p(\mathbf{x}_t|\mathbf{y}_{1:t})$ is approximated by

$$p^M(\mathbf{x}_t|y_{1:t}) = \sum_{m=1}^{M} w_t^{(m)} \delta(\mathbf{x}_t - \mathbf{x}_t^{(m)}). \tag{22}$$

(c) Resampling the particles using their weights $w_t^{(m)}$ [19].

In our work, the densities for sampling new particles are multivariate Student's t pdfs, i.e.,

$$p(\mathbf{x}_t|\mathbf{x}_{t-1}^{(m)}) = t_{\nu_t}\left(\boldsymbol{\mu}_t^{(m)}, \boldsymbol{\Upsilon}_t^{(m)}\right), \tag{23}$$

where $\boldsymbol{\mu}_t^{(m)}$ is the location vector of the pdf that belongs to the $m-$th stream, $\boldsymbol{\Upsilon}_t^{(m)}$ is its scale matrix, and ν_t represents the degrees of freedom. They are both updated after every time instant t (for details, see [22]).

We compute the weights according to

$$w_t^{(m)} \propto p\left(\mathbf{y}_t|\mathbf{x}_{1:t}^{(m)}, \mathbf{y}_{1:t-1}\right), \tag{24}$$

where the pdf of $\mathbf{y}_t$ in (25) is also a multivariate Student's t pdf.

At the end of the training process, the posteriors of interest are $p(\boldsymbol{\Psi}_k, \boldsymbol{\Sigma}_{\eta_k}|\mathcal{D}_k)$, for $k = 1, 2, \ldots, K$ and $p(\boldsymbol{\Theta}, \boldsymbol{\Sigma}_\epsilon|\mathcal{D})$. We form these posteriors from $p^{(m)}(\boldsymbol{\Psi}_k, \boldsymbol{\Sigma}_{\eta_k}|\mathcal{D}_k)$, for $k = 1, 2, \ldots, K$ and $p^{(M(k-1)+m)}(\boldsymbol{\Theta}, \boldsymbol{\Sigma}_\epsilon|\mathcal{D})$, where $m = 1, 2, \ldots, M$, and $k = 1, 2, \ldots, K$. There are many ways to form the posterior from all these posteriors. In our work, we used the mean square error estimates of the parameters of the respective normal–inverted Gamma distributions.

C. Sequential estimation of the classes

Once the training is completed, the sequential estimation of the classes proceeds as follows. We are interested in finding $P(z_t|\mathbf{y}_{1:t}, z_{1:t-1})$, where $z_1, z_2, \ldots, z_t$ are the estimates of the classes. Recall that our unknowns are $\rho_k, \boldsymbol{\Theta}_k, \sigma_{ik\eta}^2, i = 1, 2, \ldots d_x, k = 1, 2, \ldots, K; \boldsymbol{\Psi}$, and $\sigma_{i\epsilon}^2$, for $i = 1, 2, \ldots d_y$. Since we work with particle streams, suppose that at time $t-1$ we have the M posterior distributions of all the unknowns, which are used for generating particles, i.e., $p(\rho_k|z_{1:t-1}^{(m)})$, $\forall k$, $p(z_t|z_{1:t-1}^{(m)}, \boldsymbol{\rho}_t^{(m)})$, and $p(\mathbf{x}_t|\mathbf{x}_{1:t-1}^{(m)}, z_t^{(m)})$. Note that the vector $\boldsymbol{\rho}_t^{(m)}$ is defined by $\boldsymbol{\rho}_t^{(m)} = [\rho_{1t}^{(m)}\ \rho_{2t}^{(m)} \ldots \rho_{Kt}^{(m)}]^\top$, The particle generation and their weight computation at time t is implemented as follows:

1) Sample the values of ρ_k from

$$\rho_{kt}^{(m)} \sim p(\rho_k|z_{1:t-1}^{(m)}), \quad \text{for } k = 1, 2, \ldots, K, \tag{25}$$

where

$$p(\rho_k|z_{1:t-1}^{(m)}) \propto p(z_{1:t-1}^{(m)}|\rho_k)p(\rho_k), \tag{26}$$

where $p(z_{1:t-1}^{(m)}|\rho_k)$ is the likelihood of ρ_k and where the subscript t in $\rho_{kt}^{(m)}$ suggests that the particle of ρ_k was drawn from the posterior given by (26) at time instant $t-1$. One can sample from this posterior according to a scheme based on a Gibbs sampler [17]. The priors of all the ρ_ks are all Gamma pdfs with parameters a_ρ, b_ρ.

