A Two-Layer Mixture Model of Gaussian Process Functional Regressions and Its MCMC EM Algorithm

Di Wu and Jinwen Ma

***Abstract*— The mixture of Gaussian processes (GPs) is capable of learning any general stochastic process based on a given set of (sample) curves for the regression and prediction problems. However, it is ineffective for curve clustering and prediction, when the sample curves are derived from different stochastic processes as independent sources linearly mixed together. In this paper, we propose a two-layer mixture model of GP functional regressions (GPFRs) to describe such a mixture of general stochastic processes or independent sources, especially for curve clustering and prediction. Specifically, in the lower layer, the mixture of GPFRs (MGPFRs) is developed for a cluster (or class) of curves within the *input space*. In the higher layer, the mixture of MGPFRs is further established to divide the curves into clusters according to its components in the *output space*. For the parameter estimation of the two-layer mixture of GPFRs, we develop a Monte Carlo EM algorithm based on a Monte Carlo Markov chain (MCMC) method, in short, the MCMC EM algorithm. We validate the hierarchical mixture of GPFRs and MCMC EM algorithm using synthetic and real-world data sets. Our results show that our new model outperforms the conventional mixture models in curve clustering and prediction.**

***Index Terms*— Curve clustering and prediction, EM algorithm, Gaussian process (GP), mixture of Gaussian processes (MGP/mix-GP), parameter learning.**

I. Introduction

GAUSSIAN process (GP) is a powerful learning model for time series regression and classification [1]–[4]. Actually, it has been successfully applied in the fields of pattern recognition, image processing, computer vision, and so on. However, there are two major limitations in the GP model for practical applications. First, a single GP cannot deal with data with a multimodal distribution that frequently appears in practice. Second, it is usually assumed that the mean function of the GP is zero. That is, the mean function is not characterized with respect to the input variables. Nevertheless, in certain complex cases, this input dependence should be considered. To date, there is no effective solution to such problems.

In order to handle the multimodal data set, Tresp [5] proposed a mixture of GPs (MGP), where the mean functions of GPs were still assumed to be zero. Specifically, each GP was considered as an expert, and thus the MGP can be viewed as an extension of the mixture of experts [6], [7]. Since then, various MGP models have been suggested and their applications have extended greatly in recent years. For the parameter estimation of MGPs, there are three main approaches: variational Bayesian (VB) inference, Monte Carlo Markov chain (MCMC), and EM algorithm. Specifically, VB assumes that the parameters and indicator variables are conditionally independent under the given data set, and aims to approximate the true posterior with a factorized form [8], [9], which allows to compute the posterior efficiently. However, such a factorial representation may be inaccurate in that the approximate posterior deviates a lot from the true one, especially when those parameters are highly correlated. The MCMC sampling seems to be a more accurate approximation method of the posterior [10]–[13]. However, the time consumption of MCMC is rather high and it is difficult to diagnose the convergence result.

In general, EM algorithm is an efficient and effective learning approach for mixture modeling [14], [15]. As for the MGP models, some approximation mechanisms must be adopted, because the computational complexity of the exact Q-function is exponential. In the heuristic EM algorithm [5], some parameters were estimated directly without any learning process, rather than through maximizing the Q-function. Such an estimation in fact could be blind and lacked the guidance of the data set. Variational EM algorithms [16]–[18] approximated the posterior with variational inference in E-step, which shared the same shortcomings with VB. The leave-one-out cross-validation (LOOCV) EM algorithm [19] computed the predictive output distribution for each training sample via the LOOCV mechanism, and summed up the expectations of these log predictive distributions to form the Q-function. However, the LOOCV approximation was not a very effective approximation of the Q-function.

Recently, the hard-cut EM algorithms [20]–[22] were suggested to divide the samples into different components clearly via certain clustering methods and then estimate the parameters of each GP separately in the M-step. This hard-cut mechanism really speeded up the algorithm, but might impair the algorithmic convergence, especially when these GPs are strongly overlapped. On the other hand, the MCMC method was also adopted to sample the hidden

Manuscript received November 21, 2016; revised August 9, 2017 and December 4, 2017; accepted December 6, 2017. Date of publication January 5, 2018; date of current version September 17, 2018. This work was supported by the Natural Science Foundation of China under Grant 61171138. *(Corresponding author: Jinwen Ma.)*

The authors are with the Department of Information Science, School of Mathematical Sciences & LMAM, Peking University, Beijing 100871, China (e-mail: 1401110048@pku.edu.cn; jwma@math.pku.edu.cn).

This paper has supplementary downloadable material available at http://ieeexplore.ieee.org., provided by the author.

Color versions of one or more of the figures in this paper are available online at http://ieeexplore.ieee.org.

Digital Object Identifier 10.1109/TNNLS.2017.2782711

variables, so that the Q-function can be computed and estimated with a given set of simulated samples of the hidden variables via MCMC sampling [23], [24]. In fact, this Monte Carlo EM algorithm can lead to a good result when the number of simulated samples is properly selected, and we will adopt this approximation mechanism for the EM algorithm in this paper.

In the above MGP models, the mean functions of the GPs are generally assumed to be zero. Accordingly, the estimated GPs are mainly separated in the input or time region. That is, from the viewpoint of a sample curve, each estimated GP is just a piece of the curve mainly located on an interval of the input region. Therefore, the learning algorithm aims to separate those GPs from the input region. This kind of multimodal data sets of mixture of GPs are referred to as *Type I data sets*. On the other hand, there is another kind of multimodal data sets of mixture of GPs that cannot be separated from the input region. For clarity, we refer to them as *Type II data sets*. A typical data set of Type II is generated from a mixture of GPs, where all the GPs are overlapped in the whole input region. In this case, each sample curve is subject to a GP and there are a number of sample curves belonging to different GPs. Since these GPs or sample curves cannot be separated in the input region, we have to separate them in the output space. In doing so, we can utilize the mean functions of GPs in the mixture of GPs. In fact, Shi *et al.* [25] proposed the GP functional regression (GPFR) as well as the mixture of GPFRs (mix-GPFR) [26], [27] to make the mean functions be learnable and flexible with the given data set. Specifically, the mix-GPFR model had improved its original mixture model of GPs (mix-GP) with the mean functions being zero [28], [29]. For parameter learning, the conventional EM algorithm and the MCMC method were used for the mix-GPFR model as well as the mix-GP model. For each GPFR model in the mixture, its mean function is a linear sum of certain basis functions, and the mixing proportions can be learned with the data set.

For many Type II data sets, each sample curve is subject to a general stochastic process, not just a single GP. In this situation, the sample curves are generated from a mixture of some general stochastic processes or independent sources, which often appears in practical applications, especially for curve clustering and prediction. Obviously, the mix-GPFR cannot describe this problem accurately. However, using a *divide-and-conquer* strategy, we can solve this difficult problem through a two-layer mixture model of GPs. First, we consider these sample curves as Type I data set and characterize them using a general MGP. For further separation in the output space, we use the GPFR model instead of the original GP model. Second, we use the mixture of the learned MGPs to characterize the Type II data set. Together, we construct a two-layer mixture of GPFRs (TMGPFRs).

In this paper, along the above analysis and direction, we propose a TMGPFR model for the complex Type II data sets. In the lower layer, the mixture of GPFRs (MGPFR) is established for each cluster of sample curves, and in the higher layer, the mixture of MGPFRs is further established for all the sample curves or the whole data set. Furthermore, a Monte Carlo EM algorithm is developed for the parameter estimation of the TMGPFR model using MCMC sampling. For clarity, this Monte Carlo EM algorithm is referred to as the MCMC EM algorithm. Moreover, we validate the proposed TMGPFR model and the MCMC EM algorithm using synthetic data sets and two real data sets, and show that the TMGPFR model outperforms the conventional mixture models in curve clustering and prediction.

The remainder of this paper is organized as follows. The GP and GPFR models are introduced in Section II. Section III presents the TMGPFR model. The MCMC EM algorithm is further derived in Section IV. The experimental results are summarized in Section V. Finally, we conclude this paper in Section VI.

II. GP AND GPFR MODELS

We begin with a brief introduction of the GP model. In fact, the GP is a common and important stochastic process in which any group of states (as random variables) are subject to a Gaussian distribution. Suppose that $y(x) \in \mathbb{R}$ is a stochastic process with an input variable $x \in \mathbb{R}$. For any given input data set $\{x_1, \ldots, x_N\}$ with any natural number N, $y(x)$ is defined as a GP if $\mathbf{y} = [y_1, \ldots, y_N]^T$, where $y_n = y(x_n)$ is subject to a Gaussian distribution $\mathcal{N}(\boldsymbol{\mu}, \mathbf{C})$, where $\boldsymbol{\mu} = [\mu(x_1), \ldots, \mu(x_N)]^T$ and $\mathbf{C} = [c(x_n, x_{n'})]_{N\times N}$ in which $\mu(x)$ is a mean function and $c(x, x')$ is a kernel or covariance function. Mathematically, it is denoted as

$$y(x) \sim \mathrm{GP}[\mu(x), c(x, x')].$$

A widely used kernel function parameterized by $\boldsymbol{\theta}$ in the GP model takes the following form:

$$c(x_n, x_{n'}|\boldsymbol{\theta}) = (\theta^{(1)})^2 \exp\left[-\frac{1}{2}(\theta^{(2)})^2(x_n - x_{n'})^2\right] + (\theta^{(3)})^2\delta_{nn'} \quad (1)$$

where $\delta_{nn'}$ is the Kronecker delta function and $\boldsymbol{\theta} = [\theta^{(1)}, \theta^{(2)}, \theta^{(3)}]$ in which $\theta^{(3)}$ essentially controls the noise in the GP model. We adopt this kernel function in the GP model.

