Journal of Machine Learning Research 17 (2016) 1-36 Submitted 10/14; Revised 3/16; Published 8/16

Variational Dependent Multi-output Gaussian Process Dynamical Systems

Jing Zhao * jzhao2011@gmail.com

Shiliang Sun * shiliangsun@gmail.com

Department of Computer Science and Technology
East China Normal University
500 Dongchuan Road, Shanghai 200241, P. R. China

Editor: Neil Lawrence

Abstract

This paper presents a dependent multi-output Gaussian process (GP) for modeling complex dynamical systems. The outputs are dependent in this model, which is largely different from previous GP dynamical systems. We adopt convolved multi-output GPs to model the outputs, which are provided with a flexible multi-output covariance function. We adapt the variational inference method with inducing points for learning the model. Conjugate gradient based optimization is used to solve parameters involved by maximizing the variational lower bound of the marginal likelihood. The proposed model has superiority on modeling dynamical systems under the more reasonable assumption and the fully Bayesian learning framework. Further, it can be flexibly extended to handle regression problems. We evaluate the model on both synthetic and real-world data including motion capture data, traffic flow data and robot inverse dynamics data. Various evaluation methods are taken on the experiments to demonstrate the effectiveness of our model, and encouraging results are observed.

Keywords: Gaussian process, variational inference, dynamical system, multi-output modeling

1. Introduction

Dynamical systems are widespread in the research area of machine learning. Multi-output time series such as motion capture data, traffic flow data and video sequences are typical examples generated from these systems. Data generated from these dynamical systems usually have the following characteristics. 1) Implicit dynamics exist in the data, and the relationship between the observations and the time indices is nonlinear. For example, the transformation of the frames of a video over time is complex. 2) Possible dependency exists among multiple outputs. For example, for motion capture data, the position of the hand is often closely related to the position of the arm. A simple and straightforward method to model this kind of dynamical systems is to use Gaussian processes (GPs), since GPs provide an elegant method for modeling nonlinear mappings in the Bayesian nonparametric learning framework (Rasmussen and Williams, 2006). Some extensions of GPs have been developed in recent years to better model the dynamical systems. The dynamical systems

*. The authors contributed equally to this work.

modeled by GPs are called the Gaussian process dynamical systems (GPDSs). However, the existing GPDSs have a limitation of ignoring the dependency among the multiple outputs, that is, they may not make full use of the characteristics of data. Our work aims to model the complex dynamical systems more reasonably and flexibly.

Gaussian process dynamical models (GPDMs) as extensions of the GP latent variable model (GP-LVM) (Lawrence, 2004, 2005) were proposed to model human motion (Wang et al., 2006, 2008). The GP-LVM is a nonlinear extension of the probabilistic principal component analysis (Tipping and Bishop, 1999) and is a probabilistic model where the outputs are observed while the inputs are hidden. It introduces latent variables and performs a nonlinear mapping from the latent space to the observation space. The GP-LVM provides an unsupervised non-linear dimensionality reduction method by optimizing the latent variables with the maximum a posteriori (MAP) solution. The GPDM allows to model nonlinear dynamical systems by adding a Markov dynamical prior on the latent space in the GP-LVM. It captures the variability of outputs by constructing the variance of outputs with different parameters. Some research of adapting GPDMs to specific applications was developed, such as object tracking (Urtasun et al., 2006), activity recognition (Gamage et al., 2011) and synthesis, and computer animation (Henter et al., 2012).

Similarly, Damianou et al. (2011, 2014) extended the GP-LVM by imposing a dynamical prior on the latent space to the variational GP dynamical system (VGPDS). The nonlinear mapping from the latent space to the observation space in the VGPDS allows the model to capture the structures and characteristics of data in a relatively low dimensional space. Instead of seeking a MAP solution for the latent variables as in GPDMs, VGPDSs used a variational method for model training. This follows the variational Bayesian method for training the GP-LVM (Titsias and Lawrence, 2010), in which a lower bound of the logarithmic marginal likelihood is computed by variationally integrating out the latent variables that appear nonlinearly in the inverse kernel matrix of the model. The variational Bayesian method was built on the method of variational inference with inducing points (Titsias, 2009). The VGPDS approximately marginalizes out the latent variables and leads to a rigorous lower bound on the logarithmic marginal likelihood. The model and variational parameters of the VGPDS can be learned through maximizing the variational lower bound. This variational method with inducing points was also employed to integrate out the warping functions in the warped GP (Snelson et al., 2003; Lázaro-gredilla, 2012). Park et al. (2012) developed an almost direct application of VGPDSs to phoneme classification, in which a variance constraint in the VGPDS was introduced to eliminate the sparse approximation error in the kernel matrix. Besides variational approaches, expectation propagation based methods (Deisenroth and Mohamed, 2012) are also capable of conducting approximate inference in GPDSs.

However, all the models mentioned above for GPDSs ignore the dependency among multiple outputs, which usually assume that the outputs are conditionally independent. Actually, modeling the dependency among outputs is necessary in many applications such as sensor networks, geostatistics and time-series forecasting, which helps to make better predictions (Boyle, 2007). Indeed, there are some recent works that explicitly considered the dependency of multiple outputs in GPs (Álvarez et al., 2009; Álvarez and Lawrence, 2011; Wilson et al., 2012). Latent force models (LFMs) (Álvarez et al., 2009) are a recent state-of-the-art modeling framework, which can model multi-output dependencies. Later, a series of extensions of LFMs were presented such as linear, nonlinear, cascaded and switching dynamical LFMs (Álvarez et al., 2011, 2013). In addition, sequential inference methods for LFMs have also been developed (Hartikainen and Särkkä, 2012). Álvarez and Lawrence (2011) employed convolution processes to account for the correlations among outputs to construct a convolved multiple outputs GP (CMOGP) which can be regarded as a specific case of LFMs. To overcome the difficulty of time and storage complexities for large data sets, some efficient approximations for the CMOGP were constructed through the convolution formalism (Álvarez et al., 2010; Álvarez and Lawrence, 2011). This leads to a form of covariance similar in spirit to the so called deterministic training conditional (DTC) approximation (Csató and Opper, 2001), fully independent training conditional (FITC) approximation (Quiñonero-Candela and Rasmussen, 2005; Snelson and Ghahramani, 2006) and partially independent training conditional (PITC) approximation (Quiñonero-Candela and Rasmussen, 2005) for a single output (Lawrence, 2007). The CMOGP is then enhanced and extended to the collaborative multi-output Gaussian process (COGP) for handling large scale cases (Nguyen and Bonilla, 2014). Besides CMOGPs, Wilson et al. (2012) combined neural networks with GPs to construct a GP regression network (GPRN). Outputs in the GPRN are linear combinations of the shared adaptive latent basis functions with input dependent weights. However, these two models are neither introduced nor directly suitable for complex dynamical system modeling. When a dynamical prior is imposed, marginalizing over the latent variables is needed, which can be very challenging.

In this paper, we propose a variational dependent multi-output GP dynamical system (VDM-GPDS). It is a hierarchical Gaussian process model in which the dependency among all the observations is well captured. Specifically, the convolved process covariance function (Álvarez and Lawrence, 2011) is employed to capture the dependency among all the data points across all the outputs. To learn the VDM-GPDS, we first approximate the latent functions in the convolution processes, and then variationally marginalize out the latent variables in the model. This leads to a convenient lower bound of the logarithmic marginal likelihood, which is then maximized by the scaled conjugate gradient method to find out the optimal parameters.

The highlights of this paper are summarized as follows. 1) We explicitly take the dependency among multiple outputs into consideration while other methods (Damianou et al., 2011; Park et al., 2012) for GPDS modeling assume that the outputs are conditionally independent. In particular, the convolved process covariance functions are used to construct the covariance matrix of the outputs. 2) We use the variational method to compute a lower bound of the logarithmic marginal likelihood of the GPDS model. Compared to Damianou et al. (2011), our model is more reasonable in specific practical settings, and more challenging as a result of involving complex formulations and computations. 3) Our model can been seen as a multi-layer regression model which regards time indices as inputs and observations as outputs. It can be flexibly extended to handle regression problems. Compared with other dependent multi-output models such as the CMOGP, the VDM-GPDS can achieve much better performance attributed to its latent layers. 4) Our model is applicable to general dependent multi-output dynamical systems and multi-output regression tasks, rather than being specially tailored to a particular application. In this paper, we adapt the model to different applications and obtain promising results.

An earlier short version of this work appeared in Zhao and Sun (2014). In this paper, we add detailed derivations of the variational inference and provide the gradients of the objective function with respect to the parameters. Moreover, we analyze the performance and efficiency of the proposed model. In addition, we supplement experiments on real-world data and all the experimental results are measured under various evaluation criteria.

The rest of the paper is organized as follows. First, we give the model for nonlinear dynamical systems in Section 2, where we use convolution process covariance functions to construct the covariance matrix of the dependent multi-output latent variables. Section 3 gives the derivation of the variational lower bound of the marginal likelihood function and optimization methods. Prediction formulations are introduced in Section 4. Related work is analyzed and compared in Section 5. Experimental results are reported in Section 6 and finally conclusions are presented in Section 7.

2. The Proposed Model

Suppose we have multi-output time series data $\{\mathbf{y}_n, t_n\}_{n=1}^N$, where $\mathbf{y}_n \in \mathbb{R}^D$ is an observation at time $t_n \in \mathbb{R}^+$. We assume that there are low dimensional latent variables that govern the generation of the observations and a GP prior for the latent variables conditional on time captures the dynamical driving force of the observations, as in Damianou et al. (2011). However, a large difference compared with their work is that we explicitly model the dependency among the outputs through convolution processes (Álvarez and Lawrence, 2011).

Our model is a four-layer GP dynamical system. Here $\mathbf{t} \in \mathbb{R}^N$ represents the input variables in the first layer. The matrix $X \in \mathbb{R}^{N\times Q}$ represents the low dimensional latent variables in the second layer with element $x_{nq} = x_q(t_n)$. Similarly, the matrix $F \in \mathbb{R}^{N\times D}$ denotes the latent variables in the third layer, with element $f_{nd} = f_d(\mathbf{x}_n)$ and the matrix $Y \in \mathbb{R}^{N\times D}$ denotes the observations in the fourth layer whose nth row corresponds to $\mathbf{y}_n$. The model is composed of an independent multi-output GP mapping from $\mathbf{t}$ to X, a dependent multi-output GP mapping from X to F, and a linear mapping from F to Y.

Specifically, for the first mapping, $\mathbf{x}$ is assumed to be a multi-output GP indexed by time t similarly to Damianou et al. (2011), that is

$$x_q(t) \quad \sim \quad \mathcal{GP}(0, \kappa_x(t,t')), \quad q = 1, ..., Q, \tag{1}$$

where individual components of the latent function $\mathbf{x}(t)$ are independent sample paths drawn from a GP with a certain covariance function $\kappa_x(t,t')$ parameterized by $\boldsymbol{\theta}_x$. There are several commonly used covariance functions such as the squared exponential covariance function (RBF), the Matérn 3/2 function and the periodic covariance function (RBFperiodic), which can be adopted to model the time evolution of sequences. For example, an RBF or a Matérn 3/2 function is usually appropriate for a long time dependent sequence, which will lead to a full covariance matrix. For modeling the evolution of multiple independent sequences, a block-diagonal covariance matrix should be chosen, where each block can be constructed by an RBF or a Matérn 3/2 function. RBFperiodic is useful to capture the periodicity of the sequences, and multiple kernels can be used to model different time cycles. These kernel functions take the following forms.

$$\begin{aligned}
&\kappa_{x(RBF)}(t_i,t_j)=\sigma^2_{rbf}e^{\frac{(t_i-t_j)^2}{2\ell^2}},\\
&\kappa_{x(Mat\acute{e}rn3/2)}(t_i,t_j)=\sigma^2_{mat}(1+\frac{\sqrt{3}|t_i-t_j|}{\ell})e^{\frac{-\sqrt{3}|t_i-t_j|}{\ell}},\\
&\kappa_{x(RBFperiodic)}(t_i,t_j)=\sigma^2_{per}e^{-\frac{1}{2}\frac{sin^2(\frac{2\pi}{T}(t_i-t_j))}{\ell}}.
\end{aligned} \tag{2}$$

According to the conditional independency assumption among the latent variables $\{\mathbf{x}_q\}_{q=1}^{Q}$, we have

$$p(X|\mathbf{t})=\prod_{q=1}^{Q}p(\mathbf{x}_q|\mathbf{t})=\prod_{q=1}^{Q}\mathcal{N}(\mathbf{x}_q|\mathbf{0},\mathbf{K}_{\mathbf{t},\mathbf{t}}), \tag{3}$$

where $\mathbf{K}_{\mathbf{t},\mathbf{t}}$ is the covariance matrix constructed by $\kappa_x(t,t')$.

