A Nonparametric Bayesian Clustering to
Discover Latent Covariance Structure of Multiple Time Series

Anh Tong \textsuperscript{1} Jaesik Choi \textsuperscript{1}

Abstract
Analyzing time series data is important to predict future events and changes in finance, manufacturing and administrative decisions. Gaussian processes (GPs) solve regression and classification problems by choosing appropriate kernels capturing covariance structure of data. In time series analysis, GP based regression methods recently demonstrate competitive performance by decomposing temporal covariance structure. Such covariance structure decomposition allows exploiting shared parameters over a set of multiple but selected time series. In this paper, we propose an efficient variational inference algorithm for nonparametric clustering over multiple GP covariance structures. We handle multiple time series by placing an Indian Buffet Process (IBP) prior on the presence of the additive shared kernels. We propose a new variational inference algorithm to learn the nonparametric Bayesian models for the clustering and regression problems. Experiments are conducted on both synthetic data sets and real world data sets, showing promising results in term of structure discoveries. In addition, our model learns GP kernels faster but still preserves a good predictive performance.

1. Introduction
Time series data analysis is important to predict future events in various application domains including finance, manufacturing and administrative decision. Gaussian processes (GPs) solve regression and classification problems for smooth functions by choosing appropriate kernels capturing covariance structure. In time series analysis, identifying temporal covariance can reveal the underlying structure to explain changes of time series data.

In GP regression, input time series data are modeled by the ensemble of GP predictors with heterogeneous kernels (Minka & Picard). Recently, it has been shown that a kernel composition grammar can dramatically expand the set of possible GP kernels by summing and multiplying two or more kernels (Grosse et al., 2012). A kernel structure learning framework called Automatic Bayesian Covariance Discovery solves the regression problem for various real-world data and generates human-friendly reports to explain the kernel structure (Lloyd et al., 2014).

The ABCD framework solves the regression problem in the following steps: (1) enumerating possible kernel structures by expanding current best kernel; (2) evaluating each kernel structure to find best model parameters based on certain criteria; (3) generating an automated report by explaining each composite kernel with a human-readable format. For a time series, ABCD searches candidate models in a greedy manner which can roughly be formed a tree-search.

Recently, a relational version of the ABCD system called Relational ABCD (R-ABCD) (Hwang et al., 2016) solves the multivariate regression problem by exploiting a shared kernel structure over all time series. The R-ABCD assumes that there is a global kernel to explain multiple time series. However, a single global kernel structure may not represent current and future changes covering all time series. Thus, the kernel structure search for the multivariate regression problem would be very expensive when individual structures are sought separately, in general cases.

In this paper, we solve the multivariate regression problem with nonparametric Bayesian clustering and regression models. We assume that the some (latent) composite kernels are shared by a subset of time series instead of all time series. That is, we focus on seeking interpretable components over subset of data with Indian Buffet process (IBP) (Griffiths & Ghahramani, 2005). In order to represent the association of data and each composite kernel, a binary latent feature is introduced where its value expresses the presence or absence of a component in data. By considering multiple data, the problem turns out to find a binary latent matrix which contents the information of whether they share the same component (parameter sharing).

In the last decade, multi-task learning for GP regression has
been studied extensively (Bonilla et al., 2007). There exists a large body of work in nonparametric Bayesian clustering methods (Griffiths & Ghahramani, 2005; 2011) to discover latent structure. However, nonparametric clustering for GP regression with kernel decomposition is not clearly addressed yet in the following reasons. First, most of existing work assumes that a GP kernel structure is fixed or given. Second, deriving an inference algorithm with nonparametric cluster for GP (nonparametric regression) is not trivial since it requires complex interactions among model parameters.

In this paper, we propose a nonparametric Bayesian latent kernel decomposition (IBP-GP) model. Contributions of this paper are following three folds:

- We present the first approach to solve the GP regression over multiple times series by leveraging kernel decomposition and shared kernel discovery with IBP.
- We provide an efficient variational inference algorithm for IBP-GP models.
- With the IBP-GP model and variational inference algorithm, the new ABCD system extends the scalability and usability of automated GP regression framework.

This paper is structured as following. Section 2 explains existing ABCD and IBP models. Section 3 introduces the IBP-GP model and explains detail derivations of the variational inference algorithm. Section 4 reviews the work in the literature. Section 5 presents experimental results in real-world data sets followed by conclusion in Section 7. Related works and conclude our work.

2. Background

In this section, we provide a preliminary background about Gaussian Process which leads to the construction of Automatic Bayesian Discovery Covariance (ABCD) framework (Lloyd et al., 2014). Then we take a brief review about the Indian Buffet Process (IBP) (Griffiths & Ghahramani, 2005).

2.1. Automatic Bayesian Discovery Covariance

The ABCD framework is beautifully designed, composing from several essential parts e.g. a language of models, a search procedure among models, and a model evaluation. The found model can be used in various of purposes e.g. making prediction, model checking, generating description. The framework makes use of Gaussian Process which is well-studied in (Rasmussen & Williams, 2005) to perform a regression analysis.

A Gaussian Process (GP) formally is defined as a collection of (possibly infinite) random variables, any finite of which have a joint multivariate Gaussian distribution. A GP is denoted as \( f \sim GP(m, k) \), where \( m \) is the mean function, usually chosen as a zero function, \( k \) is the covariance function or kernel function.