On the other hand, the mean function $\mu(x)$ of the GP model is quite difficult to be estimated. In the literature, it is generally assumed to be zero. Occasionally, it is assumed to be a linear or other simple function of the input variable, which often leads to an unsatisfactory result. In order to improve this situation, Shi *et al.* [25] proposed the GPFR model in which a number of B-spline basis functions [30] were introduced and the mean function was assumed to be a linear combination of these basis functions with the coefficients estimated from the given data set. Specifically, the GPFR model can be described as follows.

With a mean function $\mu(x)$, a GP $y(x)$ is mathematically equivalent to that

$$y(x) = \mu(x) + \tau(x)$$

where $\tau(x) \sim \mathrm{GP}[0, c(x, x')]$. Suppose that $\boldsymbol{\phi} = [\phi_1(x), \ldots, \phi_D(x)]$ in which each $\phi_j(x)$ is a B-spline basis function.

Then, $\mu(x)$ can be approximated by the following functional regression model:

$$\mu(x) = \boldsymbol{\phi}\mathbf{b} = \sum_{j=1}^{D} b_j \phi_j(x) \tag{2}$$

where $\mathbf{b} = [b_1, \ldots, b_D]^T$ is a D-dimensional coefficient vector. Thus, the GPFR model can be described by

$$y(x) \sim \text{GPFR}(x; \mathbf{b}, \boldsymbol{\theta}).$$

In this GPFR model, there are two types of parameters, $\mathbf{b}$ and $\boldsymbol{\theta}$. In fact, the maximum likelihood (ML) approach can be applied to estimate these two types of parameters alternately step by step, which generally leads to a satisfactory estimation result. Several methods have been developed for the estimation of specific parameter $\boldsymbol{\theta}$, including the gradient-ascent ML method, expectation propagation, Laplace's approximation, variational bounds, and so on (refer to [1] and [31] for details). Generally, these methods can lead to a good estimation result. As we consider the situation that the GPs are strongly dependent on the input variable via the mean function, the GPFR model as well as the alternating ML estimation method will be utilized in our following network architecture and learning paradigm.

III. TMGPFR Model

In this section, we present the TMGPFR model for the complex Type II data sets. In the lower layer, an MGPFR is employed to model each cluster or class of complex curves, which can be divided into a number of curve segments that are subject to GPs. Particularly, the mean functions are not zero, but subject to the functional regression model given by (2). In the higher layer, a mixture of MGPFRs is further employed to describe all the curves structurally and then make the curve clustering and prediction in a more reasonable way.

A. Lower Layer: The MGPFR Model

The MGPFR model is a mixture of GPFRs located on some disjoint intervals of the input region. That is, the group of curve segments located on an interval are all subject to a GPFR. For clarity, we assume that there are a cluster of M curves or batches, as denoted by

$$\mathcal{D} = \{(x_{mn}, y_{mn}) | m = 1, \ldots, M; n = 1, \ldots, N_m\}$$

where each point in a curve is referred to as a sample. It should be noted that $\mathcal{D}$ and D are different notations. On the other hand, there are G GPFRs or disjoint intervals of the input region, i.e., the gth GPFR $y_g(x)$ is mainly located within the gth interval, being denoted by

$$y_g(x) \sim \text{GPFR}(x; \mathbf{b}_g, \boldsymbol{\theta}_g) \tag{3}$$

where $g = 1, \ldots, G$. In each curve, all samples are divided into G groups according to G GPFR models, respectively. That is, each group consists of the samples subject to a GPFR model. By combining these GPFRs in the input region, we have the MGPFR model. The details of the MGPFR model with the data set are given as follows.

The association of a sample with respect to the groups is described by an indicator variable $z_g^{(mn)}$, i.e., if the nth sample of the mth curve belongs to the gth group, $z_g^{(mn)} = 1$; otherwise, $z_g^{(mn)} = 0$. So, $z_g^{(mn)}$ is subject to

$$P(z_g^{(mn)} = 1) = \eta_g$$

where $\sum_{g=1}^{G} \eta_g = 1$. Under the condition that the indicator variables with respect to the gth group are 1, the input variable x_{mn} is subject to a Gaussian distribution, i.e.,

$$x_{mn} | (z_g^{(mn)} = 1) \sim \mathcal{N}(h_g, s_g^2)$$

where h_g and s_g are the mean and standard deviation (SD) of the distribution, respectively. For mathematical details, Gaussian distribution is described in Section 1 of the Supplementary Material. Furthermore, $y_g(x)$ is subject to a GPFR model given by (3).

It is clear that the information flow direction of the MGPFR model is just $z_g^{(mn)} \to x_{mn} \to y_{mn}$. In fact, the MGPFR model is quite different from the mix-GPFR model, since the sample curves are divided into GPFRs from the output space in the mix-GPFR model, that is, a sample curve is completely subject to a GPFR model, which limits the capability of learning more complex curves.

B. Higher Layer: The Mixture of MGPFRs

According to the above description, the MGPFR model is a complete model, which can solve the modeling of any stochastic process with a set of sample curves. Indeed, it works well in certain situations. However, if these curves belong to different clusters associated with different mean functions or kernel functions, the MGPFR model cannot characterize them well and we need to consider a more flexible architecture. Naturally, the mixture of MGPFRs owns such an architecture and we employ it as the second layer. In this way, those MGPFRs for different clusters of curves become the components of the new mixture and each sample curve belongs to one component. As a whole, it is just the TMGPFR.

We assume that there are K components corresponding to K clusters of complex curves denoted by $\mathcal{D}$ in the same manner, but generated from a TMGPFR model. For the kth component or MGPFR, there are G_k GFPRs, that is, each subcomponent stochastic process $y_{kg}(x)$ is subject to a particular GPFR model in its gth interval

$$y_{kg}(x) \sim \text{GPFR}(x; \mathbf{b}_{kg}, \boldsymbol{\theta}_{kg}) \tag{4}$$

where $g = 1, \ldots, G_k$. As the MGPFR with G_k-specific GPFRs, the kth component can be described in the same way as in Section III-A. Together, we present the mathematical details of the TMGPFR model as follows.

The association of a sample curve with respect to the components (i.e., MGPFRs) in the higher layer mixture can be described by an indicator variable $z_k^{(m)}$, i.e., if the mth sample curve belongs to the kth component, $z_k^{(m)} = 1$; otherwise,

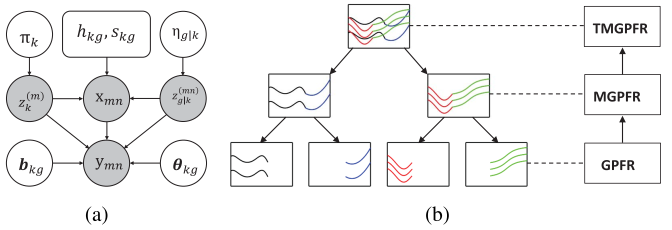


Fig. 1. (a) Information flowchart of the variables with the parameters in the TMGPFR model. (b) Hierarchical structure of the TMGPFR model: the lower layer consists of GPFRs, the middle layer consists of MGPFRs, and the higher layer is a mixture of MGPFRs.

$z_k^{(m)} = 0$. Moreover, $z_k^{(m)}$ is subject to the following probability distribution:

$$P(z_k^{(m)} = 1) = \pi_k$$

where $\sum_{k=1}^{K} \pi_k = 1$.

Furthermore, the association of a sample in a curve of the kth component (or cluster) with respect to the groups (i.e., GPFRs) in the lower layer mixture can be described by another indicator variable $z_{g|k}^{(mn)}$, i.e., if the nth sample of the mth curve belongs to the gth group on the condition that the mth curve belongs to the kth component, $z_{g|k}^{(mn)} = 1$; otherwise, $z_{g|k}^{(mn)} = 0$. Moreover, $z_{g|k}^{(mn)}$ is subject to the following probability distribution:

$$P(z_{g|k}^{(mn)} = 1) = \eta_{g|k}$$

where $\sum_{g=1}^{G_k} \eta_{g|k} = 1$. Let $z_{kg}^{(mn)} = z_k^{(m)} z_{g|k}^{(mn)}$ and assume that the input variable is subject to a Gaussian distribution in each group, that is

$$x_{mn} | (z_{kg}^{(mn)} = 1) \sim \mathcal{N}(h_{kg}, s_{kg}^2)$$

and $y(x)$ is subject to a GPFR model given by (4).

Clearly, the information flow direction of the TMGPFR model is $(z_k^{(m)}, z_{g|k}^{(mn)}) \rightarrow x_{mn} \rightarrow y_{mn}$ and the detailed information flowchart is also shown in Fig. 1(a). The complete structure of the TMGPFR model is shown in Fig. 1(b).