For the second mapping, we assume that $\mathbf{f}$ is another multi-output GP indexed by $\mathbf{x}$, whose outputs are dependent, that is

$$f_d(\mathbf{x})\sim\mathcal{GP}(0,\kappa_{f_d,f_{d'}}(\mathbf{x},\mathbf{x}')),\quad d,d'=1,...,D, \tag{4}$$

where $\kappa_{f_d,f_{d'}}(\mathbf{x},\mathbf{x}')$ is a convolved process covariance function. The convolved process covariance function captures the dependency among all the data points across all the outputs with parameters $\boldsymbol{\theta}_f=\{\{\Lambda_k\},\{P_d\},\{S_{d,k}\}\}$. The detailed formulation of this covariance function will be given in Section 2.1. From the conditional dependency among the latent variables $\{f_{nd}\}_{n=1,d=1}^{N,D}$, we have

$$p(F|X)=p(\mathbf{f}|X)=\mathcal{N}(\mathbf{f}|\mathbf{0},\mathbf{K}_{\mathbf{f},\mathbf{f}}), \tag{5}$$

where $\mathbf{f}$ is a shorthand for $[\mathbf{f}_1^\top,...,\mathbf{f}_D^\top]^\top$ and $\mathbf{K}_{\mathbf{f},\mathbf{f}}$ sized $ND\times ND$ is the covariance matrix in which the elements are calculated by the covariance function $\kappa_{f_d,f_{d'}}(\mathbf{x},\mathbf{x}')$.

The third mapping, which is from the latent variable f_{nd} to the observation y_{nd}, can be written as

$$y_{nd}=f_{nd}+\epsilon_{nd},\quad \epsilon_{nd}\sim\mathcal{N}(0,\beta^{-1}). \tag{6}$$

Since the observations $\{y_{nd}\}_{n=1,d=1}^{N,D}$ are conditionally independent on F, we get

$$p(Y|F)=\prod_{d=1}^{D}\prod_{n=1}^{N}\mathcal{N}(y_{nd}|f_{nd},\beta^{-1}). \tag{7}$$

Given the above setting, the graphical model for the proposed VDM-GPDS on the training data $\{\mathbf{y}_n,t_n\}_{n=1}^{N}$ can be depicted as Figure 1. From (3), (5) and (7), the joint probability distribution for the VDM-GPDS model is given by

$$p(Y,F,X|\mathbf{t})=p(Y|F)p(F|X)p(X|\mathbf{t})=p(\mathbf{f}|X)\prod_{d=1}^{D}\prod_{n=1}^{N}p(y_{nd}|f_{nd})\prod_{q=1}^{Q}p(\mathbf{x}_q|\mathbf{t}). \tag{8}$$

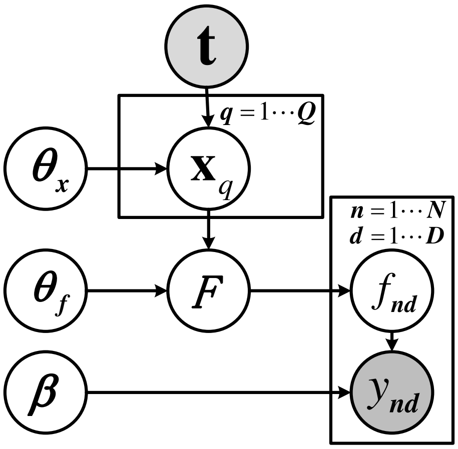


Figure 1: The graphical model for VDM-GPDS.

2.1 Convolved Process Covariance Function

Since the outputs in our model are dependent, we need to capture the correlations among all the data points across all the outputs. Bonilla et al. (2007) and Luttinen and Ilin (2012) used a Kronecker product covariance matrix with the form of $K_{FF} = K_{DD} \otimes K_{NN}$, where K_{DD} is the covariance matrix among the output dimensions, and K_{NN} is the covariance matrix calculated solely from the data inputs. This Kronecker form kernel is constructed from the processes which involve some form of instantaneous mixing of a series of independent processes. This is very limited and actually a special case of some general covariances when covariances calculated from outputs and inputs are independent (Álvarez et al., 2012). For example, if we want to model two output processes in such a way that one process was a blurred version of the other, we cannot achieve this through the instantaneous mixing (Álvarez and Lawrence, 2011). In this paper, we use a more general and flexible kernel in which K_{DD} and K_{NN} are not separated. In particular, the convolution processes (Álvarez and Lawrence, 2011) are employed to model the latent function $F(X)$.

Now we introduce how to construct the convolved process covariance functions. By using independent latent functions $\{u_k(\mathbf{x})\}_{k=1}^{K}$ and smoothing kernels $\{G_{d,k}(\mathbf{x})\}_{d=1,k=1}^{D,K}$ in the convolution processes, each latent function in (4) in the VDM-GPDS is expressed through a convolution integral,

$$f_d(\mathbf{x}) = \sum_{k=1}^{K} \int_X G_{d,k}(\mathbf{x} - \tilde{\mathbf{x}}) u_k(\tilde{\mathbf{x}}) d\tilde{\mathbf{x}}. \tag{9}$$

The most common construction is to use Gaussian forms for $\{u_k(\mathbf{x})\}_{k=1}^{K}$ and $\{G_{d,k}(\mathbf{x})\}_{d=1,k=1}^{D,K}$. So the smoothing kernel is assumed to be

$$G_{d,k}(\mathbf{x}) = S_{d,k} \mathcal{N}\left(\mathbf{x}|\mathbf{0}, P_d\right), \tag{10}$$

where $S_{d,k}$ is a scalar value that depends on the output index d and the latent function index k, and P_d is assumed to be diagonal. The latent process $u_k(\mathbf{x})$ is assumed to be

Gaussian with covariance function

$$\kappa_k\left(\mathbf{x},\mathbf{x}'\right)=\mathcal{N}\left(\mathbf{x}-\mathbf{x}'|\mathbf{0},\Lambda_k\right). \tag{11}$$

Thus, the covariance between $f_d(\mathbf{x})$ and $f_{d'}(\mathbf{x}')$ is

$$\kappa_{f_d,f_{d'}}(\mathbf{x},\mathbf{x}')=\sum_{k=1}^{K}S_{d,k}S_{d',k}\mathcal{N}(\mathbf{x}|\mathbf{x}',P_d+P_{d'}+\Lambda_k). \tag{12}$$

The covariance between $f_d(\mathbf{x})$ and $u_k(\mathbf{x}')$ is

$$\kappa_{f_d,u_k}(\mathbf{x},\mathbf{x}')=S_{d,k}\mathcal{N}\left(\mathbf{x}-\mathbf{x}'|\mathbf{0},P_d+\Lambda_k\right). \tag{13}$$

These covariance functions (11), (12) and (13) will later be used for approximate inference in Section 3. Compared with Kronecker form kernels, our used convolved kernels have the following advantages. From the perspective of constructing the process f_d, convolved kernels are constructed using the convolution process f_d in which the smoothing kernels $G_{d,k}(\mathbf{x})$ related to $\mathbf{x}$ are employed while Kronecker form kernels are constructed using $f_d(x)=\mathbf{a}_d\mathbf{u}_k(\mathbf{x})$ in which $\mathbf{a}_d$ has no relation to $\mathbf{x}$ (Álvarez and Lawrence, 2011). From the perspective of kernels, for different dimensions d and d', convolved kernels allow that the covariances $\{\kappa_{f_d,f_{d'}}(\mathbf{x},\mathbf{x}')\}$ are related to different terms $\mathcal{N}(\mathbf{x}|\mathbf{x}',P_d+P_{d'}+\Lambda_k)$ while Kronecker form kernels indicate that different covariances $\{\kappa_{f_d,f_{d'}}(\mathbf{x},\mathbf{x}')\}$ share the same term $\kappa_q(\mathbf{x},\mathbf{x}')$. Thus, our used convolved kernels are more general.

3. Inference and Optimization

As described above, the proposed VDM-GPDS explicitly models the dependency among multiple outputs, which makes it largely distinct to the previous VGPDS and other GP dynamical systems. In order to make it easy to implement by extending the existing framework of the VGPDS, in the current and the following sections, we will deduce the variational lower bound for the logarithmic marginal likelihood and the posterior distribution for prediction in a formulation similar to the VGPDS. However, many details as described in the paper are specific to our model, and some calculations are more involved.

The fully Bayesian learning for our model requires maximizing the logarithm of the marginal likelihood (Bishop, 2006)

$$p(Y|\mathbf{t})=\int p(Y|F)p(F|X)p(X|\mathbf{t})dXdF. \tag{14}$$

Note that the integration w.r.t X is intractable, because X appears nonlinearly in the inverse of the matrix $\mathbf{K}_{\mathbf{f},\mathbf{f}}$. We attempt to make some approximations for (14).

To begin with, we approximate $p(F|X)$ which is constructed by convolution process $f_d(\mathbf{x})$ in (9). Similarly to Álvarez and Lawrence (2011), a generative approach is used to approximate $f_d(\mathbf{x})$ as follows. We first draw a sample, $\mathbf{u}_k(Z)=[u_k(\mathbf{z}_1),...,u_k(\mathbf{z}_M)]^\top$, where $Z=\{\mathbf{z}_m\}_{m=1}^M$ are introduced as a set of input vectors for $u_k(\tilde{\mathbf{x}})$ and will be learned as parameters. We next sample $u_k(\tilde{\mathbf{x}})$ from the conditional prior $p(u_k(\tilde{\mathbf{x}})|\mathbf{u}_k)$. According to the above generating process, $u_k(\tilde{\mathbf{x}})$ in (9) can be approximated by the expectation

$\mathcal{E}(u_k(\tilde{\mathbf{x}})|\mathbf{u}_k)$. Let $U = \{\mathbf{u}_k\}_{k=1}^K$ and $\mathbf{u} = [\mathbf{u}_1^\top, ..., \mathbf{u}_K^\top]^\top$. The probability distribution of $\mathbf{u}$ can be expressed as

$$p(\mathbf{u}|Z) = \mathcal{N}(\mathbf{0}, \mathbf{K}_{\mathbf{u},\mathbf{u}}), \tag{15}$$

where $\mathbf{K}_{\mathbf{u},\mathbf{u}}$ is constructed by $\kappa_k(\mathbf{x}, \mathbf{x}')$ in (11). Combining (9) and (15), we get the probability distribution of $\mathbf{f}$ conditional on $\mathbf{u}, X, Z$ as

$$p(\mathbf{f}|\mathbf{u}, X, Z) = \mathcal{N}(\mathbf{f}|\mathbf{K}_{\mathbf{f},\mathbf{u}}\mathbf{K}_{\mathbf{u},\mathbf{u}}^{-1}\mathbf{u}, \mathbf{K}_{\mathbf{f},\mathbf{f}} - \mathbf{K}_{\mathbf{f},\mathbf{u}}\mathbf{K}_{\mathbf{u},\mathbf{u}}^{-1}\mathbf{K}_{\mathbf{u},\mathbf{f}}), \tag{16}$$

where $\mathbf{K}_{\mathbf{f},\mathbf{u}}$ is the cross-covariance matrix between $f_d(\mathbf{x})$ and $u_k(\mathbf{z})$ with element $\kappa_{f_d,u_k}(\mathbf{x}, \mathbf{x}')$ in (13), the block-diagonal matrix $\mathbf{K}_{\mathbf{u},\mathbf{u}}$ is the covariance matrix between $u_k(\mathbf{z})$ and $u_k(\mathbf{z}')$ with element $\kappa_k(\mathbf{x}, \mathbf{x}')$ in (11), and $\mathbf{K}_{\mathbf{f},\mathbf{f}}$ is the covariance matrix between $f_d(\mathbf{x})$ and $f_{d'}(\mathbf{x}')$ with element $\kappa_{f_d,f_{d'}}(\mathbf{x}, \mathbf{x}')$ in (12). Therefore, $p(F|X)$ is approximated by

$$p(F|X) \approx p(\mathbf{f}|X, Z) = \int p(\mathbf{f}|\mathbf{u}, X, Z)p(\mathbf{u}|Z)d\mathbf{u}, \tag{17}$$

and $p(Y|\mathbf{t})$ is converted to

$$p(Y|\mathbf{t}) \approx p(Y|\mathbf{t}, Z) = \int p(\mathbf{y}|\mathbf{f})p(\mathbf{f}|\mathbf{u}, X, Z)p(\mathbf{u}|Z)p(X|\mathbf{t})dFdUdX, \tag{18}$$

where $p(\mathbf{u}|Z) = \mathcal{N}(\mathbf{0}, \mathbf{K}_{\mathbf{u},\mathbf{u}})$ and $\mathbf{y} = [\mathbf{y}_1^\top, ..., \mathbf{y}_D^\top]^\top$. It is worth noting that the marginal likelihood in (18) is still intractable as the integration with respect to X remains difficult.