2) Given the generated particles $\rho_{kt}^{(m)}$, we draw the particles of z_t, i.e.,

$$z_t^{(m)} \sim P^{(m)}(z_t|z_{1:t-1}^{(m)}, \boldsymbol{\rho}_t^{(m)}), \tag{27}$$

where

$$P^{(m)}\left(z_t|z_{1:t-1}^{(m)}, \boldsymbol{\rho}_t^{(m)}\right) = \begin{cases} \frac{n_{z_{t-1}^{(m)}}}{\rho_{z_t^{(m)}} + n_{z_{t-1}^{(m)}}} & , z_t = z_{t-1}^{(m)} \\ \widetilde{P}_{z_t}^{(m)}, & z_t \neq z_{t-1}^{(m)} \end{cases}, \tag{28}$$

where $n_{z_{t-1}^{(m)}}$ is the number of *consecutive* samples with the same label as that of $z_{t-1}^{(m)}$ and where

$$\widetilde{P}_{z_t}^{(m)} = \frac{\rho_{z_t}^{(m)}}{\rho_{z_{t-1}^{(m)}} + n_{z_{t-1}^{(m)}}} \times \frac{\rho_{z_t}^{(m)} + N_{z_t}^{(m)}}{(\rho_{z_{t-1}^{(m)}-1} + N_{z_{t-1}^{(m)}-1}^{(m)}) + (\rho_{z_{t-1}^{(m)}+1}^{(m)} + N_{z_{t-1}^{(m)}+1}^{(m)})}, \tag{29}$$

where z_t equals $z_{t-1}^{(m)}+1$ or $z_{t-1}^{(m)}-1$, provided $z_{t-1}^{(m)} \neq K$ or $z_{t-1}^{(m)} \neq 1$, and $N_k^{(m)}$ is the total number of samples the stream $z_{1:t-1}^{(m)}$ had a value equal to k. If $z_{t-1}^{(m)} = 1$ or $z_{t-1}^{(m)} = K$, the probability of leaving the current regime to the neighboring regime is given by

$$\widetilde{P}_{z_t}^{(m)} = \frac{\rho_{z_t}^{(m)}}{\rho_{z_t^{(m)}} + n_{z_{t-1}^{(m)}}}. \tag{30}$$

3) Next we need to sample $\mathbf{x}_t$. If the sample of the latent process comes from the class $z_t^{(m)}$, we use (see (23)), i.e.,

$$\mathbf{x}_t^{(m)} \sim t_{\nu_t}\left(\boldsymbol{\mu}_{z_t^{(m)},t}^{(m)}, \boldsymbol{\Upsilon}_{z_t^{(m)},t}^{(m)}\right). \tag{31}$$

We observe that (23) is obtained by integrating out $\boldsymbol{\Theta}_k^{(m)}$ and $\boldsymbol{\Sigma}_{\eta_k}^{(m)}$.

4) We compute the weights of the particles ($\mathbf{x}_t^{(m)}$, $z_t^{(m)}$, $\boldsymbol{\rho}_t^{(m)}$) by (24). Namely, the weights are computed according to