For clarity and analysis, we denote the set of all the parameters in the TMGPFR model by

$$\Theta = \{(\pi_k, \eta_{g|k}, \Theta_{kg}) | k = 1, \ldots, K; \quad g = 1, \ldots, G_k\}$$

where $\Theta_{kg} = \{h_{kg}, s_{kg}, \mathbf{b}_{kg}, \boldsymbol{\theta}_{kg}\}$, the m-th curve data set by $\mathcal{D}_m = \{(x_{mn}, y_{mn}) | n = 1, \ldots, N_m\}$, the set of all the indicator variables for the lower layer mixture by a tensor $\mathcal{Z} = \{\mathcal{Z}_k^{(m)} | m = 1, \ldots, M; k = 1, \ldots, K\}$ with its component matrixes $\mathcal{Z}_k^{(m)} = (z_{g|k}^{(mn)})_{N_m \times G_k}$ (i.e., for $n = 1, \ldots, N_m$; $g = 1, \ldots, G_k$), and the set of all the indicator variables for the higher layer mixture by a matrix $\mathcal{A} = (z_k^{(m)})_{M \times K}$ (i.e., for $m = 1, \ldots, M$; $k = 1, \ldots, K$). Finally, the total log likelihood function, $\mathcal{L}$, can be expressed by

$$\mathcal{L}(\Theta, \mathcal{Z}, \mathcal{A}) = \sum_{m=1}^{M} \sum_{k=1}^{K} z_k^{(m)} \ln P(\mathcal{D}_m, \mathcal{Z}_k^{(m)} | \Theta) \qquad (5)$$

where

$$P(\mathcal{D}_m, \mathcal{Z}_k^{(m)} | \Theta) = \pi_k \prod_{g=1}^{G_k} \left\{ \prod_{n=1}^{N_m} [\eta_{g|k} p(x_{mn} | \Theta_{kg})]^{z_{g|k}^{(mn)}} p(\mathbf{y}_{g|k}^{(m)} | \mathbf{x}_{g|k}^{(m)}, \Theta_{kg}) \right\}$$

with $\mathbf{x}_{g|k}^{(m)} = [x_{mn} | z_{g|k}^{(mn)} = 1; n = 1, \ldots, N_m]$ is just the column vector of x_{mn} belonging to the gth group and $\mathbf{y}_{g|k}^{(m)}$ is defined for y_{mn} in the same way as $\mathbf{x}_{g|k}^{(m)}$.

IV. MCMC EM Algorithm for the TMGPFR Model

In this section, we establish a feasible EM algorithm for identifying the parameters of the TMGPFR model. Since the time complexity of the conventional EM algorithm is of exponential order in this situation, we utilize the MCMC method to estimate the Q-function so that the EM procedure can be feasibly implemented with a Monte Carlo estimated Q-function, i.e., $\hat{\text{Q}}$-function.

A. Derivation of $\hat{Q}$-function

In the design of EM algorithm, the Q-function is a key to the computation of both E-step and M-step. However, for the TMGPFR model, there are two intractable problems about the Q-function.

1) The Q-function is difficult to be explicitly expressed, since there are two different indicator variables $z_k^{(m)}$ and $z_{g|k}^{(mn)}$ involved in the two-layer mixture structure.
2) The time complexity of calculating the Q-function is of exponential order, since all samples are not independent.

In fact, the samples belonging to a GPFR model are strongly dependent. In order to overcome the first problem, we consider two kinds of indicator variables separately and express the Q-function by two consecutive steps.

We begin to derive $\mathcal{F}(\Theta, \mathcal{Z} | \hat{\Theta})$, which is the conditional expectation of the log likelihood $\mathcal{L}(\Theta, \mathcal{Z}, \mathcal{A})$ [given by (5)] with respect to the indicator variable matrix $\mathcal{A}$ for the higher layer mixture. For convenience, we let $\mathcal{Z}^{(m)} = \{\mathcal{Z}_k^{(m)} | k = 1, \ldots, K\}$. We then have

$$\mathcal{F}(\Theta, \mathcal{Z} | \hat{\Theta}) = \mathbb{E}_{\mathcal{A}}[\mathcal{L}(\Theta, \mathcal{Z}, \mathcal{A}) | \mathcal{D}, \mathcal{Z}, \hat{\Theta}] = \sum_{m=1}^{M} \sum_{k=1}^{K} \mathbb{E}[z_k^{(m)} | \mathcal{D}, \mathcal{Z}^{(m)}, \hat{\Theta}] \ln P(\mathcal{D}_m, \mathcal{Z}_k^{(m)} | \Theta).$$

We further obtain the Q-function, i.e., $\text{Q}(\Theta | \hat{\Theta})$, of the EM algorithm for the TMGPFR model, which is just the expectation of $\mathcal{F}(\Theta, \mathcal{Z} | \hat{\Theta})$ with respect to the indicator variable tensor $\mathcal{Z}$ for the lower layer mixture

$$\text{Q}(\Theta | \hat{\Theta}) = \mathbb{E}_{\mathcal{Z}}[\mathcal{F}(\Theta, \mathcal{Z} | \hat{\Theta}) | \mathcal{D}, \hat{\Theta}] = \sum_{\mathcal{Z}} P(\mathcal{Z} | \mathcal{D}, \hat{\Theta}) \mathcal{F}(\Theta, \mathcal{Z} | \hat{\Theta})$$

where $P(\mathcal{Z} | \mathcal{D}, \hat{\Theta}) \propto \prod_{m=1}^{M} \prod_{k=1}^{K} P(\mathcal{D}_m, \mathcal{Z}_k^{(m)} | \hat{\Theta})$.

As for the second problem, we utilize a specific MCMC method to estimate the above Q-function. For clarity, we denote the ith simulated sample of each indicator variable

$z_{g|k}^{(mn)}$ by $z_{g|k}^{(mni)}$ for $i = 1, \ldots, I$. In addition, we define $\mathcal{Z}^{(i)}$ by replacing every $z_{g|k}^{(mn)}$ of $\mathcal{Z}$ with $z_{g|k}^{(mni)}$, and all the notations about the ith simulated sample are defined in the same way. Actually, in our complete MCMC sampling process, we generate I simulated samples of the indicator variable tensor $\mathcal{Z}$ according to $P(\mathcal{Z}|\mathcal{D}, \hat{\Theta})$ via a specialized Gibbs sampling procedure. With this set of the MCMC samples, the $\hat{\mathrm{Q}}$-function, i.e., the estimate of Q-function, can be computed by

$$\begin{aligned}\hat{\mathrm{Q}}(\Theta|\hat{\Theta}) &= \frac{1}{I}\sum_{i=1}^{I}\mathcal{F}(\Theta, \mathcal{Z}^{(i)}|\hat{\Theta}) \\ &= \frac{1}{I}\sum_{i=1}^{I}\sum_{m=1}^{M}\sum_{k=1}^{K}\alpha_k^{(mi)} \ln P(\mathcal{D}_m, \mathcal{Z}_k^{(mi)}|\Theta) \qquad (6)\end{aligned}$$

where $\alpha_k^{(mi)}$ is computed in the following way:

$$\begin{aligned}\alpha_k^{(mi)} &= \mathbb{E}[z_k^{(m)}|\mathcal{D}, \mathcal{Z}^{(mi)}, \hat{\Theta}] \\ &\propto \hat{\pi}_k^{(m)} \prod_{g=1}^{G_k}\left[\prod_{n=1}^{N_m} p(x_{mn}|\hat{\Theta}_{kg})^{z_{g|k}^{(mni)}}\, p(\mathbf{y}_{g|k}^{(mi)}|\mathbf{x}_{g|k}^{(mi)}, \hat{\Theta}_{kg})\right]\end{aligned} \qquad (7)$$

with $\sum_{k=1}^{K}\alpha_k^{(mi)} = 1$.