Then, we introduce a lower bound of the logarithmic marginal likelihood $\log p(Y|\mathbf{t})$ using variational methods. We construct a variational distribution $q(F, U, X|Z)$ to approximate the posterior distribution $p(F, U, X|Y, \mathbf{t})$ and compute the Jensen's lower bound on $\log p(Y|\mathbf{t})$ as

$$\mathcal{L} = \int q(F, U, X|Z) \log \frac{p(Y, F, U, X|\mathbf{t}, Z)}{q(F, U, X|Z)} dXdUdF. \tag{19}$$

The variational distribution is assumed to be factorized as

$$q(F, U, X|Z) = p(\mathbf{f}|\mathbf{u}, X, Z)q(\mathbf{u})q(X). \tag{20}$$

The distribution $p(\mathbf{f}|\mathbf{u}, X, Z)$ in (20) is the same as the second term in (18), which will be eliminated in the term $\log \frac{p(Y,F,U,X|\mathbf{t},Z)}{q(F,U,X|Z)}$ in (19). The distribution $q(\mathbf{u})$ is an approximation to the posterior distribution $p(\mathbf{u}|X, Y)$, which is arguably Gaussian by maximizing the variational lower bound (Titsias and Lawrence, 2010; Damianou et al., 2011). The distribution $q(X)$ is an approximation to the posterior distribution $p(X|Y)$, which is assumed to be a product of independent Gaussian distributions $q(X) = \prod_{q=1}^Q \mathcal{N}(\mathbf{x}_q|\boldsymbol{\mu}_q, S_q)$.

After some calculations and simplifications, the lower bound with X, U and F integrated out becomes

$$\begin{aligned}\mathcal{L} = &\log \left[\frac{\beta^{\frac{ND}{2}} |\mathbf{K}_{\mathbf{u},\mathbf{u}}|^{\frac{1}{2}}}{(2\pi)^{\frac{ND}{2}} |\beta\psi_2 + \mathbf{K}_{\mathbf{u},\mathbf{u}}|^{\frac{1}{2}}} \exp\{-\frac{1}{2}\mathbf{y}^\top W \mathbf{y}\} \right] \\ &- \frac{\beta\psi_0}{2} + \frac{\beta}{2}\mathrm{Tr}(\mathbf{K}_{\mathbf{u},\mathbf{u}}^{-1}\psi_2) - \mathbf{KL}[q(X)||p(X|\mathbf{t})],\end{aligned} \tag{21}$$

where $W = \beta I - \beta^2 \psi_1 (\beta \psi_2 + \mathbf{K}_{\mathbf{u},\mathbf{u}})^{-1} \psi_1^\top$, $\psi_0 = \mathrm{Tr}(\langle \mathbf{K}_{\mathbf{f},\mathbf{f}} \rangle_{q(X)})$, $\psi_1 = \langle \mathbf{K}_{\mathbf{f},\mathbf{u}} \rangle_{q(X)}$ and $\psi_2 = \langle \mathbf{K}_{\mathbf{u},\mathbf{f}} \mathbf{K}_{\mathbf{f},\mathbf{u}} \rangle_{q(X)}$. $\mathbf{KL}[q(X)||p(X|\mathbf{t})]$ defined by $\int q(X) \log \frac{q(X)}{p(X|\mathbf{t})} dX$ is expressed as

$$\begin{aligned} \mathbf{KL}[q(X)||p(X|\mathbf{t})] =& \frac{Q}{2} \log |\mathbf{K}_{\mathbf{t},\mathbf{t}}| - \frac{1}{2} \sum_{q=1}^{Q} \log |S_q| \\ &+ \frac{1}{2} \sum_{q=1}^{Q} [\mathrm{Tr}(\mathbf{K}_{\mathbf{t},\mathbf{t}}^{-1} S_q) + \mathrm{Tr}(\mathbf{K}_{\mathbf{t},\mathbf{t}}^{-1} \boldsymbol{\mu}_q \boldsymbol{\mu}_q^\top)] + const. \end{aligned} \tag{22}$$

The detailed derivation of this variational lower bound is described in Appendix A where $\mathcal{L}$ is expressed as $\mathcal{L} = \hat{\mathcal{L}} - \mathbf{KL}[q(X)||p(X|\mathbf{t})]$.

Note that although the lower bound in (21) and the one in VGPDS (Damianou et al., 2011) look similar, they are essentially distinct and have different meanings. In particular, the variables U in this paper are the samples of the latent functions $\{u_k(\mathbf{x})\}_{k=1}^{K}$ in the convolution process while in VGPDS they are samples of the latent variables F. Moreover, the covariance functions of F involved in this paper are multi-output covariance functions while VGPDS adopts single-output covariance functions. As a result, our model is more flexible and challenging. For example, the calculation of statistics of ψ_0, ψ_1 and ψ_2 is more complex, as well as the derivatives of the parameters.

3.1 Computation of ψ_0, ψ_1, ψ_2

Recall that the lower bound in (21) requires computing the statistics $\{\psi_0, \psi_1, \psi_2\}$. We now detail how to calculate them. ψ_0 is a scalar that can be calculated as

$$\begin{aligned} \psi_0 &= \sum_{n=1}^{N} \sum_{d=1}^{D} \int \kappa_{f_d,f_d}(\mathbf{x}_n, \mathbf{x}_n) \mathcal{N}(\mathbf{x}_n | \boldsymbol{\mu}_n, S_n) \, d\mathbf{x}_n \\ &= \sum_{d=1}^{D} \sum_{k=1}^{K} \frac{N S_{d,k} S_{d,k}}{(2\pi)^{\frac{Q}{2}} |2P_d + \Lambda_k|^{\frac{1}{2}}}. \end{aligned} \tag{23}$$

ψ_1 is a $V \times W$ matrix whose elements are calculated as[1]

$$\begin{aligned} (\psi_1)_{v,w} &= \int \kappa_{f_d,u_k}(\mathbf{x}_n, \mathbf{z}_m) \mathcal{N}(\mathbf{x}_n | \boldsymbol{\mu}_n, S_n) d\mathbf{x}_n \\ &= S_{d,k} \mathcal{N}(\mathbf{z}_m | \boldsymbol{\mu}_n, P_d + \Lambda_k + S_n), \end{aligned} \tag{24}$$

where $V = N \times D$, $W = M \times K$, $d = \lfloor \frac{v-1}{N} \rfloor + 1$, $n = v - (d-1)N$, $k = \lfloor \frac{w-1}{M} \rfloor + 1$ and $m = w - (k-1)M$. Here the symbol "$\lfloor \rfloor$" means rounding down. ψ_2 is a $W \times W$ matrix whose elements are calculated as

$$\begin{aligned} (\psi_2)_{w,w'} &= \sum_{d=1}^{D} \sum_{n=1}^{N} \int \kappa_{f_d,u_k}(\mathbf{x}_n, \mathbf{z}_m) \kappa_{f_d,u_{k'}}(\mathbf{x}_n, \mathbf{z}_{m'}) \mathcal{N}(\mathbf{x}_n | \boldsymbol{\mu}_n, S_n) d\mathbf{x}_n \\ &= \sum_{d=1}^{D} \sum_{n=1}^{N} S_{d,k} S_{d,k'} \mathcal{N}(\mathbf{z}_m | \mathbf{z}_{m'}, 2P_d + \Lambda_k + \Lambda_{k'}) \mathcal{N}(\frac{\mathbf{z}_m + \mathbf{z}_{m'}}{2} | \boldsymbol{\mu}_n, \Sigma_{\psi_2}), \end{aligned} \tag{25}$$

[1] We borrow the density formulations to express ψ_1 as well as ψ_2.

where $k = \lfloor \frac{w-1}{M} \rfloor + 1$, $m = w - (k-1)M$, $k' = \lfloor \frac{w'-1}{M} \rfloor + 1$, $m' = w' - (k'-1)M$ and $\Sigma_{\psi_2} = (P_d + \Lambda_k)^\top (2P_d + \Lambda_k + \Lambda_{k'})^{-1}(P_d + \Lambda_{k'}) + S_n$.

3.2 Conjugate Gradient Based Optimization

The parameters involved in (21) include the model parameters $\{\beta, \boldsymbol{\theta}_x, \boldsymbol{\theta}_f\}$ and the variational parameters $\{\{\boldsymbol{\mu}_q, S_q\}_{q=1}^{Q}, Z\}$. In order to reduce the variational parameters to be optimized and speed up convergence, we reparameterize the variational parameters $\boldsymbol{\mu}_q$ and S_q to $\bar{\boldsymbol{\mu}}_q$ and $\boldsymbol{\lambda}_q$ as done in Opper and Archambeau (2009) and Damianou et al. (2011)

$$\begin{aligned} \boldsymbol{\mu}_q &= \mathbf{K}_{\mathbf{t},\mathbf{t}} \bar{\boldsymbol{\mu}}_q, & (26) \\ S_q &= \left(\mathbf{K}_{\mathbf{t},\mathbf{t}}^{-1} + diag(\boldsymbol{\lambda}_q) \right)^{-1}, & (27) \end{aligned}$$

where $diag(\boldsymbol{\lambda}_q) = -2\frac{\partial \hat{\mathcal{L}}}{\partial S_q}$ is an $N \times N$ diagonal, positive matrix whose N-dimensional diagonal is denoted by $\boldsymbol{\lambda}_q$, and $\bar{\boldsymbol{\mu}}_q = \frac{\partial \hat{\mathcal{L}}}{\partial \boldsymbol{\mu}_q}$ is an N-dimensional vector. Now the variational parameters to be optimized are $\{\{\bar{\boldsymbol{\mu}}_q, \boldsymbol{\lambda}_q\}_{q=1}^{Q}, Z\}$. Then the derivatives of the lower bound $\mathcal{L}$ with respect to the variational parameters $\bar{\boldsymbol{\mu}}_q$ and $\boldsymbol{\lambda}_q$ become

$$\begin{aligned} \frac{\partial \mathcal{L}}{\partial \bar{\boldsymbol{\mu}}_q} &= \mathbf{K}_{\mathbf{t},\mathbf{t}} \left(\frac{\partial \hat{\mathcal{L}}}{\partial \boldsymbol{\mu}_q} - \bar{\boldsymbol{\mu}}_q \right), & (28) \\ \frac{\partial \mathcal{L}}{\partial \boldsymbol{\lambda}_q} &= -(S_q \circ S_q) \left(\frac{\partial \hat{\mathcal{L}}}{\partial S_q} + \frac{1}{2}\boldsymbol{\lambda}_q \right). & (29) \end{aligned}$$

All the parameters are jointly optimized by the scaled conjugate gradient method to maximize the lower bound in (21). The detailed gradients with respect to the parameters are given in Appendix B.

4. Prediction

The proposed model can perform prediction for complex dynamical systems in two situations. One is prediction with only time and the other is prediction with time and partial observations. In addition, it can be adapted to regression models.

4.1 Prediction with Only Time

If the model is learned with training data Y, one can predict new outputs with only given time. In the Bayesian framework, we need to compute the posterior distribution of the predicted outputs $Y_* \in \mathbb{R}^{N_* \times D}$ on some given time instants $\mathbf{t}_* \in \mathbb{R}^{N_*}$. The expectation is used as the estimate and the autocovariance is used to show the prediction uncertainty. With the parameters as well as time $\mathbf{t}$ and $\mathbf{t}_*$ omitted, the posterior density is given by

$$p(Y_*|Y) = \int p(Y_*|F_*) p(F_*|X_*, Y) p(X_*|Y) dF_* dX_*, \tag{30}$$

where $F_* \in \mathbb{R}^{N_* \times D}$ denotes the set of latent variables (the noise-free version of Y_*) and $X_* \in \mathbb{R}^{N_* \times Q}$ represents the latent variables in the low dimensional space.