$$\begin{aligned} w_t^{(m)} &\propto p(\mathbf{y}_t|\mathbf{y}_{1:t-1}, \mathbf{x}_{1:t}^{(m)}) \\ &\times \frac{p(\mathbf{x}_t^{(m)}|\mathbf{x}_{1:t-1}^{(m)}, z_{1:t}^{(m)})}{q(\mathbf{x}_t^{(m)}|\mathbf{x}_{1:t-1}^{(m)}, z_{1:t}^{(m)})} \\ &\times \frac{p(z_t^{(m)}|z_{1:t-1}^{(m)}, \boldsymbol{\rho}_{1:t}^{(m)})}{q(z_t^{(m)}|z_{1:t-1}^{(m)}, \boldsymbol{\rho}_{1:t}^{(m)})} \\ &\times \frac{p(\boldsymbol{\rho}_t^{(m)}|\boldsymbol{\rho}_{1:t-1}^{(m)})}{q(\boldsymbol{\rho}_t^{(m)}|\boldsymbol{\rho}_{1:t-1}^{(m)})}, \end{aligned} \tag{32}$$

where with $q(\cdot)$ we denote the respective proposal distributions. In our implementation, we chose to set them equal to their corresponding pdfs $p(\cdot)$ (e.g., $q(\boldsymbol{\rho}_t^{(m)}|\boldsymbol{\rho}_{1:t-1}^{(m)}) = p(\boldsymbol{\rho}_t^{(m)}|\boldsymbol{\rho}_{1:t-1}^{(m)})$), which consequently involves using the following equation to compute the weights:

$$w_t^{(m)} \propto p(\mathbf{y}_t|\mathbf{y}_{1:t-1}, \mathbf{x}_{1:t}^{(m)}). \tag{33}$$

After we complete this cycle of steps, we estimate the probabilities of the classes at t. We use

$$\widehat{P}(z_t = k) = \sum_{m=1}^{M} w_t^{(m)} \mathbb{I}_{z_t^{(m)}=k}, \quad \forall k, \tag{34}$$

where $\mathbb{I}_{z_t^{(m)}=k}$ is the indicator function equal to one when the condition $z_t^{(m)} = k$ is satisfied.

Before we start the computations for the next time instant t, we resample the particles [19].

IV. Numerical results

A. Experiment 1

In Experiment 1, we generated two-dimensional hidden processes, where the functions associated with each class were defined by the following equations. If $z_t = c_1$, the latent process $\mathbf{x}_t$ evolved according to the following equations:

$$x_t^{[1]} = 0.9x_{t-1}^{[1]} + 0.5\sin(x_{t-1}^{[2]}) + u_t^{[1]}, \tag{35}$$

$$x_t^{[2]} = 0.5\cos(x_{t-1}^{[1]}) + 0.9x_{t-1}^{[2]} + u_t^{[2]}. \tag{36}$$

If $z_t = c_2$, we had

$$x_t^{[1]} = 0.9x_{t-1}^{[1]} - 0.5\sin(50x_{t-1}^{[1]}x_{t-1}^{[2]}) + u_t^{[1]}, \tag{37}$$

$$x_t^{[2]} = 0.5\cos(50x_{t-1}^{[1]}x_{t-1}^{[2]}) - 0.9x_{t-1}^{[2]} + u_t^{[2]}, \tag{38}$$

and if $z_t = c_3$,

$$x_t^{[1]} = 0.9x_{t-1}^{[1]} + 0.5\sin(3x_{t-1}^{[2]}) + u_t^{[1]}, \tag{39}$$

$$x_t^{[2]} = 0.5\cos(4x_{t-1}^{[1]} + 1) + 0.9x_{t-1}^{[2]} + u_t^{[2]}. \tag{40}$$

The observation process was defined by

$$\mathbf{y}_t = 0.5\mathbf{A}\mathbf{x}_t + \sin(\mathbf{B}\mathbf{x}_t) + \mathbf{v}_t, \tag{41}$$

where $\mathbf{A} \in \mathbb{R}^{10\times 2}$ is a matrix whose elements were randomly sampled from a standard Gaussian distribution $\mathcal{N}(0, 1)$, and the elements of $\mathbf{B} \in \mathbb{R}^{10\times 2}$ were independently sampled from a $Beta(1, 1)$ distribution. Thus, $d_x = 2, d_y = 10$.

We created three data sets from the respective classes c_1, c_2, and c_3. Each of them consisted of 10K samples, which were used for learning the different functions f_{z_t} in the state equation. By contrast, all the data sets were used for learning the function $h(\cdot)$ of the observation equation. The number of sampled frequencies of the Gaussian process in all cases was $J = 50$.