B. Algorithmic Implementation

Based on the $\hat{\mathrm{Q}}$-function derived in Section IV-A, the MCMC EM algorithm of the TMGPFR model can be implemented by five steps in Algorithm 1. For clarity, we divide the E-step into two substeps, denoted as "E1-step" and "E2-step" respectively.

```
Algorithm 1 MCMC EM Algorithm for TMGPFRs
Input: $\mathcal{D}$, $K$, $\{G_k\}$, $D$.
Output: $\Theta$, $\{\alpha_k^{(mi)}\}$, $\{z_{g|k}^{(mni)}\}$, $\{\hat{Q}_r\}$.
1: Initialize $z_k^{(m)}$ by clustering all the curves into $K$ components, i.e., if the $m$-th curve belongs to the $k$-th cluster, $z_k^{(m)} = 1$; otherwise, $z_k^{(m)} = 0$. In the same way, initialize $z_{g|k}^{(mn)}$ by clustering $\{x_{m1}, \ldots, x_{mN_m}\}$ into $G_k$ groups. And then, initialize the parameters of $\Theta$ with their ML estimators by an ML estimator. Set $r = 1$, where $r$ is the number of current iteration.
2: E1-step. With $\hat{\Theta}$, generate a series of simulated samples $\mathcal{Z}^{(1)}, \ldots, \mathcal{Z}^{(I)}$ of the indicator variable tensor $\mathcal{Z}$ by our specialized Hierarchical Blocking Gibbs Sampling (HBGS) method that is described in detail in Section 2 of the Supplementary Material.
3: E2-step. Calculate $\alpha_k^{(mi)}$ by (7). According to $z_{g|k}^{(mni)}$ and $\alpha_k^{(mi)}$, $\hat{\mathrm{Q}}(\Theta|\hat{\Theta})$ is further calculated by (6).
4: M-step. Solve $\Theta$ by maximizing the $\hat{\mathrm{Q}}$-function. The detailed process is given in Section 3 of the Supplementary Material.
5: If $[(\hat{Q}_r + \hat{Q}_{r-1}) - (\hat{Q}_{r-2} + \hat{Q}_{r-3})]/|\hat{Q}_{r-2} + \hat{Q}_{r-3}| < \varepsilon$ or $r \geq T$, where $\hat{Q}_r$ is the value of $\hat{\mathrm{Q}}$-function at the end of the $r$-th iteration and $T$ is set as the largest number of iterations, stop; otherwise, $r = r + 1$ and return to Step 2, i.e., E1-step.
```

During each iteration of the MCMC EM algorithm, the simulated samples $\mathcal{Z}^{(1)}, \ldots, \mathcal{Z}^{(I)}$ and the parameter set Θ are updated iteratively in the E-step and M-step, respectively. For the conventional EM algorithm, the convergence criterion is generally set by $(Q_r - Q_{r-1})/|Q_{r-1}| < \varepsilon$. However, as for the MCMC EM algorithm, $\hat{Q}_r$ may fluctuate during the learning process due to the randomness of simulated samples. Therefore, we adopt a relative long-term convergence criterion as $[(\hat{Q}_r + \hat{Q}_{r-1}) - (\hat{Q}_{r-2} + \hat{Q}_{r-3})]/|\hat{Q}_{r-2} + \hat{Q}_{r-3}| < \varepsilon$. It has been found that the threshold value $\varepsilon = 0.002$ works well in our experiments. On the other hand, we also adopt $r \leq T$ to avoid that the algorithm may be too slow in some special situations. It has also been found from the experiments that the algorithm always converges around the 20th iteration, so we set $T = 24$.

In fact, as long as the Hierarchical Blocking Gibbs Sampling (HBGS) is properly implemented, this MCMC EM algorithm is quite effective for learning the parameters of the TMGPFR model. Although it may be sensitive to the initialization of the indicator variables for the lower layer mixture, we have found that the k-means algorithm works well in practice for initialization. For the MGPFR model, we can also establish such an MCMC EM algorithm by using the HBGS method for its indicator variables in the same way. Since the MGPFR model is also a new development of the general MGP model, its experimental results with the MCMC EM algorithm are also given and compared in Section 6 of the Supplementary Material.

As a general EM algorithm, this MCMC EM algorithm needs to set K, the number of major components, correctly or consistently with the structure of the data; otherwise, we cannot get a reasonable result on curve clustering and prediction. This is a model selection problem related with the finite mixture model and the EM algorithm. In the structure of the TMGPFR model, G_k can be set sufficiently large to guarantee that each MGPFR model can describe a curve cluster accurately. The selection of K becomes very important, since it should be equal to the number of clusters or classes of the curves, which is generally a very challenging problem to estimate in practice. However, we can utilize the functional PCA method [32] to reduce the dimension of the curves remarkably and then apply certain automated model selection algorithms [33]–[36] to obtain the true value of K for a given data set. Certainly, the competitive learning mechanisms involved in these automated model selection algorithms may be introduced into the MCMC EM algorithm, so that the model selection can be made automatically during the parameter learning. In fact, a dynamical model selection mechanism was already proposed for the hard-cut EM algorithm for the MGP model with a synchronously balancing criterion [37], while the reversible jump MCMC framework was also applied to predict the possible K for the mix-GP model [29]. However, since the structure of the TMGPFR model is hierarchical and the model selection involves $G_1, \cdots, G_K$ and D, the task of automated model selection becomes rather difficult and we put off this exploration in the future. But for a relatively small-scale problem, we can use the cross-validation method to select the best K for a data set and model to make the curve clustering and prediction. The cross-validation method for model selection is further described in Section 5 of the Supplementary Material.

After the establishment of the MCMC EM algorithm, we further introduce our prediction method for the TMGPFR model. A test curve can be defined as the $(M + 1)$th curve denoted by $\mathcal{D}_{M+1} = \{(x_{M+1,n}, y_{M+1,n})|n = 1, \ldots, N_{M+1}\}$. Our purpose is to make a reasonable prediction y^* at a new input x^* in the $(M + 1)$th curve. After Θ is finally estimated by our EM algorithm, we calculate $z_{g|k}^{(M+1,n,i)}$ by the HBGS method. Thus, the predictive output is given by

$$\hat{y}_{kg}^{(i)} = \boldsymbol{\phi}^*\mathbf{b}_{kg} + \mathbf{c}^*\big(\mathbf{C}_{g|k}^{(M+1,i)}\big)^{-1}\big(\mathbf{y}_{g|k}^{(M+1,i)} - \Phi_{g|k}^{(M+1,i)}\mathbf{b}_{kg}\big)$$

where $\boldsymbol{\phi}^* = [\phi_1(x^*), \ldots, \phi_D(x^*)]$

$$\mathbf{C}_{g|k}^{(mi)} = \big[c(x_{mn}, x_{mn'})|z_{g|k}^{(mni)} = z_{g|k}^{(mn'i)} = 1;\quad n, n' = 1, \ldots, N_m\big]$$

which is the kernel-based covariance matrix of the gth GPFR, $\Phi_{g|k}^{(mi)} = [\phi_j(x_{mn})|z_{g|k}^{(mni)} = 1; n = 1, \ldots, N_m; j = 1, \ldots, D]$ is the matrix of $\phi_j(x_{mn})$, in which the number of columns is D, and $\mathbf{c}^* = [c(x^*, x_{M+1,n})|z_{g|k}^{(M+1,n,i)} = 1; n = 1, \ldots, N_{M+1}]$ represents a row vector of $c(x^*, x_{M+1,n})$. As a result, we have the overall predictive output as follows:

$$\hat{y}^* = \frac{1}{I}\sum_{i=1}^{I}\sum_{k=1}^{K}\alpha_k^{(M+1,i)}\sum_{g=1}^{G_k}\alpha_{g|k}^*\hat{y}_{kg}^{(i)}$$

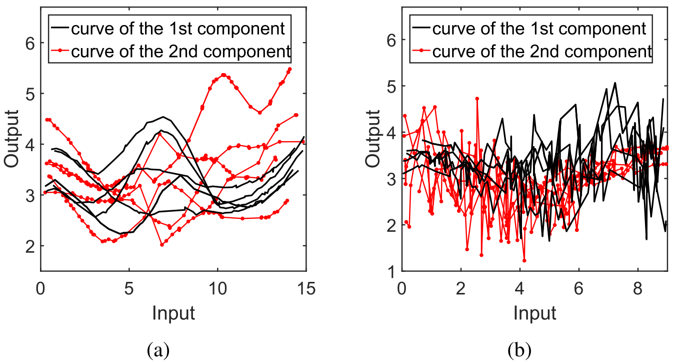
where $\alpha_k^{(M+1,i)}$ is also obtained by (7) and $\alpha_{g|k}^* = \mathbb{E}[z_{g|k}^* = 1|x^*, \Theta] = \eta_{g|k}p(x^*|\Theta_{kg})/\sum_{g=1}^{G_k}\eta_{g|k}p(x^*|\Theta_{kg})$.

V. Experimental Results

In this section, we test the TMGPFR model on some typical synthetic data sets and two real-world data sets. We also compare them with the GP, MGP, mix-GP, GPFR, mix-GPFR, and MGPFR models in the literature. It should be noted that the mix-GP, MGP, and GP models assume that their GPs have zero mean functions and the MGP model consists of some separated GPs along the input region. The connections between all these models are further discussed in Section 4 of the Supplementary Material.

A. On Synthetic Data Sets of the TMGPFR Model

We begin to test the TMGPFR model on synthetic data sets generated from different TMGPFR models without any overlap among the subcomponents, i.e., GPFRs. For clarity, we first generate the original data set from a typical TMGPFR model and then extend it to the small, noisy, and unbalanced data sets, respectively, for the more difficult situations. Specifically, the original data set is generated from the TMGPFR model with two components, which, respectively, have two disjoined subcomponents. That is, the two components are the MGPFRs that consist of two connected GPFRs or groups as their subcomponents, respectively. Specifically, the first component consists of two groups or GPFRs whose mean functions are set by $\mu_{1|1}(x) = 0.5\sin[0.125(x-4)^2] + 3$ and $\mu_{2|1}(x) = 0.06(x-8)^2 - 0.3x + 5.6483$, respectively. As for the second component, the mean functions of two GPFRs are set by $\mu_{1|2}(x) = -3(2\pi)^{-0.5}\exp[-0.125(x-4)^2] + 3.7$ and $\mu_{2|2}(x) = 0.5\arctan(0.5x-5) + 3.5277$, respectively. In this situation, we set the number of B-spline basis functions $D = 32$, and denote $\boldsymbol{\theta}_{kg} = [\theta_{kg}^{(1)}, \theta_{kg}^{(2)}, \theta_{kg}^{(3)}]$. Moreover, we set $K = 2$ and $G_1 = G_2 = G = 2$ according to the structure of the data set. In the same way, we assume $G_1 = \cdots = G_K = G$ and the remainder of this section share the same assumption. There are 200 training curves and 400 test curves. Furthermore, each training curve consists of 50 samples, while each test curve consists of 150 samples in which 40 samples are known in advance and 110 samples are needed to be predicted. As shown in Fig. 2(a), there are ten sample curves of this data set, which are rather difficult to be clustered.