The distribution $p(F_*|X_*,Y)$ is approximated by the variational distribution

$$p\left(F_*|X_*,Y\right) \approx q(\mathbf{f}_*|X_*) = \int p(\mathbf{f}_*|\mathbf{u},X_*)q(\mathbf{u})d\mathbf{u}, \tag{31}$$

where $\mathbf{f}_*^\top = [\mathbf{f}_{*1}^\top,...,\mathbf{f}_{*D}^\top]$, and $p(\mathbf{f}_*|\mathbf{u},X_*)$ is Gaussian. Since the optimal setting for $q(\mathbf{u})$ in our variational framework is also found to be Gaussian, $q(\mathbf{f}_*|X_*)$ is Gaussian that can be computed analytically. The distribution $p\left(X_*|Y\right)$ is approximated by the variational distribution $q\left(X_*\right)$ which is Gaussian. Given $p(F_*|X_*,Y)$ approximated by $q(\mathbf{f}_*|X_*)$ and $p(X_*|Y)$ approximated by $q(X_*)$, the posterior density of $\mathbf{f}_*$ (the noise-free version of $\mathbf{y}_*$) is now approximated by

$$p(\mathbf{f}_*|Y) = \int q(\mathbf{f}_*|X_*)q(X_*)dX_*. \tag{32}$$

The specific formulations of the distributions $p(\mathbf{f}_*|\mathbf{u},X_*)$, $q(\mathbf{f}_*|X_*)$ and $q\left(X_*\right)$ are given in Appendix C as a more comprehensive treatment.

However, the integration of $q(\mathbf{f}_*|X_*)$ w.r.t $q(X_*)$ is not analytically feasible. Following Damianou et al. (2011), we give the expectation of $\mathbf{f}_*$ as $\mathcal{E}(\mathbf{f}_*)$ and its element-wise autocovariance as vector $\mathcal{C}(\mathbf{f}_*)$ whose $(\tilde{n}\times d)$th entry is $\mathcal{C}(f_{\tilde{n}d})$ with $\tilde{n}=1,...,N_*$ and $d=1,...,D$.

$$\mathcal{E}(\mathbf{f}_*) = \psi_{1*}\mathbf{b}, \tag{33}$$

$$\mathcal{C}(f_{\tilde{n}d}) = \mathbf{b}^\top(\psi_{2\tilde{n}}^d - (\psi_{1\tilde{n}}^d)^\top\psi_{1\tilde{n}}^d)\mathbf{b} + \psi_{0*}^d - \mathrm{Tr}\left[(\mathbf{K}_{\mathbf{u},\mathbf{u}}^{-1} - (\mathbf{K}_{\mathbf{u},\mathbf{u}} + \beta\psi_2)^{-1})\psi_{2*}^d\right], \tag{34}$$

where $\psi_{1*} = \langle\mathbf{K}_{\mathbf{f}_*,\mathbf{u}}\rangle_{q(X_*)}$, $\mathbf{b} = \beta(\mathbf{K}_{\mathbf{u},\mathbf{u}} + \beta\psi_2)^{-1}\psi_1^\top\mathbf{y}$, $\psi_{1\tilde{n}}^d = \langle\mathbf{K}_{f_{\tilde{n}d},\mathbf{u}}\rangle_{q(\mathbf{x}_{\tilde{n}})}$, $\psi_{2\tilde{n}}^d = \langle\mathbf{K}_{\mathbf{u},f_{\tilde{n}d}}$ $\mathbf{K}_{f_{\tilde{n}d},\mathbf{u}}\rangle_{q(\mathbf{x}_{\tilde{n}})}$, $\psi_{0*}^d = \mathrm{Tr}(\langle\mathbf{K}_{\mathbf{f}_{*d},\mathbf{f}_{*d}}\rangle_{q(X_*)})$ and $\psi_{2*}^d = \langle\mathbf{K}_{\mathbf{u},\mathbf{f}_{*d}}\mathbf{K}_{\mathbf{f}_{*d},\mathbf{u}}\rangle_{q(X_*)}$. Since Y_* is the noisy version of F_*, the expectation and element-wise autocovariance of Y_* are $\mathcal{E}(\mathbf{y}_*) = \mathcal{E}(\mathbf{f}_*)$ and $\mathcal{C}(\mathbf{y}_*) = \mathcal{C}(\mathbf{f}_*) + \beta^{-1}\mathbf{1}_{N_*D}$, where $\mathbf{y}_*^\top = [\mathbf{y}_{*1}^\top,...,\mathbf{y}_{*D}^\top]$.

4.2 Prediction with Time and Partial Observations

Prediction with time and partial observations can be divided into two cases. In one case, we need to predict $Y_*^m \in \mathbb{R}^{N_*\times D_m}$ which represents the outputs on missing dimensions, given $Y_*^{pt} \in \mathbb{R}^{N_*\times D_p}$ which represents the outputs observed on partial dimensions. We call this task reconstruction. In the other case, we need to predict $Y_*^n \in \mathbb{R}^{N_m^*\times D}$ which means the outputs at the next time, given $Y_*^{pv} \in \mathbb{R}^{N_p^*\times D}$ which means the outputs observed on all dimensions at the previous time. We call this task forecasting.

For the task of reconstruction, we should compute the posterior density of Y_*^m which is given below (Damianou et al., 2011)

$$p(Y_*^m|Y_*^{pt},Y) = \int p(Y_*^m|F_*^m)p(F_*^m|X_*,Y_*^{pt},Y)p(X_*|Y_*^{pt},Y)dF_*^m dX_*. \tag{35}$$

$p(X_*|Y_*^{pt},Y)$ is approximated by a Gaussian distribution $q(X_*)$ whose parameters need to be optimized for the sake of considering the partial observations Y_*^{pt}. This requires maximizing a new lower bound of $\log p(Y,Y_*^{pt})$ which can be expressed as

$$\begin{aligned}\tilde{\mathcal{L}} = &\log\left[\frac{\beta^{\frac{ND+N_*D_p}{2}}|\mathbf{K}_{\mathbf{u},\mathbf{u}}|^{\frac{1}{2}}}{(2\pi)^{\frac{ND+N_*D_p}{2}}|\beta\tilde{\psi}_2+\mathbf{K}_{\mathbf{u},\mathbf{u}}|^{\frac{1}{2}}}\exp\{-\frac{1}{2}\tilde{\mathbf{y}}^\top\tilde{W}\tilde{\mathbf{y}}\}\right]\\ &-\frac{\beta\tilde{\psi}_0}{2} + \frac{\beta}{2}\mathrm{Tr}(\mathbf{K}_{\mathbf{u},\mathbf{u}}^{-1}\tilde{\psi}_2) - \mathbf{KL}[q(X,X_*)||p(X,X_*)|\mathbf{t},\mathbf{t}_*)],\end{aligned} \tag{36}$$

where $\tilde{W} = \beta I - \beta^2 \tilde{\psi}_1 (\beta \tilde{\psi}_2 + \mathbf{K}_{\mathbf{u},\mathbf{u}})^{-1} \tilde{\psi}_1^\top$, $\tilde{\psi}_0 = \mathrm{Tr}(\langle \mathbf{K}_{\tilde{\mathbf{f}},\tilde{\mathbf{f}}} \rangle_{q(X,X_*)})$, $\tilde{\psi}_1 = \langle \mathbf{K}_{\tilde{\mathbf{f}},\mathbf{u}} \rangle_{q(X,X_*)}$ and $\tilde{\psi}_2 = \langle \mathbf{K}_{\mathbf{u},\tilde{\mathbf{f}}} \mathbf{K}_{\tilde{\mathbf{f}},\mathbf{u}} \rangle_{q(X,X_*)}$. The vector $\tilde{\mathbf{y}}$ splices the vectorization of matrix Y and the vectorization of matrix Y_*^{pt}, i.e. $\tilde{\mathbf{y}} = [vec(Y); vec(Y_*^{pt})]$. The vector $\tilde{\mathbf{f}}$ corresponds to the noise-free version of $\tilde{\mathbf{y}}$. Moreover, parameters of the new variational distribution $q(X, X_*)$ are jointly optimized because of the coupling of X and X_*. Then the marginal distribution $q(X_*)$ is obtained from $q(X, X_*)$. Note that when multiple sequences such as X_* and X are independent, only the separated variational distribution $q(X_*)$ is optimized.

For the task of forecasting, we focus on real-time forecasting for which the outputs are dependent on the previous ones and the training set Y is not used in the prediction stage. The variational distribution $q(X_*)$ can be directly computed as (70). Then the posterior density of Y_*^n is computed as (66), but with Y_* and Y replaced with Y_*^n and Y_*^{pv}, respectively. $\mathcal{E}(\mathbf{y}_*^n)$ is the estimate of the output Y_*^n. An application for forecasting is given in Section 6.3.

4.3 Adaptation to Regression Models

Since the VDM-GPDS can been seen as a multi-layer regression model which regards time indices as inputs and observations as outputs. It can be flexibly extended to solve regression problems. Specifically, the time indices in the dynamical systems are replaced with the observed input data V. In addition, the kernel functions for the latent variables X are replaced by some appropriate functions such as automatic relevance determination (ARD) kernels:

$$\kappa_x(\mathbf{v}, \mathbf{v}') = \sigma_{ard}^2 e^{\frac{1}{2} \sum_{p=1}^{P} \omega_p (v_p - {v'}_p)^2}. \tag{37}$$

Model inference and optimization remain the same except for some changes for model parameters $\boldsymbol{\theta}_x$. Compared with other dependent multi-output regression models such as the CMOGP, the VDM-GPDS can achieve much better performance. This could be attributed to its use of latent layers.

5. Related Work

Damianou et al. (2011) described a GP dynamical system with variational Bayesian inference called VGPDS in which the latent variables X are imposed a GP prior to model the dynamical driving force and capture the high dimensional data's characteristics. After introducing inducing points, the latent variables are variationally integrated out. The outputs of VGPDS are generated from multiple independent GPs with the same latent variables X and the same parameters, resulting in the advantage that VGPDS can handle high dimensional situations. However, the explicit dependency among the multiple outputs is ignored in this model, while this kind of dependency is very important for many applications. In contrast, the CMOGP (Álvarez and Lawrence, 2011) and GPRN (Wilson et al., 2012) model the dependency of different outputs through convolved process covariance functions and an adaptive network, respectively. Nevertheless, these two methods are not directly suitable for dynamical system modeling. If applied to dynamical systems with time as inputs, they cannot well capture the complexity of dynamical systems because there is only one nonlinear mapping between the input and output included.

Our model is capable of capturing the dependency among outputs as well as modeling the dynamical characteristics. It is also very different from the GPDM (Wang et al., 2006, 2008) which models the variance of each output with different scale parameters and employs Markov dynamical prior on the latent variables. The Gaussian prior for the latent variables in the VDM-GPDS can model the dynamical characteristics in the systems better than the Markov dynamical prior, since it can model different kinds of dynamics by using different kernels such as using periodic kernels to model periodicity. Moreover, in contrast to the GPDM that estimates the latent variables X through the MAP, the VDM-GPDS integrates out the latent variables with variational methods. This is in the same spirit of the technique used in Damianou et al. (2011), which can provide a principled approach to handle uncertainty in the latent space and determine its principal dimensions automatically. In addition, the multiple outputs in our model are modeled by convolution processes as in Álvarez and Lawrence (2011), which can flexibly capture the correlations among the outputs.

6. Experiments

In this part, we design five experiments to evaluate our model for four different kinds of applications including prediction with only time as inputs, reconstruction of the missing data, real-time forecasting and solving robot inverse dynamics problem. Two experiments are performed on synthetic data and three on real-world data. A number of models such as the CMOGP/COGP, GPDM, VGPDS and VDM-GPDS are tested on the data. The root mean squire error (RMSE) and mean standardized log loss (MSLL) (Rasmussen and Williams, 2006) are taken as the performance measures. In particular, let $\hat{Y}^*$ be the estimate of matrix Y^*, and then the RMSE can be formulated as

$$\mathrm{RMSE}(Y^*, \hat{Y}^*) = \left[\frac{1}{D} \frac{1}{N} \sum_d \sum_n (y_n^{*d} - \hat{y}_n^{*d})^2 \right]^{\frac{1}{2}}. \tag{38}$$

MSLL is the mean negative log probability of all the test data under the learned model Γ and training data Y, which can be formulated as

$$\mathrm{MSLL}(Y^*, \Gamma) = \frac{1}{N} \sum_n \left\{ -\log p(\mathbf{y}_n^* | \Gamma, Y) \right\}. \tag{39}$$

The lower value of the RMSE and MSLL we get, the better the performance of the model is. Our code is implemented based on the framework of publicly available code for the VGPDS and CMOGP.