Selecting kernel functions plays a crucial role in learning GP and previously requires expert knowledge. The ABCD manages to have a search procedure for kernel function in lieu of manually selecting a kernel based on experience. This search procedure basically relies on a language of models or, in another words, a language of kernels which is constituted from a grammar and base kernels. The base kernels reveal the high-level properties of data. They can be listed as: white noise (WN) explaining uncorrelated noise; constant (C) representing constant functions; linear (LIN) encoding linear functions, squared exponential (SE) ciphering smoothness, periodic (PER) describing periodic functions (see Appendix for details). The grammar enables us to explore and generate new kernels from base ones based on composition rules under some conditions:

\[
(k_1 + k_2)(\cdot, \cdot) = k_1(\cdot, \cdot) + k_2(\cdot, \cdot),
\]

\[
(k_1 \times k_2)(\cdot, \cdot) = k_1(\cdot, \cdot) \times k_2(\cdot, \cdot).
\]

These composition rules on base kernels were first introduced in (Duvenaud et al., 2013). The ABCD framework inherited this idea and extended it with changepoint (CP) operators and changewindows (CW) operators to capture sudden changes in time series

\[
CP(k_1, k_2) = k_1 \times \sigma + k_2 \times \bar{\sigma},
\]

\[
CW(k_1, k_2) = k_1 \times \sigma_1 \times \bar{\sigma}_2 + k_2 \times \sigma_1 \times \bar{\sigma}_2,
\]

where \( \sigma = \sigma(x)\sigma(x') \) and \( \bar{\sigma} = (1-\sigma(x))(1-\sigma(x')) \), analogously \( \sigma_1, \sigma_2, \bar{\sigma}_1, \bar{\sigma}_2 \) are defined in the same manner.

The kernels produced from such a language of kernels not only have richer structures to be expressive enough to fit data but also provide meaningful encoding information e.g. (SE + LIN) showing a smoothness with a linear trend. As the result of the grammar construction, all kernels are in the form of sum of following products:

\[
K \prod_m \text{LIN}^{(m)} \prod_n \sigma^{(n)},
\]

where \( K \) could either be absent or be one of WN, C, SE, PER, SE \( \times \prod_k \text{PER}^{(k)} \), the subscript indicates the kernels have different hyperparameters.

By having a language of models, the framework searches for the best fit kernel via building a search tree. Starting from a certain base kernel, the candidate kernels at the same tree depth are generated by the grammar. All of the
One of the most precious parts from the ABCD framework is its ability to express the models in the form of natural-language description. We hardly find this feature as an inherent characteristic in other machine learning methods. (Lloyd et al., 2014) details the key for the ABCD framework to unlock this feature by forming the best fit kernels into the sum of products of kernels where each product of kernels is converted into a noun phrase.

(Hwang et al., 2016) proposed a relational version of ABCD which works with strongly correlated multiple time series having the assumption that those share the same kernel structure. Because of that, it avoided a huge computational cost since it finds a shared structure using the ABCD’s search procedure. This model showed improvements both qualitatively and quantitatively when the sequences have a certain correlation. However, for the general cases, given multiple data may partially share some information, even they may not share any at all. This paper attempts to tackle this situation by borrowing a nonparametric method reviewed in the next subsection.

### 2.2. Indian Buffet Process

The Indian Buffet Process (IBP) is an unsupervised learning methods proposed in (Griffiths & Ghahramani, 2005) to discover latent features among a set of objects. This nonparametric process defines a distribution over a binary matrix $Z$ with finite rows and infinite column. The matrix indicates feature assignments where the element at $i$-th row and $j$-th column expresses the presence or absence of $j$-th feature in object $i$-th. The IBP latent feature model is the generalization of Dirichlet Process which works on a cluster-assignment binary matrix having the summation of each row equals to 1.

The IBP prior over the matrix $Z$ is well-established in (Griffiths & Ghahramani, 2005), having a culinary metaphor about how customers choose dishes in an Indian buffet restaurant. Each row of $Z$ is considered as a customer or observation. Each column of $Z$ describes dishes or features where the number of them can be infinite. The first customer comes and takes the first Possion($\alpha$) dishes. With probability $m_k/n$, the next customers choose the dishes which are tried by previous customers, where $m_k$ is the number of customers who tried dish $k$ before customer $n$. Beside trying previously sampled dishes, each customer will try Poisson($\alpha/n$) new dishes.

Performing “left-order-form” (lof) on $Z$ will create equivalence classes [$Z$]. Thanks to the the exchangeability properties of the IBP, it is able to define a distribution over [$Z$] as

$$p([Z]) = \frac{\alpha^{K_+}}{\prod_{h=1}^{N-1} K_h!} \cdot e^{-\alpha H_N} \prod_{k=1}^{K_+} (N - m_k)! (m_k - 1)$$

where $\alpha$ is the concentration hyperparameter, $H_N$ is the harmonic number, $H_N = \sum_{i=1}^{N} \frac{1}{i}$, $K_+$ is the number of nonzeros columns, and $m_k = \sum_{n=1}^{N} z_{nk}$ is the number of customers using dish $k$.