(a)

(b)

Fig. 2. (a) Sketch of ten sample curves in the original data set of the TMGPFR model. (b) Sketch of ten sample curves in the original data set of the mix-GGPFR model.

According to the experimental results, the sketches of the variation of $\hat{\alpha}_1^{(m)} = \frac{1}{I}\sum_{i=1}^{I}\alpha_1^{(mi)}$ on 200 training curves at three iterations of the MCMC EM algorithm are shown in Fig. 3(a). In fact, we can easily evaluate our EM algorithm by these resulted values of $\hat{\alpha}_1^{(m)}$ (i.e., an estimate of the posterior probability of $z_1^{(m)}$). It is found that our MCMC EM algorithm is effective on the evolution of $\hat{\alpha}_1^{(m)}$, because $\hat{\alpha}_1^{(m)} \approx 1$ for all the curves of the first component and $\hat{\alpha}_1^{(m)} \approx 0$ for all the curves of the second component just after three iterations. Therefore, our MCMC EM algorithm for the TMGPFR model is both effective and efficient in this case, which is further shown by the following experimental results on the more complex data sets in Fig. 3(b)–(d). In light of the maximum posterior probability under the resulted TMGPFR model, we assign any curve into one of the components or MGPFRs. That is, all curves can be clustered into K classes.

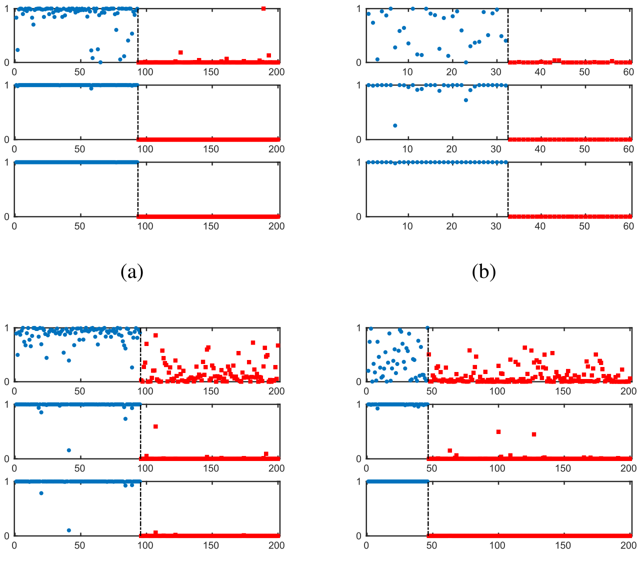
The estimates of the parameters in the TMGPFR model via our MCMC EM algorithm are listed in Table I. The sketches of the real and predicted mean functions are shown in Fig. 4(a) and (b). It is found that the average estimates (AEs) and true values (TVs) are almost identical, and the estimated SDs of the parameters are rather small except for the cases of π_k and $\theta_{kg}^{(1)}$. The estimated SD of π_k is relatively larger due to the randomness of the training data set. On the other hand,

TABLE I

AEs OF THE PARAMETERS OF THE TMGPFR MODEL WITH ESTIMATED SDs VERSUS THE TVs OVER 50 TRIALS ON THE ORIGINAL DATA SET

	π_1	π_2
TV	0.5000	0.5000
AE	0.4989	0.5011
SD	0.0224	0.0224

		$\eta_{g\|k}$	h_{kg}	s_{kg}	$\theta_{kg}^{(1)}$	$\theta_{kg}^{(2)}$	$\theta_{kg}^{(3)}$
$k=1, g=1$	TV	0.5000	4.5000	2.1587	0.6325	0.4472	0.0200
	AE	0.5003	4.5225	2.1620	0.6287	0.4487	0.0198
	SD	0.0042	0.0383	0.0134	0.0129	0.0045	0.0002
$k=1, g=2$	TV	0.5000	12.000	1.4270	0.2236	0.2828	0.0200
	AE	0.4997	11.994	1.4324	0.2241	0.2834	0.0197
	SD	0.0042	0.0199	0.0116	0.0092	0.0094	0.0002
$k=2, g=1$	TV	0.5000	3.0000	1.4270	0.3162	0.3464	0.0200
	AE	0.4996	2.9957	1.4343	0.3117	0.3484	0.0200
	SD	0.0054	0.0306	0.0194	0.0138	0.0095	0.0004
$k=2, g=2$	TV	0.5000	10.500	2.1587	0.7071	0.5477	0.0200
	AE	0.5004	10.469	2.1680	0.7033	0.5497	0.0198
	SD	0.0054	0.0354	0.0159	0.0264	0.0063	0.0002



(a)

(b)

(c)

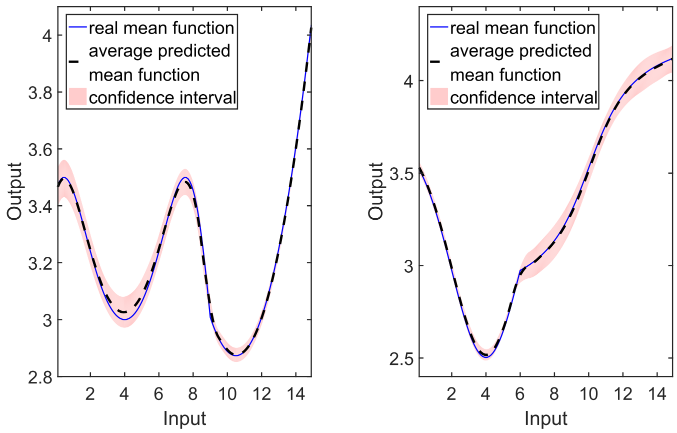
(d)

Fig. 3. Sketches of variation of $\hat{\alpha}_1^{(m)}$ on all the samples at three particular iterations of the MCMC EM algorithm for the TMGPFR model on four data sets of TMGPFR. We give each curve a number for distinction, and the horizontal axis in each iteration is the number of the curve. The round and square points for the values of $\hat{\alpha}_1^{(m)}$ represent the corresponding curves belonging to the first and second components, respectively. Moreover, the dashed-dotted line separates the points of two components. That is, the points on the left and right of the line represent the corresponding curves belonging to the first and second components, respectively. (a) and (b) At the first three iterations on the original and small data sets, respectively. (c) and (d) At the first, third, and fifth iterations on the noisy and unbalanced data sets, respectively.

since the TMGPFR model is robust to $\theta_{kg}^{(1)}$, it is less critical that the estimated SD of $\theta_{kg}^{(1)}$ is relatively larger.

We further compare the TMGPFR and other state-of-the-art mixture models on more complex data sets of the TMGPFR model. Specifically, they come from the original data set, but change in different ways as follows.

1) *The Small Data Set:* Being a smaller size data set of the original one. There are 60 training curves and 120 test curves in the small data set. Moreover, each training curve consists of 30 samples, while each test curve has 24 known samples and 66 predictive samples.
2) *The Noisy Data Set:* Being generated with a larger noise than the original data set. In the noisy data set, $\theta_{1,1}^{(3)} = \theta_{1,2}^{(3)} = \theta_{2,1}^{(3)} = \theta_{2,2}^{(3)} = 0.2000$.
3) *The Unbalanced Data Set:* Being an unbalanced data set of the original data set with different values of π_k and $\eta_{g|k}$. In the unbalanced data set, $\pi_2 = \eta_{1|1} = \eta_{2|2} = 0.7500$ and $\pi_1 = \eta_{1|2} = \eta_{2|1} = 0.2500$.



(a)

(b)

Fig. 4. (a) and (b) Sketches of the real and predicted mean functions of two components in the TMGPFR model on the original data set of TMGPFR, respectively.