6.1 Synthetic Data

In this section, we evaluate our method on synthetic data generated from a complex dynamical system. The latent variables X are independently generated by the Ornstein-Uhlenbeck (OU) process (Archambeau et al., 2007)

$$dx_q = -\gamma x_q dt + \sqrt{\sigma^2} dW, \quad q = 1, ..., Q. \tag{40}$$

The outputs Y are generated through a multi-output GP

$$y_d(\mathbf{x}) \sim \mathcal{GP}(0, \kappa_{f_d, f_{d'}}(\mathbf{x}, \mathbf{x}')), \quad d, d' = 1, ..., D, \tag{41}$$

	Spline	**CMOGP**	**GPDM**	**VGPDS**	**VDM-GPDS**
RMSE(y_1)	1.91±0.43	1.75±0.38	1.70±0.18	1.51±0.31	$\mathbf{1.43 \pm 0.23}$
RMSE(y_2)	4.23±1.01	3.46±0.67	3.32±0.27	2.99±0.53	$\mathbf{2.82 \pm 0.35}$
RMSE(y_3)	6.88±1.91	5.19±0.99	4.83±0.28	4.24±0.85	$\mathbf{4.09 \pm 0.59}$
RMSE(y_4)	6.99±1.52	7.50±0.94	5.98±0.55	5.16±0.92	$\mathbf{5.00 \pm 0.60}$

Table 1: Averaged RMSE (%) with the standard deviation (%) for predictions on the output-dependent synthetic data.

	CMOGP	**GPDM**	**VGPDS**	**VDM-GPDS**
MSLL(y_1)	-2.63 ± 0.22	$2.21\times10^4\pm4.86\times10^4$	-2.73 ± 0.08	$\mathbf{-2.79 \pm 0.08}$
MSLL(y_2)	-1.99 ± 0.13	$1.93\times10^4\pm5.23\times10^4$	-2.14 ± 0.15	$\mathbf{-2.18 \pm 0.15}$
MSLL(y_3)	-1.49 ± 0.21	$3.92\times10^4\pm8.17\times10^4$	-1.66 ± 0.24	$\mathbf{-1.66 \pm 0.21}$
MSLL(y_4)	-1.08 ± 0.21	$9.90\times10^4\pm1.98\times10^5$	-1.31 ± 0.41	$\mathbf{-1.32 \pm 0.25}$

Table 2: Averaged MSLL with the standard deviation for predictions on the output-dependent synthetic data.

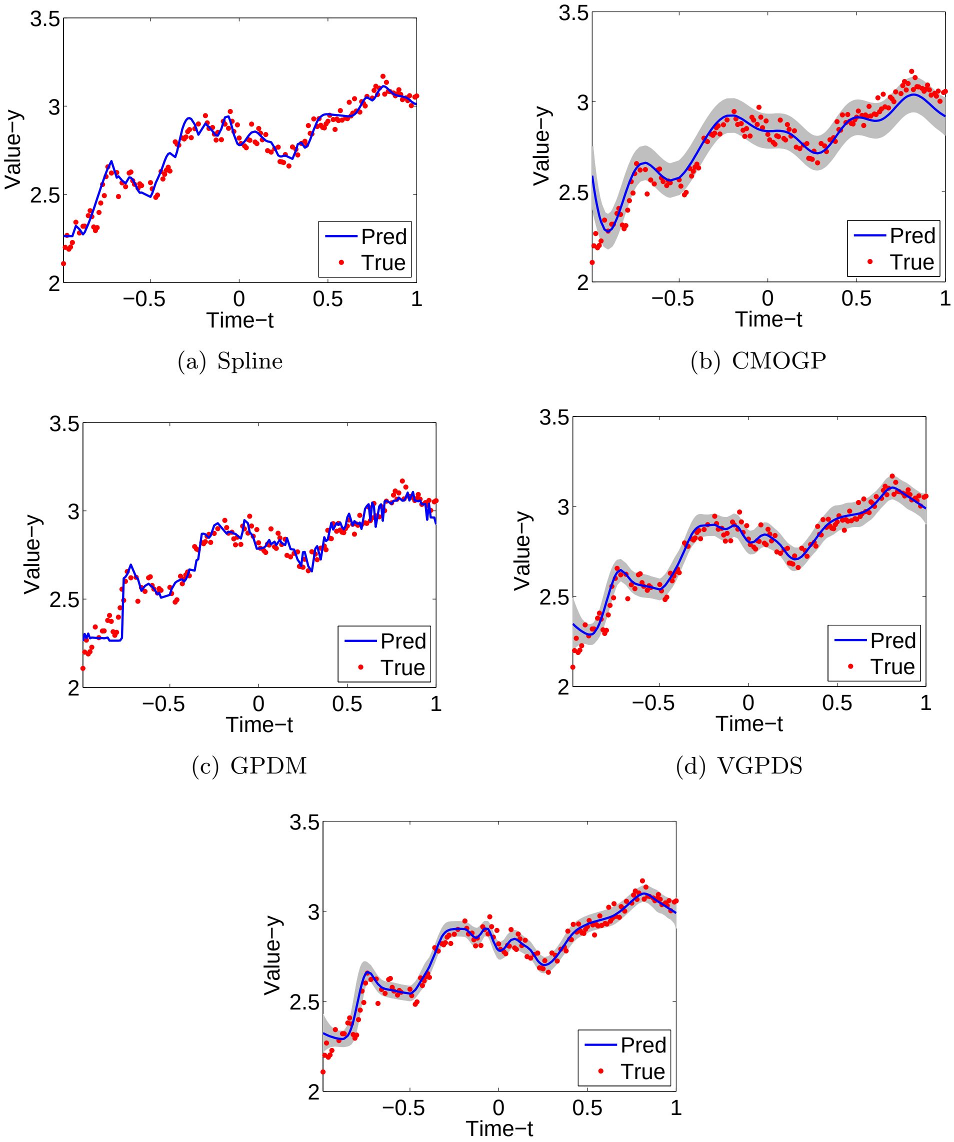
where $\kappa_{f_d,f_{d'}}(\mathbf{x},\mathbf{x}')$ defined in (12) is the multi-output covariance function. In this paper, the number of the latent functions in (9) is set to one, i.e., $K = 1$, which is also the common setting used in Álvarez and Lawrence (2011).

We sample the synthetic data by two steps. First we use the differential equation with parameters $\gamma = 0.5$, $\sigma = 0.01$ to sample $N = 200$, $Q = 2$ latent variables at time interval $[-1, 1]$. Then we sample $D = 4$ dimensional outputs, each of which has 200 observations through the multi-output GP with the following parameters $S_{1,1} = 1$, $S_{2,1} = 2$, $S_{3,1} = 3$, $S_{4,1} = 4$, $P_1 = [5, 1]^\top$, $P_2 = [5, 1]^\top$, $P_3 = [3, 1]^\top$, $P_4 = [2, 1]^\top$ and $\Lambda = [4, 5]^\top$. For approximation, 30 random inducing points are used. In addition, white Gaussian noise is added to each output.

6.1.1 Prediction

Here we evaluate the performance of our method for predicting the outputs given only time over the synthetic data. We randomly select 50 points from each output for training with the remaining 150 points for testing. This is repeated for ten times. The CMOGP, GPDM and VGPDS are performed as comparisons. The cubic spline interpolation (spline for short) is also chosen as a baseline. The latent variables X in the GPDM, VGPDS and VDM-GPDS with two dimensions are initialized by using the principal component analysis on the observations. Moreover, the Matérn 3/2 covariance function is used in the VGPDS and VDM-GPDS.

Table 1 and Table 2 present the RMSE and MSLL for predictions, respectively. The best results are shown in bold. From the tables, we can find that for prediction on the data of each dimension, our model obtains the lowest RMSE and MSLL. We analyze the reasons as follows. First, since the data in this experiment are generated from a complex



(a) Spline

(b) CMOGP

(c) GPDM

(d) VGPDS

(e) VDM-GPDS

Figure 2: Predictions for $y_4(t)$ with the five methods. Pred and True indicate predicted and observed values, respectively. The shaded regions represent two standard deviations for the predictions.

dynamical system that combines two GP mappings, the CMOGP which consists of only one GP mapping cannot capture the complexity well. Moreover, the VDM-GPDS models the explicit dependency among the multiple outputs while the VGPDS and GPDM do not. The assumption of multi-output dependency is appropriate for the generative model. Further,

	Spline	**CMOGP**	**GPDM**	**VGPDS**	**VDM-GPDS**
RMSE(y_1)	3.81±0.82	11.49±0.26	3.82±1.55	$\mathbf{2.18 \pm 0.06}$	2.21±0.06
RMSE(y_2)	2.58±0.68	3.59±0.72	3.45±1.70	2.06±0.19	$\mathbf{2.05 \pm 0.13}$
RMSE(y_3)	3.26±0.90	1.75±0.10	3.57±1.71	$\mathbf{1.68 \pm 0.09}$	1.72±0.12
RMSE(y_4)	9.06±1.17	8.34±10.89	7.10±1.28	4.48±0.23	$\mathbf{4.45 \pm 0.20}$

Table 3: Averaged RMSE (%) with the standard deviation (%) for predictions on the output-independent synthetic data.

	CMOGP	**GPDM**	**VGPDS**	**VDM-GPDS**
MSLL(y_1)	−0.74±0.02	$5.22\times10^2 \pm 5.23\times10^2$	$\mathbf{-2.34 \pm 0.04}$	−2.33±0.04
MSLL(y_2)	−1.91±0.24	$1.10\times10^3 \pm 2.22\times10^3$	−2.36±0.10	$\mathbf{-2.36 \pm 0.13}$
MSLL(y_3)	−2.62±0.05	$2.10\times10^2 \pm 3.43\times10^2$	−2.50±0.08	$\mathbf{-2.52 \pm 0.11}$
MSLL(y_4)	−1.38±0.66	$5.19\times10^2 \pm 1.16\times10^3$	−1.46±0.17	$\mathbf{-1.48 \pm 0.18}$

Table 4: Averaged MSLL with the standard deviation for predictions on the output-independent synthetic data.

the GPDM cannot work well in the case in which data on many time intervals are lost. Prediction with the GPDM results in very high MSLL. To sum up, our model gives the best performance among the five models as expected.

In order to give intuitive representations, we draw one prediction result from the ten experiments in Figure 2 where the shaded regions in 2(b), 2(c), 2(d) and 2(e) represent two standard deviations for the predictions. Through the figures, it is clear that the VDM-GPDS has higher accuracies and smaller variances. Note that the GPDM has very small variances, but low accuracies, which leads to the high MSLL as in Table 2. With all the evaluation measures considered, the VDM-GPDS gives the best performance of prediction with only time as inputs.

In addition, to verify the flexibility of the VDM-GPDS, we perform experiments on the output-independent data which are generated analogously to Section 6.1. In particular, the output-independent data are generated using Equation (41) but with $\kappa_{f_d,f_{d'}}(\mathbf{x},\mathbf{x}') = 0$ for $d \neq d'$ after generating X. Note that the GPDM and VGPDS do not make the assumption of output dependency. The results in terms of RMSE and MSLL are shown in Table 3 and Table 4 where we can see that our model performs as well as the VGPDS and significantly better than the CMOGP and GPDM.

6.1.2 Reconstruction

In this section, we compare the VDM-GPDS with the k-nearest neighbor best (k-NNbest) method which chooses the best k from $\{1, \ldots, 5\}$, the CMOGP, GPDM and VGPDS for recovering missing points given time and partially observed outputs. We set $S_{4,1} = -4$ to generate data in this part, which makes the output y_4 be negatively correlated with the others. We remove all outputs y_1 or y_4 at time interval $[0.5, 1]$ from the 50 training points,

	k-NNbest	**CMOGP**	**GPDM**	**VGPDS**	**VDM-GPDS**
RMSE(y_1)	1.87±0.62	1.90±0.31	2.69±3.67	1.49±0.94	$\mathbf{0.98 \pm 0.34}$
RMSE(y_4)	13.51±2.54	9.31±0.87	12.61±2.43	6.79±6.07	$\mathbf{5.56 \pm 1.88}$

Table 5: Averaged RMSE (%) with the standard deviation (%) for reconstructions of the missing points for y_1 and y_4.