Once the training was completed, we tested the method on a new set of data that was 5k samples long. We did not use the model for generating the sequence of classes, and instead chose that each new class was 1k samples long and that the sequence of classes was $1 \to 2 \to 3 \to 2 \to 1$.

The results are shown in Fig. 1. The top plot presents the estimated probabilities of the three classes. The middle and bottom plots depict the two-dimensional state process and the 10-dimensional observed time series, respectively. The results clearly demonstrate that the method was capable of estimating the correct classes.

In Fig. 2, on the left, we see the estimated class in blue, expressed as real numbers (where the classes are enumerated as 0, 1, and 2) along with the actual class. In the middle plot, we display the true class and the decided class based on the values of the estimated classes. We observe excellent agreement between the true and estimated classes. On the right plot, we display the actual estimated probabilities of the classes.

B. Experiment 2

We repeated Experiment 1 with the same sequence of classes and state processes but changed the number of observations to $d_y = 15$. The results are shown in Fig. 3. As expected, the accuracy of the estimated classes somewhat increased. We note, however, that in this experiment, the number of unknown parameters went up to 1,500, from 1,000 in Experiment 1.

V. Conclusions

In this paper we addressed the problem of sequential classification of high-dimensional time series. At each time instant, the observed vector corresponds to one of a predefined number of classes. Our hypothesized generative model of the observed data involves two latent processes, of which the first represents the sequence of classes, while the second is a low-dimensional state process generated as a Markovian process. The observed high-dimensional process is generated from the low-dimensional state process. All the functions in the state and observation equations were assumed to be unknown. We proposed a method based on Gaussian processes that requires the estimation of both linear and nonlinear unknowns. The linear unknowns were integrated out, and the nonlinear ones were estimated using particle filtering. The method is fully Bayesian and produces, as output, the estimated probabilities of the sequence of classes to which the observed vectors belong. The method has been tested on simulated data, and the results suggest that it has the potential to estimate the classes

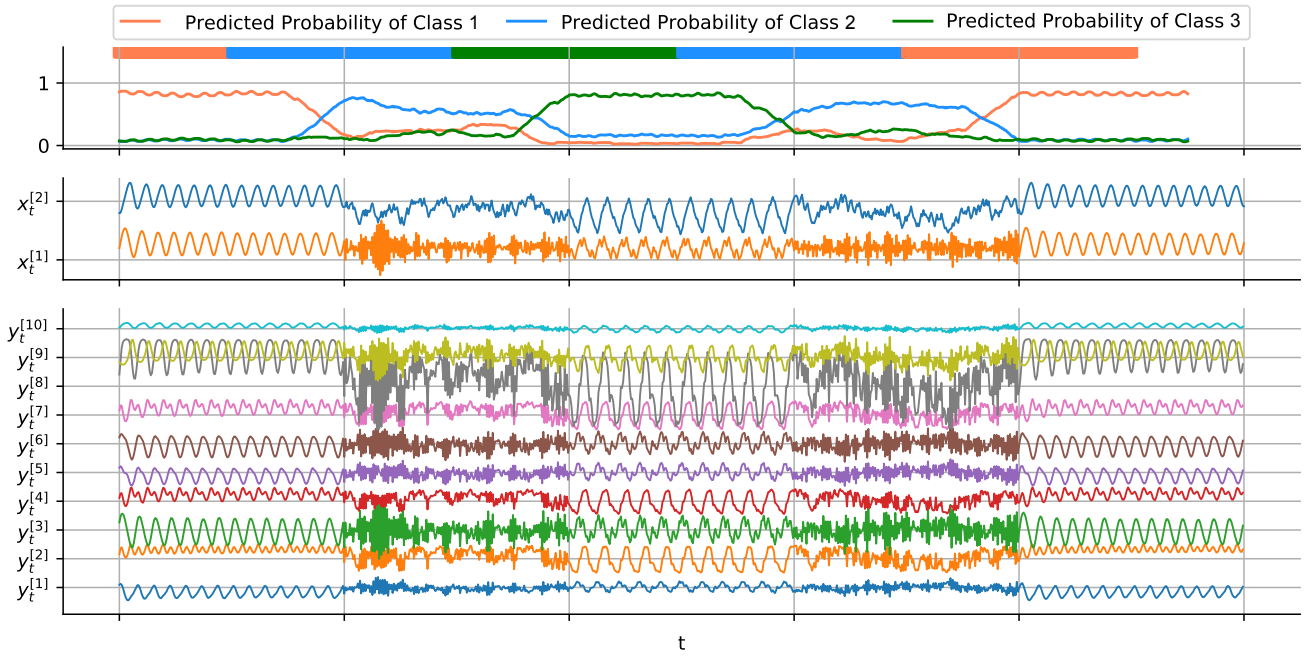
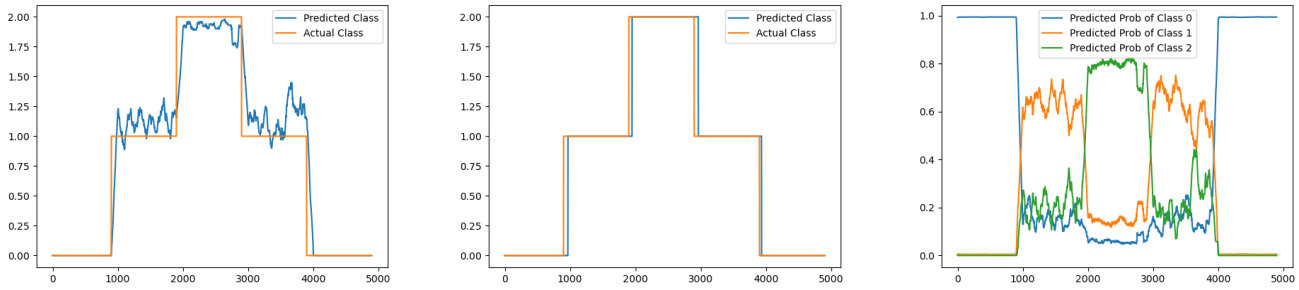