The prediction experimental results of the TMGPFR model on the original, small, noisy, and unbalanced data sets are given in Table II, in which we further compare the TMGPFR model with the other mixture models, such as the MGPFR, mix-GPFR [26], MGP, and mix-GP [28], as well as the pure GPFR and GP models. It should be noted that the MGPFR and MGP models aim to model a stochastic process via a number of GPFRs (with adaptive mean functions) or GPs (with zero mean functions) being linearly mixed along the input region, while the mix-GPFR and mix-GP models aim to model a stochastic process via a number of GPFRs or

TABLE II

AVERAGE PREDICTED RMSEs OF THE TMGPFR AND COMPARATIVE MODELS WITH SDs OVER 30 TRIALS ON THE ORIGINAL, SMALL, NOISY, AND UNBALANCED DATA SETS GENERATED FROM THE TMGPFR MODELS. THE BEST RESULTS ARE SHOWN IN BOLD FONT

Model	Algorithm	K	G	Original	Small	Noisy	Unbalanced
TMGPFR	MCMC EM	2	2	**0.0581 ± 0.0033**	**0.0755 ± 0.0069**	**0.2293 ± 0.0010**	**0.0602 ± 0.0068**
MGPFR	MCMC EM	-	2	0.1041 ± 0.0065	0.1342 ± 0.0151	0.2435 ± 0.0027	0.1087 ± 0.0074
mix-GPFR	Conventional EM	2	-	0.0755 ± 0.0041	0.1057 ± 0.0135	0.2345 ± 0.0016	0.0726 ± 0.0036
MGP	MCMC EM	-	2	0.1462 ± 0.0139	0.1856 ± 0.0238	0.2564 ± 0.0034	0.0715 ± 0.0160
mix-GP	Conventional EM	2	-	0.1119 ± 0.0145	0.1587 ± 0.0267	0.2487 ± 0.0034	0.1255 ± 0.0133
GPFR	MLE	-	-	0.0775 ± 0.0042	0.1082 ± 0.0145	0.2358 ± 0.0015	0.0741 ± 0.0033
GP	Gradient method	-	-	0.1046 ± 0.0112	0.1624 ± 0.0279	0.2506 ± 0.0035	0.1205 ± 0.0127

TABLE III

AVERAGE CURVE CARs OF THE TMGPFR, MIX-GPFR, AND MIX-GP MODELS OVER 30 TRIALS ON THE ORIGINAL, SMALL, NOISY, AND UNBALANCED DATA SETS GENERATED FROM THE TMGPFR MODELS

Model	Original	Small	Noisy	Unbalanced
TMGPFR	**99.93%**	**99.98%**	**99.54%**	**99.98%**
mix-GPFR	55.31%	62.38%	53.36%	61.64%
mix-GP	54.00%	54.52%	52.71%	61.57%

GPs being linearly mixed in the output space. According to the average predicted root-mean-square errors (RMSEs) on the four synthetic data sets listed in Table II, we find that the TMGPFR model with our proposed EM algorithm works well on curve prediction and outperforms the other models considerably. In addition, the average predicted RMSEs of the GPFR, MGPFR, and mix-GPFR models are always much smaller than those of the GP, MGP, and mix-GP models, respectively. Therefore, the mean functions play an important role on the GP modeling. We also find that the average predicted RMSEs of the GPFR model are even much smaller than those of the MGPFR model, which further demonstrates that the GPFR model is more robust than the MGPFR model. For curve clustering, the average curve classification accuracy rates (CARs) of the TMGPFR model on the four data sets are listed in Table III, which are all above 99.5% and are remarkably better than those of the two comparative models. As the components of the MGPFR or MGP model are only a set of piecewise curve segments and thus unable to make curve clustering analysis on the whole input region, we do not compare them on curve clustering.

B. On Synthetic Data Sets of the Mix-GGPFR Models

We further test the TMGPFR model on synthetic data sets generated from different stochastic processes. For convenience, we utilize the generalized GPFR (GGPFR) model to generate each cluster of sample curves. That is, the whole data set is generated from a mixture of GGPFRs. In fact, the GGPFR model is different from the GPFR model via letting the parameter $\boldsymbol{\theta}$ in (1) be a function of the input x, i.e., $\boldsymbol{\theta} = \boldsymbol{\theta}(x)$. In this case, each sample curve is generated globally and strongly depended on the input variable. Specifically, we generate the original synthetic data set by a mix-GGPFR model with $\boldsymbol{\theta}_k = [\theta_k^{(1)}, \theta_k^{(2)}, \lambda_k^{(1)} x_n + \lambda_k^{(2)}]$ for $k = 1, 2$, with its parameters being listed in Table IV. The mean functions of the first and second components are just the same as $\mu_{1|1}(x)$ and $\mu_{1|2}(x)$ in Section V-A, respectively. The input variables are generated from a uniform distribution in the interval $[0, 9]$. The numbers of training and test samples remain the same as those of the original data set in Section V-A. In our learning process, we set K and D in the same way as in Section V-A. As for G, we test it from $G = 2$ and find out that $G = 2$ is good enough and the improvement of the performance of the TMGPFR model is too trivial with a larger G. So, we also set $G = 2$ as in Section V-A. As shown in Fig. 2(b), we can find ten sample curves in the original data set, which are quite noisy and overlapped, and therefore difficult for clustering analysis. In this sense, we do not need to produce the noisy data set from it. But for the other situations, the small and unbalanced data sets are also produced from the original data set as follows.

TABLE IV

PARAMETERS OF THE MIX-GGPFR MODEL FOR THE ORIGINAL DATA SET

	π_k	$\theta_k^{(1)}$	$\theta_k^{(2)}$	$\lambda_k^{(1)}$	$\lambda_k^{(2)}$
$k = 1$	0.5000	0.2000	1.0000	0.1000	0.0000
$k = 2$	0.5000	0.1414	0.7071	-0.1000	0.9000

1) *The Small Data Set:* All the numbers of training and test samples are the same as those of small data set in Section V-A.
2) *The Unbalanced Data Set:* $\pi_1 = 0.2500$ and $\pi_2 = 0.7500$.

The prediction and curve clustering results on these data sets are listed in Tables V and VI, respectively. It is clear that the TMGPFR model is the best on both prediction and curve clustering. Moreover, the TMGPFR model can cluster all these complex curves correctly or almost correctly on each of the three data sets. Theoretically, the time complexities of the learning algorithms for the GP, GPFR, mix-GP, and mix-GPFR models are $\mathcal{O}[M(\max N_m)^3]$, while the time complexities of the MCMC EM algorithms for the TMGPFR, MGPFR, and MGP models are $\mathcal{O}[M(\max N_m)^4]$. Therefore, the MCMC EM algorithms for the TMGPFR, MGPFR, and MGP models

TABLE V

AVERAGE PREDICTED RMSEs AND AVERAGE COMPUTING TIME (MINUTE) OF THE SEVEN MODELS WITH SDs OVER 30 TRIALS ON THE ORIGINAL, SMALL, AND UNBALANCED DATA SETS GENERATED FROM THE MIX-GGPFR MODELS. THE BEST RESULTS ARE SHOWN IN BOLD FONT

Model	Original		Small		Unbalanced	
	RMSE	Time	RMSE	Time	RMSE	Time
TMGPFR	**0.5348 ± 0.0029**	233.3 ± 20.15	**0.5463 ± 0.0066**	40.55 ± 5.263	**0.5335 ± 0.0029**	310.7 ± 25.16
MGPFR	0.5426 ± 0.0031	74.18 ± 2.744	0.5523 ± 0.0061	8.928 ± 0.143	0.5396 ± 0.0030	54.66 ± 4.859
mix-GPFR	0.5398 ± 0.0036	3.390 ± 0.290	0.5581 ± 0.0077	0.723 ± 0.095	0.5388 ± 0.0028	3.817 ± 0.293
MGP	0.5537 ± 0.0036	57.78 ± 2.141	0.5691 ± 0.0086	6.294 ± 0.333	0.5519 ± 0.0033	37.80 ± 1.784
mix-GP	0.5561 ± 0.0043	2.605 ± 0.235	0.5879 ± 0.0150	0.429 ± 0.043	0.5524 ± 0.0026	2.974 ± 0.196
GPFR	0.5416 ± 0.0023	0.688 ± 0.046	0.5523 ± 0.0065	0.114 ± 0.014	0.5387 ± 0.0025	0.699 ± 0.098
GP	0.5569 ± 0.0124	0.288 ± 0.096	0.5795 ± 0.0165	0.034 ± 0.009	0.5519 ± 0.0027	0.261 ± 0.092

TABLE VI

AVERAGE CURVE CARs OF THE TMGPFR, MIX-GPFR, AND MIX-GP MODELS OVER 30 TRIALS ON THE ORIGINAL, SMALL, AND UNBALANCED DATA SETS GENERATED FROM THE MIX-GGPFR MODELS. THE BEST RESULTS ARE SHOWN IN BOLD FONT

Model	Original	Small	Unbalanced
TMGPFR	**100.0%**	**99.88%**	**100.0%**
mix-GPFR	84.00%	73.78%	79.80%
mix-GP	58.36%	65.86%	56.57%

become rather slower as the numbers of samples in the sample curves become larger, which is actually demonstrated by the recorded time consumptions on the same desktop computer running MATLAB 2016b listed in Tables V and VII. However, the MCMC EM algorithm for the TMGPFR model can be implemented efficiently on these synthetic data sets with hundreds of sample curves. Fortunately, the computing time of the MCMC EM algorithm can be overcome with the development of computing power. In many practical applications, the accuracies of the prediction and curve clustering are much more important than the computing time complexity.