	CMOGP	**GPDM**	**VGPDS**	**VDM-GPDS**
MSLL(y_1)	-1.74±0.20	$1.40\times10^3\pm5.96\times10^4$	-2.29±0.46	$\mathbf{-2.86 \pm 0.09}$
MSLL(y_4)	0.31±0.62	$7.40\times10^4\pm8.34\times10^4$	-1.64±0.69	$\mathbf{-2.35 \pm 0.10}$

Table 6: Averaged MSLL with the standard deviation for reconstructions of the missing points for y_1 and y_4.

resulting in 35 points as training data. Note that the CMOGP considers all the present outputs as the training set while the GPDM, VGPDS and VDM-GPDS only consider the outputs at time interval $[-1, 0.5)$ as the training set.

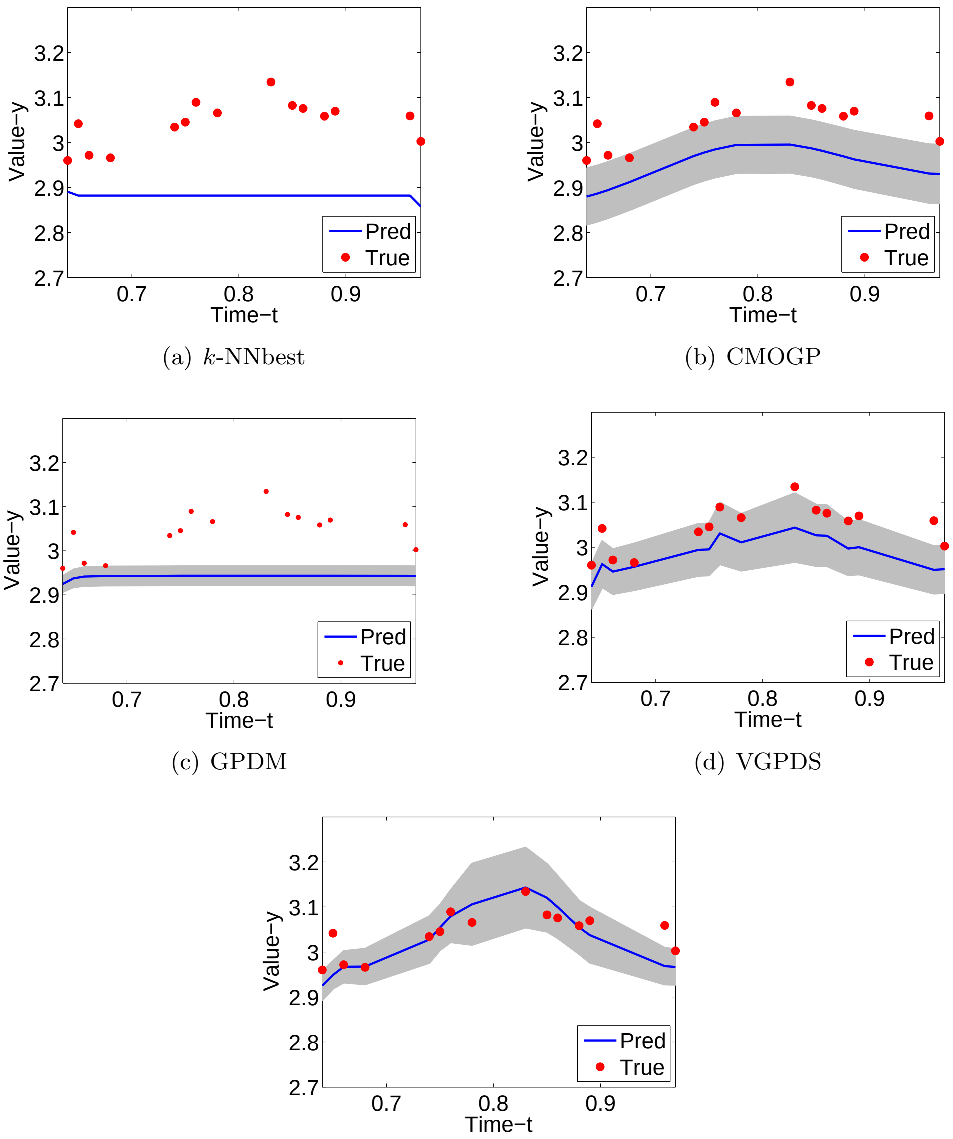
Table 5 and Table 6 show the averaged RMSE and MSLL with the standard deviation for reconstructions of the missing points for y_1 and y_4. The proposed model performs best with the lowest RMSE and MSLL. Specifically, our model can make full use of the present data on some dimensions to reconstruct the missing data through the dependency among outputs. This advantage is shown by comparing with the GPDM and VGPDS. In addition, the two Gaussian process mappings in the VDM-GPDS help to well model the dynamical characteristics and complexity of the data. This advantage is shown by comparing to the CMOGP.

Figure 3 shows one reconstruction result for y_4 from the ten experiments by five different methods. It can be seen that the results of the VDM-GPDS are the closest to the true values among the compared methods. This indicates the superior performance of our model for the reconstruction task.

6.2 Human Motion Capture Data

In order to demonstrate the validity of the proposed model on real-world data, we employ ten sequences of runs/jogs from subject 35 (see Figure 4 for a skeleton) and two sequences of runs/jogs from subject 16 in the CMU motion capture database for the reconstruction task. In particular, our task is to reconstruct the right leg or the upper body of one test sequence on the motion capture data given training sequences. We preprocess the data as in Lawrence (2007) and divide the sequences into training and test data. Nine independent training sequences are all from subject 35 and the remaining three testing sequences are from subject 35 and subject 16 (one from subject 35 and two from subject 16). The average length of each sequence is 40 frames and the output dimension is 59.

We conduct this reconstruction with six different methods, the nearest neighbor in the angle space (NN) and the scaled space (NN sc.) (Taylor et al., 2006), the CMOGP, GPDM,



(a) k-NNbest

(b) CMOGP

(c) GPDM

(d) VGPDS

(e) VDM-GPDS

Figure 3: Reconstructions of the missing points for $y_4(t)$ with the five methods. Pred and True indicate predicted and observed values, respectively. The shaded regions represent two standard deviations for the predictions.

VGPDS and VDM-GPDS. For the CMOGP, periodic time indices with different cycles are used as inputs where the length of each sequence is a cycle. For the GPDM, parameters and latent variables are set as in Wang et al. (2006). For the VGPDS and VDM-GPDS, the RBF kernel is adopted in this set of experiments to construct $\mathbf{K}_{\mathbf{t},\mathbf{t}}$ which is a block-

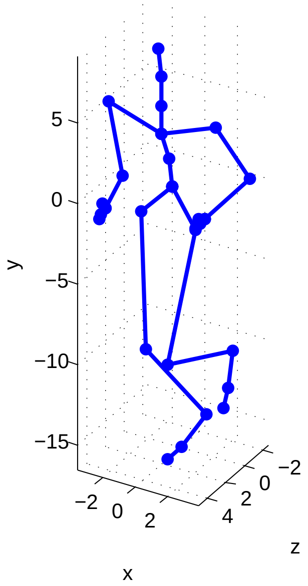


Figure 4: A skeleton of subject 35 running at a moment.

	NN sc.	**NN**	**CMOGP**	**GPDM**	**VGPDS**	**VDM-GPDS**
RMSE(LS)	0.82	0.85	1.15	0.68	0.65	**0.64**
RMSE(LA)	6.75	7.94	13.53	5.61	5.54	**5.30**
RMSE(BS)	1.00	1.40	3.56	0.68	0.66	**0.60**
RMSE(BA)	5.63	9.57	5.02	3.39	2.81	**2.60**
RMSE(LS)	1.03	1.40	1.37	0.91	0.89	**0.85**
RMSE(LA)	10.10	9.73	15.19	8.90	**8.13**	8.65
RMSE(BS)	2.88	3.00	4.70	2.85	3.80	**2.83**
RMSE(BA)	7.45	7.83	8.04	7.13	10.64	**6.69**

Table 7: The RMSE for reconstructions of the missing points of the motion capture data. The values listed above the double line are the results for the one test sequence from subject 35 while the values listed below the double line are the averaged results for the two test sequences from subject 16.

diagonal matrix because the sequences are independent. Moreover, the latent variables X in the VGPDS and VDM-GPDS with nine dimensions are initialized by using principal component analysis on the observations. For parameter optimization of the VDM-GPDS and VGPDS, the maximum numbers of iteration steps are set to be identical.

Table 7 gives the RMSE for reconstructions of the missing points with the six methods. The values listed above the double line are the results for reconstruction of the test sequence from subject 35. The values listed below the double line are the averaged results for reconstruction of the two test sequences from subject 16. LS and LA correspond to the reconstructions of the right leg in the scaled space and angle space. Similarly, BS and

	NN sc.	NN	CMOGP	GPDM	VGPDS	VDM-GPDS
MSLL(Leg)	·	·	1.55	−3.17	−3.10	−**6.94**
MSLL(Body)	·	·	-26.00	−40.78	−**48.19**	− 45.53
MSLL(Leg)	·	·	5.35	11.17	4.60	−**1.03**
MSLL(Body)	·	·	−2.58	121.58	82.95	−**11.54**

Table 8: The MSLL for reconstructions of the missing points of the motion capture data. The values listed above the double line are the results for the one test sequence from subject 35 while the values listed below the double line are the averaged results for the two test sequences from subject 16.

BA correspond to the reconstructions of the upper body in the same two spaces. Table 8 shows the MSLL in the original space for the same reconstructions as in Table 7. As expected, the results on subjects whose motions are used to learn the models show significantly smaller RMSE and MSLL than those for the test motions from subjects not used in the training set. No matter under what circumstances, our model generally outperforms the other approaches. We conjecture that this is because the VDM-GPDS effectively considers both the dynamical characteristics and the dependency among the outputs in the complex dynamical system. Specifically, for the CMU data, the dependency among different parts of the entire body can be well captured by the VDM-GPDS. When only parts of data on each frame of the test sequence are observed, the missing data on the corresponding frame can be recovered by utilizing the observed data. Meanwhile, the GP prior on the latent variables makes sure the continuity and smoothness of movements.

6.3 Traffic Flow Forecasting

The problem addressed here is to predict the future traffic flows of a certain road link. We focus on the short-term predictions, in the time interval of 15 minutes, which is a difficult but very important application. A real urban transportation network is used for this experiment. The raw data are of 25 days, which include 2400 recording points for each road link. Note that the data are not continuous in these 25 days, though data are complete within each recording day. Specifically, the day numbers with recordings are $\{[1 \sim 19], [21], [25 \sim 29]\}$. The first seven days are used for training, and the remaining 18 days are for testing. Figure 5 shows a patch taken from the urban traffic map of highways as in Sun et al. (2006). Each circle node in the sketch map denotes a road link. An arrow shows the direction of the traffic flow, which reaches the corresponding road link from its upstream link. Paths without arrows are of no traffic flow records.

We predict the traffic volumes (vehicles/hour) of the road links (Bb, Ch, Dd, Eb, Fe, Gd, Hi, Ib, Jh, Ka) based on their own historic traffic flows and their direct upstream flows. A VDM-GPDS is trained for each road link. Take Gd as an example. Its three direct upstream flows Fe, Fg and Fh and its own flows are used to construct the four outputs in a VDM-GPDS. We use multiple kernels, including RBF and RBFperiodic to capture the periodicity inherent in the data. The present periodicity in the data contains two cycles. The short cycle is one day and the long cycle is one week. In the prediction stage, we use

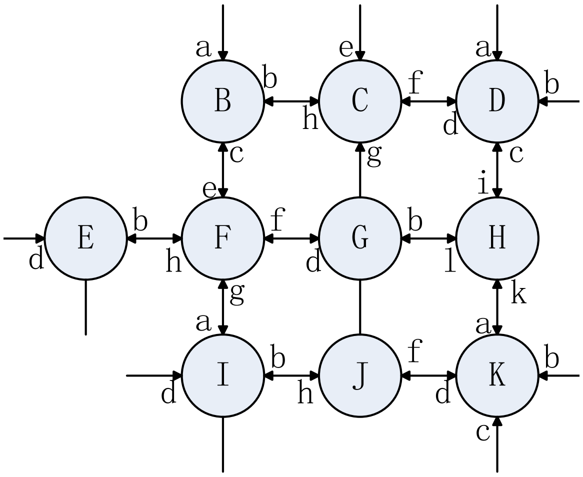


Figure 5: A patch taken from the urban traffic map of highways.