Fig. 1: Top: The estimated probabilities of the classes; Middle: The generated two-dimensional state process; Bottom: The generated 10-dimensional observed time series.



(a) (b) (c)

Fig. 2: (a) The estimated class as a real number; (b) The decided class based on the estimated probability; (c) The normalized probabilities of the classes. (On this plot the classes are denoted as Class 0, 1, and 2.)

of high-dimensional multivariate time series with minimal assumptions about the generative model of the data.

References

[1] H. Abbaspourazad, E. Erturk, B. Pesaran, and M. M. Shanechi. Dynamical flexible inference of nonlinear latent factors and structures in neural population activity. *Nature Biomedical Engineering*, 8(1):85–108, 2024.

[2] M. Ajirak, I. Elbau, N. Solomonov, and L. Grosenick. Discrete representation learning for multivariate time series. In *European Signal Processing Conference*, 2024.

[3] M. Ajirak, C. Heiselman, J. G. Quirk, and P. M. Djurić. Boost ensemble learning for classification of CTG signals. In *ICASSP 2022 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, 2022.

[4] C. Andrieu, M. Davy, and A. Doucet. Efficient particle filtering for jump Markov systems. application to time-varying autoregressions. *IEEE Transactions on Signal Processing*, 51(7):1762–1770, 2003.

[5] M. S. Arulampalam, S. Maskell, N. Gordon, and T. Clapp. A tutorial on particle filters for online nonlinear/non-Gaussian Bayesian tracking. *IEEE Transactions on Signal pProcessing*, 50(2):174–188, 2002.

[6] M. P. Balenzuela, A. G. Wills, C. Renton, and B. Ninness. Parameter estimation for jump Markov linear systems. *Automatica*, 135:109949, 2022.

[7] M. F. Bugallo, S. Xu, and P. M. Djurić. Performance comparison of EKF and particle filtering methods for maneuvering targets. *Digital Signal Processing*, 17(4):774–786, 2007.

[8] P. M. Djurić, J. H. Kotecha, J. Zhang, Y. Huang, T. Ghirmai, M. F. Bugallo, and J. Míguez. Particle filtering. *IEEE Signal Processing Magazine*, 20(5):19–38, 2003.