One of typical applications of IBP is the linear-Gaussian latent feature model. Data represented by $X$ is factorized into an IBP latent matrix $Z$ multiplying with a feature matrix $A$ plus a noise matrix $\mathcal{E}$.

$$X = ZA + \mathcal{E}.$$  \hfill (2)

### 3. Latent kernel decomposition model

Given $N$ time series, each time series has $D$ data points. Let denote $x_n$ be the $n$-th time series, specifically $x_n = [x_{n1}, ..., x_{nD}]$ where $x_{ni}$ is the data point of the $n$-th time series at the $d$-th time step $t_d$. All time series are accumulated in a single $N \times D$ matrix $X$. Our aim is to decompose
$x_n$. into additive components

$$x_n = \sum_{k=1}^{K} z_{nk} f_k + \epsilon, \quad (3)$$

where $z_{nk} \in \{0,1\}$ is the element of IBP matrix indicating whether time series $x_n$ has component $f_k$; $\epsilon$ is a row vector having each elements independently follows a Gaussian noise $N(0, \sigma^2_X)$; for any $k = 1 \ldots K$, $f_k$ is a GP with a covariance kernel $c_k(\cdot, \cdot; \theta_k)$ which is parameterized by its hyperparameters $\theta_k$

$$f_k \sim GP(0, c_k(\cdot, \cdot; \theta_k)).$$

Here we use the GP notation abusively for $f_k$ which actually finite dimensional. When stacking $f_k$ into $K \times D$ matrix $F$, our model described in (3) can be written as

$$X = ZF + \epsilon. \quad (4)$$

Here we define matrix $Z$ with finite rows and possibly infinite columns in which its elements are $z_{nk}$

$$Z = \begin{bmatrix} z_{11} & z_{12} & \cdots \\ \vdots & \ddots & \vdots \\ z_{N1} & z_{N2} & \cdots \end{bmatrix}.$$  

In the aspect of multi-task learning, $f_k$ are shared among any arbitrary pair of time series when both time series posses $z_{nk}$ equal to 1. The process of discovering the $z_{nk}$ is automatic as a consequence of learning matrix $Z$ which we place the IBP prior on.

Although the our model represented in (4) looks similar to linear-Gaussian model in (2), the correlation among elements in the $k$-th row of $F$ described by $c_k$ distinguishes our model from linear-Gaussian model since there is no correlation between among elements in $A$. Figure 1 is a simple explanation for our model.

### 3.1. Likelihood $p(X|Z)$

It is essential to obtain the form of the likelihood of data given the IBP matrix $p(X|Z)$. Since the matrix $Z$ is imposed a IBP prior, the number of its columns assumingly can go to infinity or we may have infinite kernel components. We are going to find the answer for the question that what if such that case happens, the probability distribution $P(X|Z)$ is still valid. (Griffiths & Ghahramani, 2011) gave a detailed answer for this question on the linear-Gaussian model. How about the latent kernel decomposition model? The road to find the answer is simply to marginalize $F$ from the $p(X|Z, F)$. We will present it with details in the following.

$$\begin{bmatrix} f_1, & \cdots, & f_K \end{bmatrix} \begin{bmatrix} c_1^{-1} & 0 & 0 \\ 0 & \ddots & 0 \\ 0 & 0 & c_K^{-1} \end{bmatrix} \begin{bmatrix} f^T_1 \\ \vdots \\ f^T_K \end{bmatrix}$$

Figure 2. Illustration of representing $\sum_{k=1}^{K} f_k c_k^{-1} f_k^T$. The first row vector is the concatenation of $f_k$, being written as $\vartheta(F^T)^T$. The block matrix is a block diagonal matrix where the block at diagonal position are $c_k^{-1}$. The last column vector is the transpose of the first row vector.

The distribution $p(X|Z, F, \sigma_X)$ of $X$ given $Z, F$ and $\sigma_X$ is

$$(2\pi\sigma_X^2)^{-ND/2} \exp\left\{ -\frac{1}{2\sigma_X^2} \text{tr}((X-ZF)^T(X-ZF)) \right\}, \quad (5)$$

where $\text{tr}(\cdot)$ is the trace of matrix. Since each row $f_k$ in $F$ representing a feature is placed a GP prior with covariance kernel $c_k$

$$p(f_k|c_k) = (2\pi)^{-D/2} |c_k|^{-1/2} \exp(-\frac{1}{2} f_k c_k^{-1} f_k^T),$$

where $| \cdot |$ denotes the determinant of matrix. Since these features are independent, the prior distribution $p(F|c_1, \ldots, c_K)$ over $F$ is constituted from the product of $p(f_k|c_k)$

$$(2\pi)^{-KD/2} \prod_{k=1}^{K} |c_k|^{-1/2} \exp(-\frac{1}{2} \sum_{k=1}^{K} f_k c_k^{-1} f_k^T) \quad (6)$$

The exponentiated term in (6) can be written as

$$\sum_{k=1}^{K} f_k c_k^{-1} f_k = \vartheta(F^T)^T \bigoplus_{k=1}^{K} c_k^{-1} \vartheta(F^T), \quad (7)$$

where $\vartheta(\cdot)^T$ is the vectorization which transforms a matrix into a column vector by stacking the columns of a matrix, $\bigoplus$ is the the direct sum operation of matrix. This term can be illustrated in Figure 2.