	RW	**VGPDS**	**VDM-GPDS**
RMSE(Bb)	84.95	81.90	**80.88**
RMSE(Ch)	72.49	65.33	**62.08**
RMSE(Dd)	66.45	61.68	**57.97**
RMSE(Eb)	153.46	148.42	**140.69**
RMSE(Fe)	151.16	143.35	**131.7**4
RMSE(Gd)	174.18	162.81	**147.14**
RMSE(Hi)	95.57	92.89	**85.19**
RMSE(Ib)	142.85	129.21	**121.15**
RMSE(Jh)	146.52	141.50	**128.66**
RMSE(Ka)	94.15	88.23	**75.23**

Table 9: The RMSE for forecasting results on the traffic flow data.

the historic traffic flows of the four road links to predict the flows of *Gd* in the next interval. The historic time for forecasting is fixed as four intervals. We compare our model with the Random Walk (RW) and VGPDS. The RW is to forecast the current value using the last value (Williams, 1999), which is chosen as a baseline. According to the descriptions about real-time forecasting in Section 4.2, the VGPDS can be adapted to apply to this experiment. Moreover, in the previous experiments, the VGPDS performs best among the compared models except the VDM-GPDS. Therefore, it is sufficient to compare our model with the RW and VGPDS. Note that the realization of the VGPDS also takes the periodicity into consideration.

Table 9 and Table 10 show the RMSE and MSLL for forecasting results with three methods over the testing sets, respectively. It is obvious that the VDM-GPDS achieves the best performance, even for the road links with large volumes (and large fluctuations) such as *Gd*. This is attributed to the fact that our model well captures both the temporal and spatial dependency of the traffic flow data. In particular, the relationship between the traffic flows of the objective road link and its upstream links is captured by the multi-

	RW	**VGPDS**	**VDM-GPDS**
MSLL(Bb)	·	5.87	**5.85**
MSLL(Ch)	·	6.04	**5.90**
MSLL(Dd)	·	5.64	**5.57**
MSLL(Eb)	·	6.40	**6.33**
MSLL(Fe)	·	6.41	**6.37**
MSLL(Gd)	·	6.53	**6.44**
MSLL(Hi)	·	5.93	**5.90**
MSLL(Ib)	·	6.33	**6.25**
MSLL(Jh)	·	6.38	**6.30**
MSLL(Ka)	·	6.07	**5.86**

Table 10: The MSLL for forecasting results on the traffic flow data.

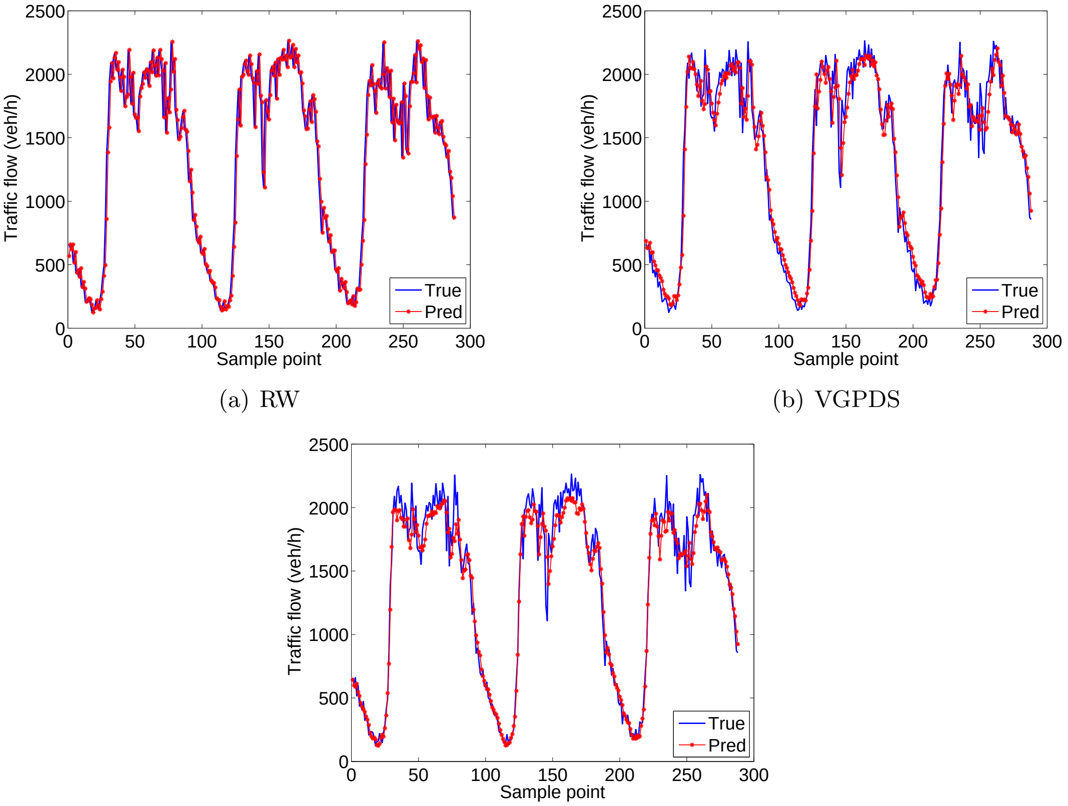
output dependency in the VDM-GPDS; the relationship between the traffic flows of the objective road link and its own historical series is captured by the dynamical characteristics modeled in the VDM-GPDS. Therefore, the entire cause information is well collected by the VDM-GPDS to predict the traffic flows of the objective road link.

To be intuitive, we give the final forecasting results of the performed models for the road link *Gd* in the last three days in Figure 6. The VDM-GPDS has shown great superiority to the compared models. As seen from Figure 6(a) and Figure 6(b), the forecasting results with the RW and VGPDS often lag.

6.4 Robot Inverse Dynamics Problem

The robot inverse dynamics problem is an important task in the robot areas (Sciavicco and Vijayakumar, 2000). For a goal of touching or grasping a subject using a robotic manipulator, it usually needs the following procedures. First, the inverse kinematic calculates the robot joint coordinates given the pose of the end-effector. Then trajectory planing decides a trajectory describing how a robot should move to achieve the desired task. Finally, given the trajectory, i.e., the motion specified by the joint angles, velocities and accelerations, the torques needed at the joints to drive it along the trajectory are computed by the inverse dynamics. What we concerned here is the robot inverse dynamics problem. Analytical models for the inverse dynamics are often infeasible, for example due to uncertainty in the physical parameters of the robot, or the difficulty of modeling frictions. This leads to the need to learn the inverse dynamics by some machine learning methods (Chai et al., 2009; Nguyen and Bonilla, 2014).

We approximate the inverse dynamics model of a 7-degree-of-freedom anthropomorphic robot arm (see Figure 7). The inverse dynamics model of the robot is strongly nonlinear due to a vast amount of superpositions of sine and cosine functions in robot dynamics. The data consist of 21 input dimensions: 7 joint positions, velocities, and accelerations. The goal of learning is to approximate the appropriate torque command of one robot motor in response to the input vector. We choose 4449 data points from the original data set which consists of 48933 data points. We use 100, 300 and 500 points for training, respectively and the rest for testing. All the experiments are repeated for ten times. We consider joint



(a) RW

(b) VGPDS

(c) VDM-GPDS

Figure 6: Forecasting results of the performed models for the road link *Gd* on the last three days.

learning for the two couples, the 2nd and 3rd, the 4th and 7th torques. Note that, the 2nd and 3rd torques are negatively correlated while the 4th and 7th torques are positively correlated.

We adapt our dynamical model to a regression model as described in Section 4.3. In order to demonstrate the performance of our model on the regression problem. We compare our model with the single Gaussian process regression (sGPR), multi-task Gaussian process (MTGP), collaborative multi-output Gaussian process (COGP) and VGPDS. The sGPR is to learn the torque for each joint separately. The MTGP regards the torques from different joints as different tasks. The COGP is a scalable method which is extended from the CMOGP by introducing stochastic variational inference. As the COGP is an enhanced version of the CMOGP for robot inverse dynamics problems (Nguyen and Bonilla, 2014), we do not include the CMOGP in this experiment. For fairness, we set the batch size in the COGP the same as the number of training points. Note that the exact inference is used for the sGPR and MTGP. For the COGP, VGPDS and VDM-GPDS, variational inference is employed and the same size of inducing points are included. Particularly, 15, 20 and

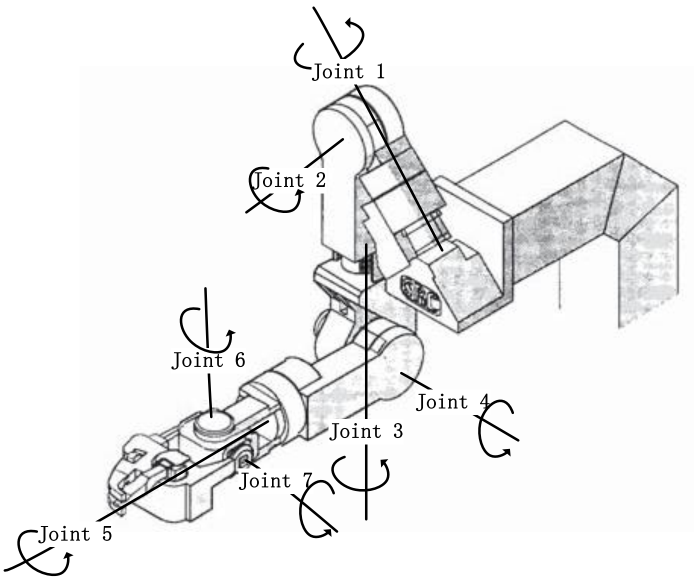


Figure 7: Sketch of the SARCOS dextrous arm (Vijayakumar and Schaal, 2000).

Method	**2nd joint**		**3rd joint**	
($N = 100$)	**RMSE**	**MSLL**	**RMSE**	**MSLL**
sGPR	6.33±1.33	3.85±0.86	4.48±1.04	4.06±1.18
MTGP	5.52±0.79	3.88±0.19	3.20±0.38	3.10±0.27
COGP	5.11±0.44	4.17±0.69	3.18±0.23	3.39±0.38
VGPDS	4.89±0.36	6.20±0.87	2.96±0.26	5.73±0.89
VDM-GPDS	**4.55 ± 0.34**	**2.95 ± 0.14**	**2.68 ± 0.22**	**2.34 ± 0.15**

Method	**4th joint**		**7th joint**	
($N = 100$)	**RMSE**	**MSLL**	**RMSE**	**MSLL**
sGPR	5.01±2.06	4.33±1.54	1.04±0.22	2.76±1.53
MTGP	3.27±0.35	3.44±0.49	0.72±0.07	2.11±0.55
COGP	3.26±0.25	2.68±0.24	0.68±0.05	1.30±0.21
VGPDS	**3.19 ± 0.20**	**2.36 ± 0.08**	**0.65 ± 0.03**	**0.86 ± 0.66**
VDM-GPDS	3.19±0.26	2.44±0.12	0.66±0.04	0.95±0.10

Table 11: Averaged RMSE and MSLL with the standard deviation for robot inverse dynamics learning with 100 training points.

30 inducing points are used for 100, 300 and 500 training points, respectively. For the VDM-GPDS and VGPDS, the dimensionality of the latent space is set to two. Table 11, 12 and 13 show the results for different methods in terms of averaged RMSE and MSLL. For intuition, we also plot the RMSE results in Figure 8.