[9] A. Doucet, N. J. Gordon, and V. Krishnamurthy. Particle filters for state estimation of jump Markov linear systems. *IEEE Transactions on Signal Processing*, 49(3):613–624, 2001.

[10] A. Doucet and A. M. Johansen. A tutorial on particle filtering and smoothing: Fifteen years later. *Handbook of Nonlinear Filtering*, 12(656-704):3, 2009.

[11] P. Dua and D. Tuteja. Regime shifts in the behaviour of international currency and equity markets: A Markov-switching analysis. *Journal of*

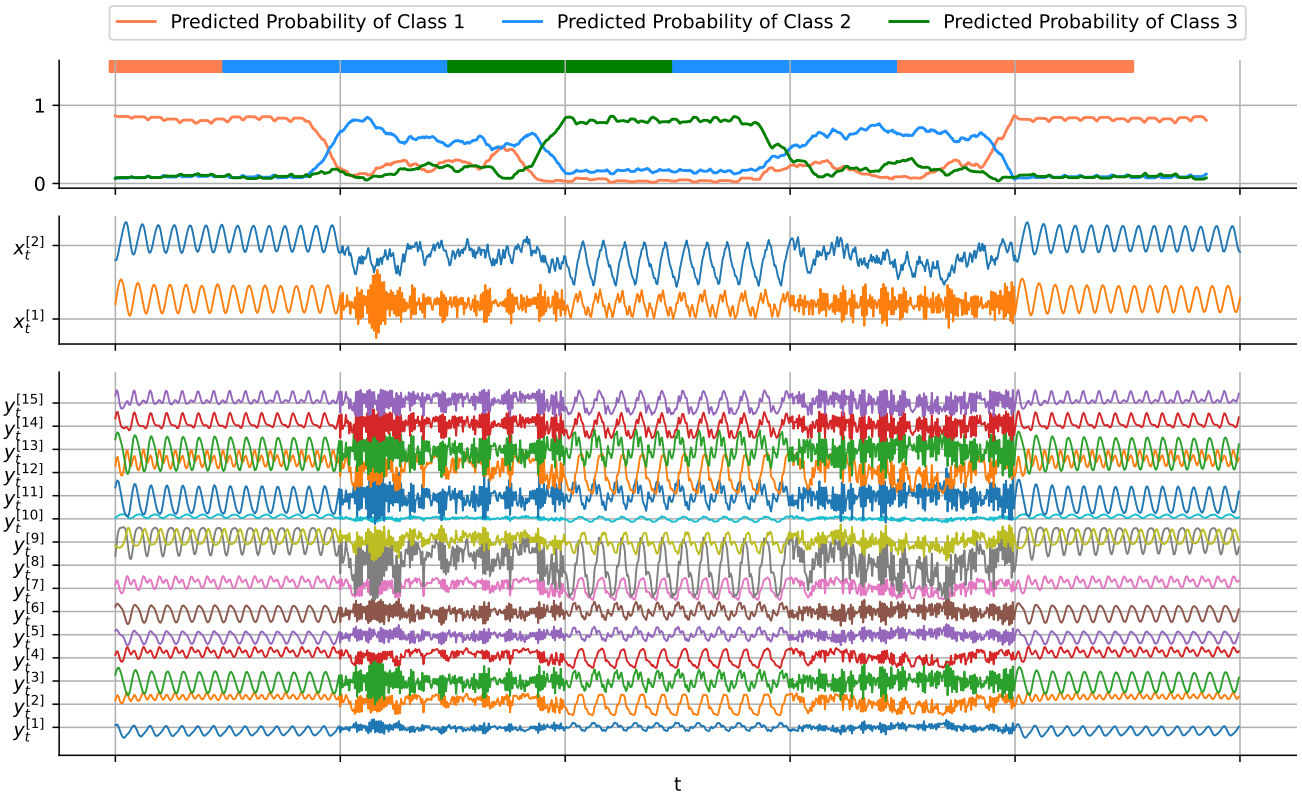


Fig. 3: Top: The estimated probabilities of the classes; Middle: The generated two-dimensional state process; Bottom: The generated 15-dimensional observed time series.