We carefully investigate the exponential expression when multiplying equations (5) and (6). Our aim is to marginalize out $F$ to get the likelihood $p(X|Z) = \int p(X|Z, F)p(F)dF$. The exponentiated term in $p(X|Z)$ should be manipulated to have a quadratic form of $F$ to make the integration tractable. One direction is to convert the equation (7) to the form of trace of matrix $\text{tr}(\sum_{k=1}^{K} F c_k^{-1} F^T \Delta_k)$. However, there exists a dimensional disagreement between two matrices in trace where this matrix is a $K \times K$ matrix while the one in (5) is a matrix $D \times D$. Even performing the cyclic permutation in trace,
we still cannot get a good form to proceed the derivation. The more effective solution is to transform trace forms in (5) into vectorization. Some properties\(^2\)\(^3\)\(^4\) between trace and vectorization are applied to get the followings:

\[
\text{tr}(X^TZF) = \text{tr}(FX^TZ) = \vartheta(F)^T \vartheta(X^TZ),
\]

\[
\text{tr}(F^TZ^TX) = \text{tr}(Z^TXF^T) = \vartheta(X^TZ)^T \vartheta(F^T),
\]

where the invariance of cyclic permutation of trace and the conversion from trace to vectorization are used respectively;

\[
\text{tr}[(ZF)^T(ZF)] = \text{tr}[(ZF)(ZF)^T] = \vartheta(F^TZ)^T \vartheta(F^TZ) = \vartheta(F^T)^T(Z^TZ \otimes I_D) \vartheta(F^T),
\]

where the first two “=”s are done as the same as the previous derivation, then the vectorization is written in the form of Kronecker product. Note that \(I_D\) is the \(D \times D\) identity matrix, \(\otimes\) is the Kronecker product.

By using the above results and the technique of completing the square, we get the exponentiated term in \(p(X|Z, F)p(F)\)

\[
\text{tr}\left\{ \frac{1}{\sigma_X^2} (X - ZF)^T (X - ZF) \right\} + \sum_{k=1}^{K} f_k c_k^{-1} f_k^T
\]

\[
\frac{1}{\sigma_X} \text{tr}(X^TX) - \frac{1}{\sigma_X^2} \vartheta(X^TZ)^TG \vartheta(X^TZ)
\]

\[
+ (G \vartheta(X^TZ) - \vartheta(F^T))^T \frac{G^{-1}}{\sigma_X^2} (G \vartheta(X^TZ) - \vartheta(F^T))
\]

\[
\tag{8}
\]

Equations (5), (6) and (8) lead to the likelihood function by integrating out \(F\) since (b) in (9) makes the integration tractable; only (a) left in the likelihood function (see Appendix)

\[
\frac{(2\pi)^{-ND/2} \sigma_X^{K^D - N^D}}{\sqrt{\left| Z^T Z \otimes I_D + \sigma_X^2 \bigoplus_{k=1}^{K} c_k^{-1} \right| \prod_{k=1}^{K} |c_k|}} \exp\left\{ \frac{1}{2\sigma_X^2} \left[ -\text{tr}(X^TX) + \vartheta(X^TZ)^TG \vartheta(X^TZ) \right] \right\}
\]

\[
\tag{9}
\]

Theorem 1. The likelihood function in (9) is still well-defined when \(K\) goes to infinity.

\(^2\)Cyclic permutation: \(\text{tr}(ABC) = \text{tr}(BCA) = \text{tr}(CAB)\)

\(^3\)Trace to vectorization: \(\text{tr}(A^TB) = \vartheta(A)^T \vartheta(B)\)

\(^4\)Vectorization to Kronecker product: \(\vartheta(ABC) = (C^T \otimes A) \vartheta(B)\)

Proof. We are going to show that if the number of columns goes to infinity, the distribution in (9) is still well-defined. To validate this, we perform log on \(Z\). All nonzero features are accumulated on the left, denoted as \(Z_0\); the rest of columns are zero vectors, denoted as \(Z_+\) has \(K_+\) columns, \(Z_0\) has \(K_0\) columns and \(K = K_+ + K_0\). Note that \(K_+\) is finite. The determinant in (9) is written as

\[
\left| Z^T Z \otimes I_D + \sigma_X^2 \bigoplus_{k=1}^{K} c_k^{-1} \right|
\]

\[
= \left| \begin{bmatrix} Z_+^T Z_+ & 0 \\ 0 & 0 \end{bmatrix} \right| \otimes I_D + \sigma_X^2 \bigoplus_{k \in K_+} c_k^{-1} \bigoplus_{k \in K_0} c_k^{-1} \right| (10)
\]

The term \(\prod_{k \in K_0} c_k^{-1}\) will be canceled out since the existence of \(\prod_{k=1}^{K} c_k\) in the denominator of (9). The same can be done for \(\sigma_X^{2K_0D}\) by taking a root square first then eliminating by \(\sigma_X^{K+D}\). Only \(\prod_{k \in K_+} c_k\) and \(\sigma_X^{K+D}\) remains and are finite. Also, the term containing \(Z\) in the exponentiated expression of (9) is written as

\[
\vartheta(X^TZ)^TG \vartheta(X^TZ) = \vartheta(X^TZ)^T
\]

\[
+(Z_+^T Z_+ \otimes I_D + \sigma_X^2 \bigoplus_{k \in K_+} c_k^{-1} \vartheta(X^TZ_+)\)
\]

\[
\tag{11}
\]

is finite. \(\Box\)

3.2. Kernel components

The kernel component \(c_k\) is not explicitly defined so far. Depending on different settings, we can flexibly select various strategies for kernel components. It is noteworthy that one can create a set of target kernel components based on his belief or experience about a data set. For example, kernel components can be in a collection in (1) for a fix number of \(m\) and \(n\). Here we present two different strategies focusing on two aspects including interpretability and accuracy.