C. On Electrical Load Prediction

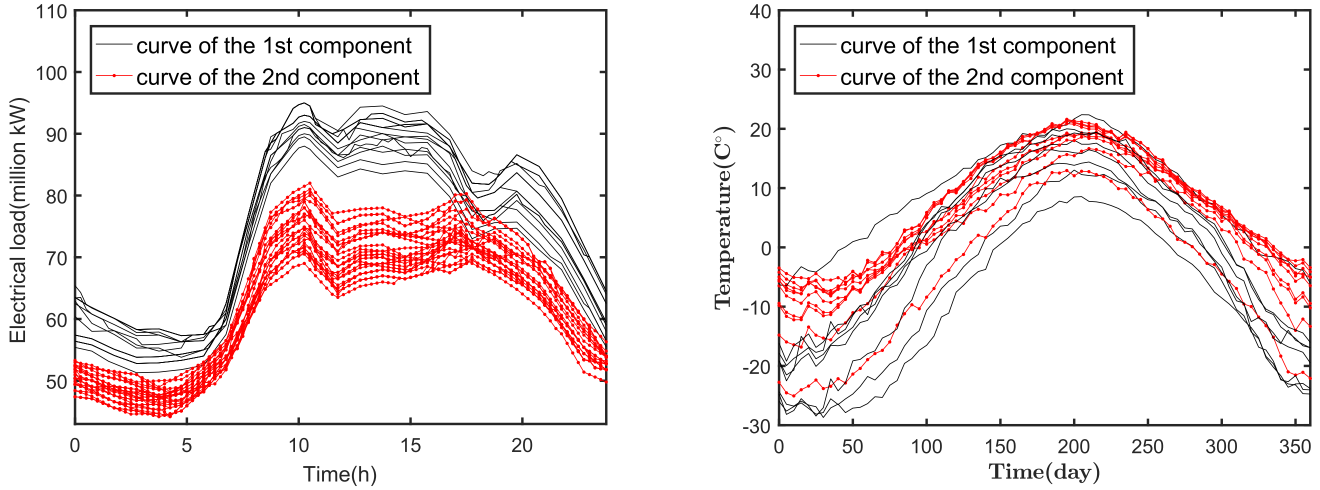
Furthermore, we apply the TMGPFR model to solving the problem of electrical load prediction, which plays a vital role in optimal unit commitment, startup and shut-down of thermal plants, and control of reserve and exchanging electric power in interconnected systems [38]. The electrical load (million kW) data set was issued by the Northwest China Grid Company. In this data set, there are 100 curves, which all consist of 96 samples or points. In fact, these curves were the observations of electrical load for 100 days, respectively, and through each day, the electrical load was recorded every quarter hour so that there are 96 (24 × 4) samples in each curve. In this situation, we use 50 curves for training and 50 curves for testing. In each test curve, there are 48 known samples and 48 predictive samples.

For comparison, we apply the TMGPFR model and six other possible models to the electrical load prediction task. The average predicted RMSEs and average time consumptions of the TMGPFR and comparative models are listed in Table VII. We use a twofold cross validation procedure for model selection [39] on the training data set with various choices $[K, G, D]$ and select the optimal values of $[K, G, D]$ in Table VII. It is found that the average predicted RMSE of the TMGPFR model is the smallest among all the models and its SD is also small. Therefore, the TMGPFR model is the most effective model for this electrical load prediction task. As the average predicted RMSEs of the MGPFR, mix-GPFR, and GPFR models are much smaller than those of the MGP, mix-GP, and GP models, the mean functions of GPs are very important for this practical application. As shown in Fig. 5(a), according to the resulted TMGPFR model, 50 training curves are clustered into two classes corresponding to the two components, being illustrated by two curve styles, respectively, which is very reasonable in interpretation.

D. On Weather Prediction

Finally, we apply the TMGPFR model to solve the weather prediction problem with a real-world data set that recorded daily temperature (°C) averages over the year from 1961 to 1994 in 35 Canadian weather stations [40]. In fact, there are 35 curves in the data set and each curve consists of the observations of a weather station in Canada as its samples. In fact, 73 (365/5) samples in each curve were the mean temperatures of every five days within one year. We use 18 curves for training and 17 curves for testing. In each test curve, there are 37 known samples and 36 predictive samples.

In a similar manner, we apply the TMGPFR model and six other possible models to solve the weather prediction problem. The prediction RMSEs and time consumptions of the TMGPFR and comparative models are also listed in Table VII. We use a twofold cross validation procedure for model selection. It is found that the TMGPFR model is the most effective model for this weather prediction task. However, the MGPFR model does not perform well on this data set and it is even worse than the mix-GFFR, mix-GP, and GP models in this situation. As shown in Fig. 5(b), the 18 training curves are clustered into two classes by the resulted TMGPFR model. It appears that the curves of two classes are just the observations of the stations in the south and north of Canada, respectively, which also demonstrates that



(a)

(b)

Fig. 5. (a) Two clusters of the resulted TMGPFR model on the training curves of electrical load data set. (b) Two clusters of the resulted TMGPFR model on the training curves of weather data set.

TABLE VII

Average Predicted RMSEs and Average Computing Time (Minute) of Seven Models With SDs Over 30 Trials on the Electrical Load Data Set and the Weather Data Set. Algorithms for Related Models Are the Same as Those in Table II. The Best Results Are Shown in Bold Font

Model	Electrical load					Weather				
	K	*G*	*D*	RMSE	Time	*K*	*G*	*D*	RMSE	Time
TMGPFR	2	2	62	**0.5191 ± 0.1484**	274.2 ± 141.2	2	2	62	**0.6168 ± 0.0149**	33.99 ± 0.969
MGPFR	-	2	32	0.5868 ± 0.1253	10.18 ± 2.132	-	3	32	0.8519 ± 0.0792	3.288 ± 0.187
mix-GPFR	3	-	32	1.0109 ± 0.5783	9.165 ± 1.414	3	-	62	0.6660 ± 0.0106	1.714 ± 0.110
MGP	-	4	-	27.500 ± 30.786	88.82 ± 38.13	-	5	-	0.8934 ± 0.0139	27.36 ± 5.067
mix-GP	4	-	-	1.0606 ± 1.1509	7.200 ± 1.297	2	-	-	0.7636 ± 0.0001	0.435 ± 0.063
GPFR	-	-	62	3.5640 ± 2.7995	0.452 ± 0.207	-	-	62	2.4777 ± 3.0918	0.151 ± 0.044
GP	-	-	-	23.911 ± 31.688	0.369 ± 0.396	-	-	-	0.7646 ± 0.0001	0.093 ± 0.065

the structure discovered by the TMGPFR model is reasonable for interpretation.

VI. Discussion and Conclusion

In this paper, we have proposed the TMGPFR model for a very general and complex class of data set, where some independent stochastic processes are linearly mixed together. In the lower layer of the TMGPFR model, we use a number of mixtures of Gaussian process functional regressions (MGPFRs) as the processing units to characterize the same number of general stochastic processes in an independent manner. In the higher layer, we use a mixture of MGPFRs to describe these linearly mixed stochastic processes structurally. Therefore, the TMGPFR model can deal with the more complex data sets involved in many practical applications, especially in curve clustering and prediction. For its parameter learning, the MCMC EM algorithm is developed. It is demonstrated by computer simulations that the TMGPFR model estimated with the MCMC EM algorithm outperforms the conventional mixture models in curve clustering and prediction. Finally, the TMGPFR model is successfully applied to two real world problems of the electrical load prediction and the weather prediction.

Thus far, we have assumed that the model size (i.e., K and G_k) of the TMGPFR model is known or easily identifiable from cross validation. Development of automatic model selection methods for TMGPFRs remains an important research topic in the future. Furthermore, the dimension of the input in the TMGPFR model is only one. However, it is rather straightforward to adapt the TMGPFR model for high-dimensional input data.

Acknowledgment

The authors thank the anonymous referees for their valuable comments and suggestions.

References

[1] C. E. Rasmussen and C. K. I. Williams, *Gaussian Processes for Machine Learning*. Cambridge, MA, USA: MIT Press, 2006, ch. 2.

[2] R. C. Grande, T. J. Walsh, G. Chowdhary, S. Ferguson, and J. P. How, "Online regression for data with changepoints using Gaussian processes and reusable models," *IEEE Trans. Neural Netw. Learn. Syst.*, vol. 28, no. 9, pp. 2115–2128, Sep. 2016.

[3] M. Lazaro-Gredilla and S. Van Vaerenbergh, "A Gaussian process model for data association and a semidefinite programming solution," *IEEE Trans. Neural Netw. Learn. Syst.*, vol. 25, no. 11, pp. 1967–1979, Nov. 2014.

[4] L. Munoz-Gonzalez, M. Lazaro-Gredilla, and A. R. Figueiras-Vidal, "Divisive Gaussian processes for nonstationary regression," *IEEE Trans. Neural Netw. Learn. Syst.*, vol. 25, no. 11, pp. 1991–2003, Nov. 2014.

[5] V. Tresp, "Mixtures of Gaussian processes," in *Proc. Adv. Neural Inf. Process. Syst. (NIPS)*, vol. 13. 2000, pp. 654–660.

[6] L. Xu, M. I. Jordan, and G. E. Hinton, "An alternative model for mixtures of experts," in *Proc. Adv. Neural Inf. Process. Syst. (NIPS)*, vol. 7. 1995, pp. 633–640.