Quantitative Economics, 19(Suppl 1):309–336, 2021.

[12] Y. El-Laham, L. Yang, P. M. Djurić, and M. F. Bugallo. Particle filtering under general regime switching. In *2020 28th European Signal Processing Conference (EUSIPCO)*, pages 2378–2382. IEEE, 2021.

[13] S. Frühwirth-Schnatter. *Finite Mixture and Markov Switching Models*. Springer, 2006.

[14] M. Jahn and C. H. Weiß. Nonlinear GARCH-type models for ordinal time series. *Stochastic Environmental Research and Risk Assessment*, 38(2):637–649, 2024.

[15] C.-J. Kim and C. R. Nelson. *State-space models with REGIME Switching: Classical and Gibbs-sampling approaches with applications*. MIT press, 2017.

[16] M. Lázaro-Gredilla, J. Quinonero-Candela, C. E. Rasmussen, and A. R. Figueiras-Vidal. Sparse spectrum Gaussian process regression. *The Journal of Machine Learning Research*, 11:1865–1881, 2010.

[17] F. Leisen, L. Rossini, and C. Villa. A note on the posterior inference for the Yule-Simon distribution, 2017. https://ssrn.com/abstract=2900147.

[18] H. Li. *Learning to Rank for Information Retrieval and Natural Language Processing*. Springer Nature, 2022.

[19] T. Li, M. Bolic, and P. M. Djurić. Resampling methods for particle filtering: Classification, implementation, and strategies. *IEEE Signal Processing Magazine*, 32(3):70–86, 2015.

[20] W. Li, X. Chen, W. Wang, V. Elvira, and Y. Li. Differentiable bootstrap particle filters for regime-switching models. In *2023 IEEE Statistical Signal Processing Workshop (SSP)*, pages 200–204. IEEE, 2023.

[21] Y. Liu, M. Ajirak, and P. M. Djurić. Inference with deep gaussian process state space models. In *2022 30th European Signal Processing Conference (EUSIPCO)*, pages 792–796. IEEE, 2022.

[22] Y. Liu, M. Ajirak, and P. M. Djurić. Sequential estimation of Gaussian process-based deep state-space models. *IEEE Transactions on Signal Processing*, 2023.

[23] L. Martino, J. Read, V. Elvira, and F. Louzada. Cooperative parallel particle filters for online model selection and applications to urban mobility. *Digital Signal Processing*, 60:172–185, 2017.

[24] S. McGinnity and G. W. Irwin. Multiple model bootstrap filter for maneuvering target tracking. *IEEE Transactions on Aerospace and Electronic systems*, 36(3):1006–1012, 2000.

[25] C. Pandarinath, K. C. Ames, A. A. Russo, A. Farshchian, L. E. Miller, E. L. Dyer, and J. C. Kao. Latent factors and dynamics in motor cortex and their application to brain–machine interfaces. *Journal of Neuroscience*, 38(44):9390–9401, 2018.

[26] Z. Qu and F. Zhuo. Likelihood ratio-based tests for markov regime switching. *The Review of Economic Studies*, 88(2):937–968, 2021.

[27] B. Ristic and M. S. Arulampalam. Tracking a manoeuvring target using angle-only measurements: algorithms and performance. *Signal Processing*, 83(6):1223–1238, 2003.

[28] C. Y. Song, H.-L. Hsieh, B. Pesaran, and M. M. Shanechi. Modeling and inference methods for switching regime-dependent dynamical systems with multiscale neural observations. *Journal of Neural Engineering*, 19(6):066019, 2022.

[29] I. Urteaga, M. F. Bugallo, and P. M. Djurić. Sequential Monte Carlo methods under model uncertainty. In *2016 IEEE Statistical Signal Processing Workshop*, pages 1–5. IEEE, 2016.

[30] C. K. Williams and C. E. Rasmussen. *Gaussian Processes for Machine Learning*, volume 2. MIT press Cambridge, MA, 2006.