3.2.1. Searching for interpretable components

We design the multilevel search algorithm for our model. That is, we maintain a set of interpretable components \(S_i\) at search level \(i\) where the expressiveness of components increases along with the search level. Whenever the variational inference learns a IBP matrix \(Z\), we can exclude kernel components what make \(Z\) have zero columns. From the found kernel components, we perform a kernel expansion procedure. Since the additive operator is completely represented by the matrix multiplication of \(Z\) and its components, we rule out the kernel expansion for addition but keep other operators like \(S \rightarrow S \times B, B \rightarrow B'\)
3.2.2. SM kernel

A possible choice for $c_k$ is the component taking from spectral mixture (SM) kernel ((Wilson & Adams, 2013))

$$k(\tau) = \sum_{q=1}^{Q} \exp \{-2\pi^2 \tau v_q \} \cos(2\pi \tau \mu_q)$$

where we pick $c_k(x, x')$ as a component $\exp \{-2\pi^2 \tau v_q \} \cos(2\pi \tau \mu_q)$. As the nature of the nonparametric method, we do not have to pre-define the number of components $Q$ which coincides with the number of features $K$ in IBP, and automatically adapts when the more data comes.

3.3. Variational Inference

Since (Griffiths & Ghahramani, 2005) introduced IBP along with Gibbs sampling approach, there has been a large body of works on improving learning latent models (mainly focus on solving linear Gaussian model) include accelerated version of Gibbs sampling (Doshi-Velez & Ghahramani, 2009), variational inference (Doshi et al., 2009), stochastic variational inference (Shah et al., 2015), or submodular approach (Reed & Ghahramani, 2013). In previous section, we obtain the closed form of likelihood which basically can light the path for a Gibbs sampling algorithm. However, there are two inherent problems: one is that $G$ is a high-dimensional matrix causing expensive matrix inversions; another is that the involvement of GP hyperparameters requires gradient-based methods which are not compatible with sampling methods, not mentioned to the slow convergence of MCMC methods.

In this paper, we present a variational inference for the latent kernel model. We have a generative process for the data and latent variables

$$\pi_k \sim \text{Beta}(\alpha/K, 1), \quad z_{nk} \sim \text{Bernoulli}(\pi_k),$$
$$f_k \sim \mathcal{GP}(0, c_k), \quad x_n \sim \mathcal{N}(z_n, F, \sigma_X^2 I),$$

where $k \in \{1, ..., K\}$, $n \in \{1, ..., N\}$. The construction of IBP matrix is the finite variational approach and can be formulated for the infinite variational case as in (Doshi et al., 2009) using the stick-breaking construction. For this model, the joint distribution of data and all latent variables $p(\pi, Z, F, X)$ is

$$\prod_{k=1}^{K} \left( p(\pi_k)p(f_k) \prod_{n=1}^{N} p(z_{nk}|\pi_k) \right) \prod_{n=1}^{N} p(x_n | z_n, F)$$

The mean field approximation of the posterior distribution is factorized as

$$q(\pi, Z, F) = \prod_{k} q(\pi_k)q(z_{nk})q(F),$$

where $q_{\pi_k}(\pi_k)$, $q_{z_{nk}}(z_{nk})$ and $q_{f_k}(f_k)$ are respectively Beta($\alpha$, $\tau_k$; $\tau_k$, $\tau_k$), Bernoulli($z_{nk}$; $\nu_{nk}$), $\mathcal{N}(f_k; \phi_k, \Phi_k)$.

The variational method aims to solve the minimization problem of the KL divergence between $p$ and $q$ which is equivalent to maximize the evidence lower bound (ELBO) $\mathcal{L}[q]$,

$$\log p(X | \omega) \geq \mathbb{E}_{\pi, Z, F} [\log p(\pi, Z, F, X | \omega)] + H[q] \triangleq \mathcal{L}[q],$$

where $\mathbb{E}$ with subscripts indicates the expectation over the distribution of these subscripts, the set of parameters $\omega = \{\alpha, \tau_k, \sigma_X^2\}$, and $H$ is the entropy. Then the $\mathbb{E}_{\pi, Z, F} [\log p(\pi, Z, F, X | \theta)]$ is written as

$$\sum_{k=1}^{K} \mathbb{E}_{\pi} [\log p(\pi_k | \alpha)] + \sum_{k=1}^{K} \sum_{n=1}^{N} \mathbb{E}_{\pi, Z} [\log p(z_{nk} | \pi_k)]$$
$$+ \sum_{k=1}^{K} \mathbb{E}_{F} [\log p(f_k | c_k)] + \sum_{n=1}^{N} \mathbb{E}_{Z, F} [\log p(x_n | z_n, F, \sigma_X^2 I)].$$