[7] Y. Yang and J. Ma, "Asymptotic convergence properties of the EM algorithm for mixture of experts," *Neural Comput.*, vol. 23, no. 8, pp. 2140–2168, Aug. 2011.

[8] P. Chen, N. Zabaras, and I. Bilionis, "Uncertainty propagation using infinite mixture of Gaussian processes and variational Bayesian inference," *J. Comput. Phys.*, vol. 284, pp. 291–333, Mar. 2015.

[9] W. Fan, H. Sallay, and N. Bouguila, "Online learning of hierarchical Pitman–Yor process mixture of generalized Dirichlet distributions with feature selection," *IEEE Trans. Neural Netw. Learn. Syst.*, vol. 28, no. 9, pp. 2048–2061, Jun. 2016.

[10] E. Meeds and S. Osindero, "An alternative infinite mixture of Gaussian process experts," in *Proc. Adv. Neural Inf. Process. Syst. (NIPS)*, vol. 18. 2006, pp. 883–890.

[11] C. E. Rasmussen and Z. Ghahramani, "Infinite mixtures of Gaussian process experts," in *Proc. Adv. Neural Inf. Process. Syst. (NIPS)*, vol. 2. 2002, pp. 881–888.

[12] A. Tayal, P. Poupart, and Y. Li, "Hierarchical double Dirichlet process mixture of Gaussian processes," in *Proc. Assoc. Adv. Artif. Intell. (AAAI)*, Jul. 2012, pp. 1126–1133.

[13] S. Sun, "Infinite mixtures of multivariate Gaussian processes," in *Proc. Int. Conf. Mach. Learn. Cybern. (ICMLC)*, Jul. 2013, pp. 1011–1016.

[14] J. Ma, L. Xu, and M. I. Jordan, "Asymptotic convergence rate of the EM algorithm for Gaussian mixtures," *Neural Comput.*, vol. 12, no. 12, pp. 2881–2907, Dec. 2000.

[15] A. P. Dempster, N. M. Laird, and D. B. Rubin, "Maximum likelihood from incomplete data via the EM algorithm," *J. Roy. Stat. Soc., B, Methodol.*, vol. 39, no. 1, pp. 1–38, 1977.

[16] S. Sun and X. Xu, "Variational inference for infinite mixtures of Gaussian processes with applications to traffic flow prediction," *IEEE Trans. Intell. Transp. Syst.*, vol. 12, no. 2, pp. 466–475, Jun. 2011.

[17] T. V. Nguyen and E. V. Bonilla, "Fast allocation of Gaussian process experts," in *Proc. 31st Int. Conf. Mach. Learn. (ICML)*, Jan. 2014, pp. 145–153.

[18] C. Yuan and C. Neubauer, "Variational mixture of Gaussian process experts," in *Proc. Adv. Neural Inf. Process. Syst. (NIPS)*, vol. 21. 2008, pp. 1897–1904.

[19] Y. Yang and J. Ma, "An efficient EM approach to parameter learning of the mixture of Gaussian processes," in *Advances in Neural Networks—ISNN* (Lecture Notes in Computer Science), vol. 6676. Berlin, Germany: Springer, May 2011, pp. 165–174.

[20] Z. Chen, J. Ma, and Y. Zhou, "A precise hard-cut EM algorithm for mixtures of Gaussian processes," in *Proc. 10th Int. Conf. Intell. Comput. (ICIC)*, vol. 8589. Aug. 2014, pp. 68–75.

[21] Z. Chen and J. Ma, "The hard-cut EM algorithm for mixture of sparse Gaussian processes," in *Proc. 11th Int. Conf. Intell. Comput. (ICIC)*, vol. 9227. Aug. 2015, pp. 13–24.

[22] L. Li, P. Wang, K.-H. Chao, Y. Zhou, and Y. Xie, "Remaining useful life prediction for lithium-ion batteries based on Gaussian processes mixture," *PLoS ONE*, vol. 11, no. 9, p. e0163004, Sep. 2016.

[23] D. Wu, Z. Chen, and J. Ma, "An MCMC based EM algorithm for mixtures of Gaussian processes," in *Advances in Neural Networks—ISNN* (Lecture Notes in Computer Science), vol. 9377. Cham, Switzerland: Springer, Oct. 2015, pp. 327–334.

[24] G. C. G. Wei and M. A. Tanner, "A Monte Carlo implementation of the EM algorithm and the poor man's data augmentation algorithms," *J. Amer. Stat. Assoc.*, vol. 85, pp. 699–704, Sep. 1990.

[25] J. Q. Shi, B. Wang, R. Murray-Smith, and D. M. Titterington, "Gaussian process functional regression modeling for batch data," *Biometrics*, vol. 63, pp. 714–723, Sep. 2007.

[26] J. Q. Shi and B. Wang, "Curve prediction and clustering with mixtures of Gaussian process functional regression models," *Stat. Comput.*, vol. 18, pp. 267–283, Sep. 2008.

[27] D. Wu and J. Ma, "A DAEM algorithm for mixtures of Gaussian process functional regressions," in *Proc. 12th Int. Conf. Intell. Comput. (ICIC)*, vol. 9773. Jul. 2016, pp. 294–303.

[28] R. Kamnik, J. Q. Shi, R. Murray-Smith, and T. Bajd, "Nonlinear modeling of FES-supported standing-up in paraplegia for selection of feedback sensors," *IEEE Trans. Neural Syst. Rehabil. Eng.*, vol. 13, no. 1, pp. 40–52, Mar. 2005.

[29] Z. Qiang and J. Ma, "Automatic model selection of the mixtures of Gaussian processes for regression," in *Advances in Neural Networks—ISNN* (Lecture Notes in Computer Science), vol. 9377. Cham, Switzerland: Springer, Oct. 2015, pp. 335–344.

[30] C. de Boor, "On calculating with B-splines," *J. Approx. Theory*, vol. 6, no. 1, pp. 50–62, 1972.

[31] C. E. Rasmussen and H. Nickisch, "Gaussian processes for machine learning (GPML) toolbox," *J. Mach. Learn. Res.*, vol. 11, pp. 3011–3015, Nov. 2010.

[32] J.-L. Wang, J.-M. Chiou, and H.-G. Müller, "Functional data analysis," *Annual Rev. Stat. Appl.*, vol. 3, pp. 257–295, Jun. 2016.

[33] Y. M. Cheung, "Maximum weighted likelihood via rival penalized EM for density mixture clustering with automatic model selection," *IEEE Trans. Knowl. Data Eng.*, vol. 17, no. 6, pp. 750–761, Jun. 2005.

[34] J. Ma and T. Wang, "A cost-function approach to rival penalized competitive learning (RPCL)," *IEEE Trans. Syst., Man, Cybern., B, Cybern.*, vol. 36, no. 4, pp. 722–737, Aug. 2006.

[35] J. Ma and J. Liu, "The BYY annealing learning algorithm for Gaussian mixture with automated model selection," *Pattern Recognit.*, vol. 40, no. 7, pp. 2029–2037, Jul. 2007.

[36] H. Jia, Y.-M. Cheung, and J. Liu, "Cooperative and penalized competitive learning with application to kernel-based clustering," *Pattern Recognit.*, vol. 47, pp. 3060–3069, Sep. 2014.

[37] L. Zhao and J. Ma, "A dynamic model selection algorithm for mixtures of Gaussian processes," in *Proc. IEEE 13th Int. Conf. Signal Process. (ICSP)*, Nov. 2016, pp. 1095–1099.

[38] M. Mohandes, "Support vector machines for short-term electrical load forecasting," *Int. J. Energy Res.*, vol. 26, pp. 335–345, Mar. 2002.

[39] S. Arlot and A. Celisse, "A survey of cross-validation procedures for model selection," *Stat. Surv.*, vol. 4, pp. 40–79, 2010, doi: 10.1214/09-SS054.

[40] S. López-Pintado and J. Romo, "On the concept of depth for functional data," *J. Amer. Stat. Assoc.*, vol. 104, no. 486, pp. 718–734, 2009.

Di Wu received the B.S. degree in mathematics from Shaanxi Normal University, Xi'an, China, in 2012. He is currently pursuing the Ph.D. degree in applied mathematics with the School of Mathematical Sciences, Peking University, Beijing, China.

His current research interests include machine learning, data mining, and neural networks.

Jinwen Ma received the M.S. degree in applied mathematics from Xi'an Jiaotong University, Xi'an, China, in 1988, and the Ph.D. degree in probability theory and statistics from Nankai University, Tianjin, China, in 1992.

From 1992 to 1999, he was a Lecturer or an Associate Professor with the Department of Mathematics, Shantou University, Shantou, China, where he became a Full Professor with the Institute of Mathematics in 1999. In 2001, he joined the Department of Information Science, School of Mathematical Sciences, Peking University, Beijing, China, where he is currently a Full Professor and a Ph.D. Tutor. From 1995 to 2003, he also visited several times at the Department of Computer Science and Engineering, The Chinese University of Hong Kong, Hong Kong, as a Research Associate or a fellow. He was also a Research Scientist with the Amari Research Unit, RIKEN Brain Science Institute, Wako, Japan, from 2005 to 2006. He has authored over 100 academic papers on neural networks, pattern recognition, bioinformatics, and information theory.