Since the IBP-GP model does not let every element in $F$ freely follows an independent Gaussian noise like linear-Gaussian model, most of the terms except $\mathbb{E}_{F} [\log p(f_k | c_k)]$ can derived be similarly to (Doshi et al., 2009) and its technical report. For the term $\mathbb{E}_{F} [\log p(f_k | c_k)]$, the row of feature matrix $f_k$ is controlled by a GP characterized by $c_k$, and $f_k^T$ is approximated by a Gaussian random vector having a mean $\phi_k$ and a covariance $\Phi_k$, this term in ELBO is

$$\mathbb{E}_{F} [\log p(f_k | c_k)] = -\frac{1}{2} (\log 2\pi c_k + \text{tr} (c_k^{-1} \phi_k) + \phi_k^T c_k^{-1} \phi_k).$$

Note that the variational inference for IBP-GP model relies on not only the optimization procedures on parameters $\tau, \phi, \nu$ but also the GP hyperparameters. Suppose $\theta_{kj}$ is the $j$-th hyperparameter of the $k$-th kernel component $c_k$, the partial derivative of ELBO with respect to $\theta_{kj}$ is

$$\frac{\partial \mathcal{L}}{\partial \theta_{kj}} = -\frac{1}{2} \text{tr} \left( (\alpha + \beta + I) c_k^{-1} \frac{\partial c_k}{\partial \theta_{kj}} \right),$$

where $\alpha = c_k^{-1} \phi_k$ and $\beta = c_k^{-1} \phi_k^T \phi_k$. This part of algorithm is the most computationally expensive computation. However, it can be implemented in a parallel manner after obtaining approximation of $\phi_k$ and $\Phi_k$.

Parameter updates are performed using the exponential family updates (Wainwright & Jordan, 2008) respectively.
on parameter $\tau, \nu$, and $\mu$ (see (Doshi et al., 2009)). For the case of $\phi$, we will try to maximize the following function of $\phi_k$

$$
\log g_{\phi_k}(f_k) = -\frac{1}{2} f_k \left( c_k^{-1} + \sum_{n=1}^{N} \frac{\nu n k}{\sigma X} I \right) f_k^T + f_k \left( \sum_{n=1}^{N} \frac{\nu n k}{\sigma X} (x_n - \sum_{l \neq k} \nu n l \phi_l) \right)^T + c,
$$

(14)

where $c$ is constant. From (14), the closed form solution is derived as $\bar{\phi}_k = \left( \sum_{n=1}^{N} \frac{\nu n k}{\sigma X} (x_n - \sum_{l \neq k} \nu n l \phi_l) \right) \Phi_k$, where $\Phi_k$ is updated first as $(c_k^{-1} + \sum_{n=1}^{N} \frac{\nu n k}{\sigma X} I)^{-1}$.

4. Related works

(Schaechtle et al., 2015) introduced stochastic grammar for ABCD by randomly pick a kernel. Finally, the best of kernel is selected based on the likelihood of that kernel comparing to others’. It provides sampling method base Venture probabilistic programming language, then there is no exact form of likelihood function. Moreover, it only applies single time series. Our model only has a tractable derivation but also works on multiple time series.

In the multiple time series setting, (Hwang et al., 2016) introduced a shared kernel approach to capture general information among multiple sequences. The data either should have a strong correlation or is processed by a clustering algorithm to satisfy the model assumption. Here our model discovers clusters and learning shared structures simultaneously.

(Fox et al., 2014) proposed a model to learn dynamical behaviors shared among sequences by using Beta process to learn behavior feature matrix. Our model can be apply for this application by selecting appropriate kernel components. The sequences will be modeled by a collections of windows of kernel. By observing the presence and absence of windows, we can say about the transition of behaviors between these windows.

5. Experiments

In this section, we provide the experiment results on a synthetic data set and three real-world data sets.

5.1. Synthetic data

We generate synthetic three time-series according to several chosen kernels to verify our kernel selection algorithm. Each time series consists of 200 data points, generated from different kernels (see Table 1).

| Generating kernel | Found kernel |
|-------------------|--------------|
| LIN+SE            | SE+LIN+SE×LIN|
| LIN+LIN×SE       | SE+SE        |
| LIN+LIN×SE+PER   | SE+LIN+SE+LIN+SE×LIN+SE×PER |

Table 1. Comparing the true generated kernel with the found structure

Although the generated data seems similar in terms of global shapes, the data has clear differences in terms of local structures. The first column of Figure 1 show the three generated time series. It is easy to notice that all time series share trending shapes with smoothing caused by LIN and SE kernels. However, the first time series has more LIN trending component than the second. In addition, PER kernel is one of the components playing the role of creating the periodicity in the third time series.

The decomposition from this synthetic data can be visually seen in Figure 1, and discovered kernels are in Table 1. Although the synthetic data shared the LIN structure, it looks like having smoothness more than having a clear LIN trending. The discovered kernel from the first time series contains the LIN structure but is dominant by the smoothness of global structures among three times series. One of the interesting observations is that the IBP-GP can distinguish the PER kernel of the third time series in form of the product with SE.

5.2. Real world data

We tested our algorithm on three different data sets: stock prices, housing markets and currency exchanges. These data sets are well-described in (Hwang et al., 2016) and publicly accessible.

The evaluation criterion is the predictive performance in terms of root mean squared error (RMSE). It is hard to make comparison IBP-GP with ABCD and R-ABCD based on other criteria like negative log likelihood (NLL) and Bayesian Information criteria (BIC). In lieu of these criteria, model complexity is controlled by the IBP concentration parameter $\alpha$ and the precision of decomposition $\sigma_X^2$. This can be considered a drawback of IBP-GP because the kernel selection relies on learning $Z$ which does not have any theoretical supports for how to choose the best-fitted selection.

When conducting experiments, we keep $\alpha$ small and decrease it when the number of searched components increases, specifically $\alpha$ was configured in the range from 0.05 to 0.3. Moreover, the bigger value the variance of Gaussian noise $\sigma_X^2$ is picked, the fewer number of learned kernel components we obtain. Thus, it should be chosen properly according to the scales of time series. For example, in stock data sets, the fifth stock has relatively high
value, then we set higher $\sigma_X^2$ in order to get fewer kernel components described this time series in the sense of preferring more simple models than complex ones.

In Table 2, the predictive performance of IBP-GP model with two settings of kernel component is competitive. In both settings, the IBP-GP models produce better results on three data sets when comparing to ABCD. Since ABCD is known as one of the state-of-the-art regression methods, then we can verify the competitiveness of predictive performance from the IBP-GP. However, IBP-GP does not outperform R-ABCD always. As an example, R-ABCD performs better in the housing markets data set where values change smoothly.

In the view of discovering qualitative kernel components, the IBP-GP model demonstrates promising results and captures global structures among multiple time series. In the appendix, we provide the kernel components which are decomposed from the three stocks, containing an important change point (COMP 7) describing the 911 event as discussed in (Hwang et al., 2016). By looking up to the decomposition of the currency exchange data sets, we can find out the similar observation but the meaningful change point kernels are scattered across their components (e.g. 2-nd component - COMP 2, 6-th component - COMP 6), interpreting the main changes in financial markets in 2015.

5.3. Scalability

The aspect of scalability of our model is considered by carrying experiments by changing the numbers of the input time series. We took respectively 10, 20 and 40 time series from S&P500 index (QuantQuote). Each time series has 60 data points from May 16th, 2013 to August 9th, 2013.

To make a fair comparison, we run methods on a single machine and record the run time. Here, we investigate to the workload of IBP-GP. In fact, a considerable amount of time IBP-GP spends is to perform gradient ascents for equation (12) but the number of equations is usually constant. While ABCD proposes candidate kernels whose number tends to grow quickly, then it greatly affects the total learning time. Figure 3 illustrates run time per time series for ABCD and our IBP-GP. It is clear that IBP-GP process more time series with the same duration compared to the ABCD.

6. Discussion

Applicability. There are some potential applications which can be pointed out as followings:

Comparison report: By taking the advantage of the learned latent matrix $Z$ and the descriptive properties of found GP covariance structures, we can generate a readable report containing the comparison among time series. For example, the generated text can have formats like

```
"[T] all share [description]"
"[T] has [description] while [T] does not"
```

where the replacement of [T] is time series, [description] is generated by ABCD based on the found GP structure.

Query time series system: A similar system is mentioned in (Schaechtle et al., 2015) but the system should train and store encoded GPs. By using this method, there is requirements of preprocessed data. If we limit a reasonable range of queries (number of components) and the algorithm is implemented in distributed manner, we can obtain results within an acceptable response time.

7. Conclusion

We presented the combination of IBP and GP where IBP help finding kernel structures for GP. This is the first approach for kernel selection problem in the ABCD framework. We also contribute the derivation of variational inference for the proposed model. We show that our model provides a flexible framework by designing the set of kernel components according to various purposes e.g. accuracy, interpretability, and switching dynamic problem.
References

Bonilla, Edwin V., Chai, Kian Ming Adam, and Williams, Christopher K. I. Multi-task gaussian process prediction. In Advances in Neural Information Processing Systems, pp. 153–160, 2007.

Doshi, Finale, Miller, Kurt, Gail, Jurgen Van, and Teh, Yee Whye. Variational inference for the indian buffet process. In International Conference on Artificial Intelligence and Statistics, pp. 137–144, 2009.

Doshi-Velez, Finale and Ghahramani, Zoubin. Accelerated sampling for the indian buffet process. In International Conference on Machine Learning, pp. 273–280, 2009.

Duvenaud, David, Lloyd, James Robert, Grosse, Roger, Tenenbaum, Joshua B., and Ghahramani, Zoubin. Structure discovery in nonparametric regression through compositional kernel search. 2013.

Fox, E.B., Hughes, M.C., Sudderth, E.B., and Jordan, M.I. Joint modeling of multiple related time series via the beta process with application to motion capture segmentation. Annals of Applied Statistics, 8(3):1281–1313, 2014.

Griffiths, Thomas L. and Ghahramani, Zoubin. Infinite latent feature models and the indian buffet process. In Advances in Neural Information Processing Systems, pp. 475–482, 2005.

Griffiths, Thomas L. and Ghahramani, Zoubin. The indian buffet process: An introduction and review. Journal of Machine Learning Research, 12:1185–1224, 2011.

Grosse, R.B., Salakhutdinov, R., Freeman, W.T., and Tenenbaum, J.B. Exploiting compositionality to explore a large space of model structures. In Uncertainty in Artificial Intelligence, 2012.

Hwang, Yunseong, Tong, Anh, and Choi, Jaesik. Automatic construction of nonparametric relational regression models for multiple time series. In International Conference on Machine Learning, pp. 3030–3039, 2016.

Lloyd, James Robert, Duvenaud, David, Grosse, Roger, Tenenbaum, Joshua B., and Ghahramani, Zoubin. Automatic construction and Natural-Language description of nonparametric regression models. In Association for the Advancement of Artificial Intelligence, 2014.

Minka, T. P. and Picard, R. W. Learning how to learn with point sets. http://research.microsoft.com/minka/papers/point-sets.html.

QuantQuote. QuantQuote Free Historical Stock Data. http://quantquote.com/files/quantquote_daily_sp500_83986.zip.

Rasmussen, Carl Edward and Williams, Christopher K. I. Gaussian Processes for Machine Learning (Adaptive Computation and Machine Learning). The MIT Press, 2005. ISBN 026218253X.

Reed, Colorado and Ghahramani, Zoubin. Scaling the indian buffet process via submodular maximization. In International Conference on Machine Learning, pp. 1013–1021, 2013.

Schaechtele, Ulrich, Zinberg, Ben, Radul, Alexey, Stathis, Kostas, and Mansinghka, Vikash K. Probabilistic programming with gaussian process memoization. CoRR, abs/1512.05665, 2015. URL http://arxiv.org/abs/1512.05665.

Shah, Amar, Knowles, David A., and Ghahramani, Zoubin. An empirical study of stochastic variational inference algorithms for the beta bernoulli process. In International Conference on Machine Learning, pp. 1594–1603, 2015.

Wainwright, Martin J. and Jordan, Michael I. Graphical models, exponential families, and variational inference. Foundations and Trends in Machine Learning, 1(1-2):1–305, 2008.

Wilson, Andrew Gordon and Adams, Ryan Prescott. Gaussian process kernels for pattern discovery and extrapolation. In International Conference on Machine Learning, volume 28, pp. 1067–1075, 2013.
Appendix

Base kernels

| Base Kernels                  | Encoding Function               | $k(x, x')$ |
|-------------------------------|--------------------------------|------------|
| White Noise (WN)              | Uncorrelated noise              | $\sigma^2 \delta(x, x')$ |
| Constant (C)                  | Constant functions              | $\sigma^2$  |
| Linear (LIN)                  | Linear functions                | $\sigma^2(x - l)(x - l')$ |
| Squared Exponential (SE)      | Smooth functions                | $\sigma \exp\left(\frac{(x-x')^2}{2\sigma^2}\right)$ |
| Periodic (PER)                | Periodic functions              | $\sigma^2 \exp\left(\frac{2\pi(x-x')}{l}\right) - I_0\left(\frac{1}{l}\right)$ |

Search grammar in the ABCD framework

\[
S \rightarrow S + B \\
S \rightarrow CP(S, S) \\
S \rightarrow C \\
S \rightarrow B \\
S \rightarrow S \times B \\
S \rightarrow CW(S, S) \\
\]

where $S$ represents any kernel subexpression, $B$ and $B'$ are base kernels.

Details derivation of $p(X|Z)$

\[
p(X|Z, \sigma_X, c_1, ..., c_K) = \int p(X|Z, F, \sigma_X) P(F|c_1, ..., c_k) dF
\]

\[
= \frac{1}{(2\pi)^{(N^D+K^D)/2} \sigma_X^{-N^D} \prod_{k=1}^{K} |c_k|} \exp\left\{ - \frac{1}{2\sigma_X^2} \text{tr}(X^T X) + \frac{1}{2\sigma_X^2} \vartheta(X^T Z) G \vartheta(X^T Z) \right\}
\]

\[
= \int \exp\left\{ - \frac{1}{2} (G \vartheta(X^T Z) - \vartheta(F^T)) (\sigma_X^2 G)^{-1} (G \vartheta(X^T Z) - \vartheta(F^T)) \right\} dF
\]

\[
= \frac{\sqrt{2\pi^{K^D} |\sigma_X^2 G|}}{(2\pi)^{N^D/2} \sigma_X^{-N^D} \prod_{k=1}^{K} |c_k|} \exp\left\{ - \frac{1}{2\sigma_X^2} \text{tr}(X^T X) + \frac{1}{2\sigma_X^2} \vartheta(X^T Z) G \vartheta(X^T Z) \right\}
\]

\[
= \frac{1}{(2\pi)^{N^D/2} \sigma_X^{-N^D - K^D} \sqrt{(|I_D \otimes Z^T Z + \sigma_X^2 \sum_{k=1}^{K} c_k^{-1} \otimes \Delta_k| \prod_{k=1}^{K} |c_k|} \exp\left\{ - \frac{1}{2\sigma_X^2} \text{tr}(X^T X) + \frac{1}{2\sigma_X^2} \vartheta(X^T Z) G \vartheta(X^T Z) \right\}
\]

Time series decompositions

We provide the illustrative figures about how times series are decomposed in several real world problem. Those figures are viewed in vertical way. The the first column is the original data. The remain ones are decomposed components. The gray in the background indicates the value of $z_{nk}$ equals to 1.
Figure 1. The decomposition of three stocks. There are 21 found components in the data
Figure 2. The decomposition of six stocks
Figure 3. The decomposition of nine stocks
Figure 4. The decomposition of 4 currency exchanges