From the tables and figures, we find that our model performs best on the whole, which confirms that the VDM-GPDS also works well for regression tasks. Comparing the VDM-

Method	**2nd joint**		**3rd joint**	
($N = 300$)	**RMSE**	**MSLL**	**RMSE**	**MSLL**
sGPR	4.19±0.83	2.85±0.32	2.61±0.51	2.44±0.37
MTGP	4.37±0.89	3.36±0.35	2.53±0.56	2.77±0.19
COGP	3.78±0.14	5.08±0.79	2.27±0.09	3.49±0.43
VGPDS	3.67±0.18	2.62±0.04	2.11±0.12	2.06±0.04
VDM-GPDS	**3.52 ± 0.09**	**2.59 ± 0.04**	**2.01 ± 0.06**	**2.00 ± 0.03**

Method	**4th joint**		**7th joint**	
($N = 300$)	**RMSE**	**MSLL**	**RMSE**	**MSLL**
sGPR	2.21±0.37	2.39±0.59	0.55±0.13	0.92±0.51
MTGP	2.01±0.20	2.40±0.12	0.47±0.04	1.27±0.19
COGP	2.25±0.18	2.65±0.25	0.52±0.02	1.71±0.21
VGPDS	2.31±0.38	2.12±0.15	0.48±0.06	0.59±0.10
VDM-GPDS	**1.95 ± 0.09**	**1.92 ± 0.04**	**0.45 ± 0.01**	**0.53 ± 0.04**

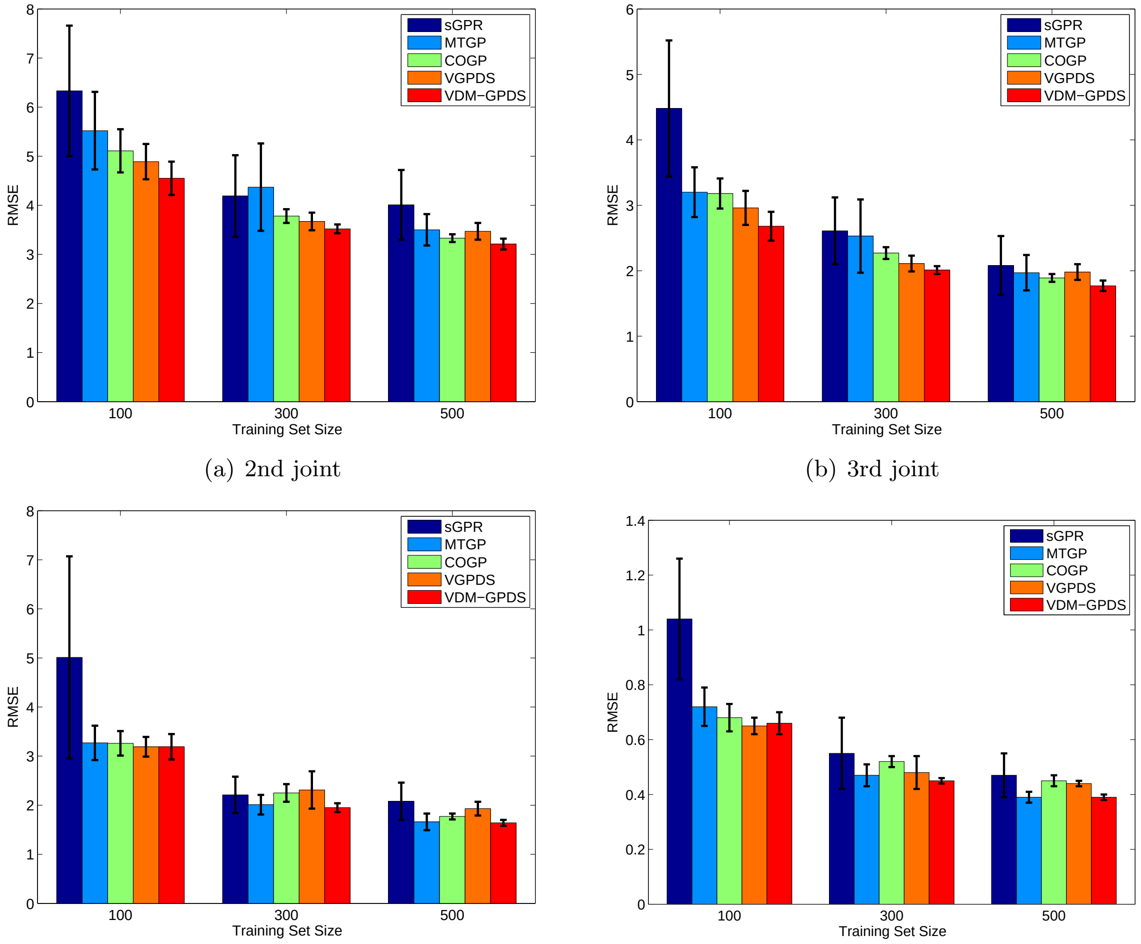
Table 12: Averaged RMSE and MSLL with the standard deviation for robot inverse dynamics learning with 300 training points.

Method	**2nd joint**		**3rd joint**	
($N = 500$)	**RMSE**	**MSLL**	**RMSE**	**MSLL**
sGPR	4.01±0.71	2.78±0.27	2.08±0.45	2.05±0.32
MTGP	3.50±0.32	3.42±0.27	1.97±0.27	2.59±0.24
COGP	3.33±0.08	5.24±0.49	1.89±0.06	3.54±0.34
VGPDS	3.47±0.17	2.57±0.03	1.98±0.12	2.02±0.03
VDM-GPDS	**3.20 ± 0.11**	**2.52 ± 0.04**	**1.77 ± 0.08**	**1.93 ± 0.04**

Method	**4th joint**		**7th joint**	
($N = 500$)	**RMSE**	**MSLL**	**RMSE**	**MSLL**
sGPR	2.08±0.38	2.28±0.51	0.47±0.08	0.76±0.45
MTGP	1.66±0.17	2.28±0.19	0.39±0.02	1.27±0.15
COGP	1.77±0.06	2.46±0.10	0.45±0.02	2.20±0.22
VGPDS	1.93±0.14	2.04±0.08	0.44±0.01	0.55±0.04
VDM-GPDS	**1.64 ± 0.06**	**1.83 ± 0.02**	**0.39 ± 0.01**	**0.44 ± 0.03**

Table 13: Averaged RMSE and MSLL with the standard deviation for robot inverse dynamics learning with 500 training points.

GPDS with the COGP, we further verify the assumption that the latent space can well grasp the characteristics of the data generation. Thus, the VDM-GPDS can well model the inverse dynamics model and make better prediction. Compared with the VGPDS, our model still shows advantages. This is attributed to the assumption of the dependency among



(a) 2nd joint

(b) 3rd joint

(c) 4th joint

(d) 7th joint

Figure 8: Averaged RMSE with the standard deviation for the 2nd, 3rd, 4th and 7th joint torques prediction. (better in color)

the multiple outputs. No matter for dynamical system modeling or static data regression, the proposed model is reasonable and applicable.

6.5 Performance and Efficiency Analysis

The proposed VDM-GPDS outperforms several previous methods for predicting outputs and recovering missing points for dynamical systems. In order to quantify the superior results, we evaluate the performance increases with the averaged performance increasing ratio to the VGPDS in terms of RMSE. The increasing ratios of the four experiments (Sections 6.1.1, 6.1.2, 6.2, 6.3 and 6.4) are 4.49%, 26.17%, 10.40%, 7.35% and 7.97%, respectively.

However, high effectiveness often comes together with low efficiency. The VDM-GPDS is a four-layer GP system that is more complex than the conventional methods. Particularly,

	CMOGP	GPDM	VGPDS	VDM-GPDS
computational complexity	$O(D^3N^3)$	$O(N^3)$	$O(M^2NQ)$	$O(M^2NDQK)$
execution time	50.7	26.8	74.2	181.1

Table 14: Computational complexities and execution time (in ms) for different models.

since the proposed method explicitly models the dependency among outputs, the dependent multi-output covariance matrix in the VDM-GPDS is a full matrix with size $ND \times ND$ and operations involving it cannot be factorized. This is in contrast to the independent multi-output covariance matrix in the GPDM and VGPDS, which is block-diagonal. As in Titsias (2009), inducing points are employed for the variational inference for the VDM-GPDS. The number of the inducing points M is much smaller than that of the data points N, which can improve the computational efficiency. For the VDM-GPDS, the most time-consuming calculation is to compute ψ_2 whose computational complexity is $O(M^2NDQK)$.

In order to give clear comparisons in terms of efficiency, we list the computational complexities of four models and the execution time (in ms) of one step for learning the models on the synthetic data in Table 14. Through the table, we find that the VDM-GPDS costs a lot. Nevertheless, our model can obtain high performance improvements as discussed above. We believe that getting performance improvements is worth the time cost.

7. Conclusion

In this paper, we have proposed a dependent multi-output GP for modeling complex dynamical systems. We give the reasonable assumption that the different outputs of the systems are generally dependent. The convolved process covariance function is employed to model the dependency among all the data points across all the outputs. We adapt the variational inference method involving inducing points to our model so that the latent variables are variationally integrated out. The model and variational parameters are jointly optimized with the scaled conjugate gradient method. Through small adaptations, our model can handle regression problems.

Modeling the possible dependency among multiple outputs can help to make better predictions. The effectiveness of the proposed model for complex dynamical systems is empirically demonstrated through multiple experiments. However, when the dimensionality of the output is very high, our model may take a long time to converge. This opens the possibility for future work to accelerate training for high dimensional dynamical systems.

Acknowledgments

The corresponding author Shiliang Sun would like to thank supports from National Natural Science Foundation of China under Projects 61370175 and 61075005, and Shanghai Knowledge Service Platform Project (No. ZF1213).

Appendix A. Derivation of the Lower Bound

In order to approximately compute the marginal likelihood $p(Y|\mathbf{t})$, we compute the variational lower bound of it by involving the variational distribution $q(F, U, X|Z)$. The variational lower bound $\mathcal{L}$ can be expressed as

$$\begin{aligned}\mathcal{L} &= \int q(F, U, X|Z) \log \frac{p(Y, F, U, X|\mathbf{t}, Z)}{q(F, U, X|Z)} dX dU dF \\ &= \int p(\mathbf{f}|\mathbf{u}, X, Z) q(\mathbf{u}) q(X) \log \frac{p(\mathbf{y}|\mathbf{f}) p(\mathbf{u}|Z) p(X|\mathbf{t})}{q(\mathbf{u}) q(X)} d\mathbf{f} d\mathbf{u} dX,\end{aligned} \tag{42}$$

since

$$p(Y, F, U, X|\mathbf{t}, Z) = p(\mathbf{y}|\mathbf{f}) p(\mathbf{f}|\mathbf{u}, X, Z) p(\mathbf{u}|Z) p(X|\mathbf{t}), \tag{43}$$

and

$$q(F, U, X|Z) = p(\mathbf{f}|\mathbf{u}, X, Z) q(\mathbf{u}) q(X). \tag{44}$$

For neatness, the above expression is split into two parts as $\mathcal{L} = \hat{\mathcal{L}} - \mathbf{KL}[q(X)||p(X|\mathbf{t})]$. Specifically, $\hat{\mathcal{L}}$ is expressed by

$$\hat{\mathcal{L}} = \int p(\mathbf{f}|\mathbf{u}, X, Z) q(\mathbf{u}) q(X) \log \frac{p(\mathbf{y}|\mathbf{f}) p(\mathbf{u}|Z)}{q(\mathbf{u})} d\mathbf{f} d\mathbf{u} dX. \tag{45}$$

$\mathbf{KL}[q(X)||p(X|\mathbf{t})]$ is the relative entropy of $q(X)$ and $p(X|\mathbf{t})$, expressed as

$$\begin{aligned}\mathbf{KL}[q(X)||p(X|\mathbf{t})] = &\int q(X) \log \frac{q(X)}{p(X|\mathbf{t})} \\ = &\frac{Q}{2} \log |\mathbf{K}_{\mathbf{t},\mathbf{t}}| - \frac{1}{2} \sum_{q=1}^{Q} \log |S_q| \\ &+ \frac{1}{2} \sum_{q=1}^{Q} [\mathrm{Tr}(\mathbf{K}_{\mathbf{t},\mathbf{t}}^{-1} S_q) + \mathrm{Tr}(\mathbf{K}_{\mathbf{t},\mathbf{t}}^{-1} \boldsymbol{\mu}_q \boldsymbol{\mu}_q^{\top})] + const,\end{aligned} \tag{46}$$

since

$$p(X|\mathbf{t}) = \prod_{q=1}^{Q} \mathcal{N}(\mathbf{x}_q|\mathbf{0}, \mathbf{K}_{\mathbf{t},\mathbf{t}}), \tag{47}$$

and

$$q(X) = \prod_{q=1}^{Q} \mathcal{N}(\mathbf{x}_q|\boldsymbol{\mu}_q, S_q). \tag{48}$$

So far, $\mathbf{KL}[q(X)||p(X|\mathbf{t})]$ can be calculated analytically as the above, we need to calculate $\hat{\mathcal{L}}$. By using the facts that $\log \frac{p(\mathbf{y}|\mathbf{f}) p(\mathbf{u}|Z)}{q(\mathbf{u})} = \log p(\mathbf{y}|\mathbf{f}) + \log \frac{p(\mathbf{u}|Z)}{q(\mathbf{u})}$ and $\int p(\mathbf{f}|\mathbf{u}, X, Z) d\mathbf{f} = 1$, $\hat{\mathcal{L}}$ is converted into

$$\begin{aligned}\hat{\mathcal{L}} = &\int q(\mathbf{u}) q(X) \int p(\mathbf{f}|\mathbf{u}, X, Z) \log p(\mathbf{y}|\mathbf{f}) d\mathbf{f} d\mathbf{u} dX \\ &+ \int q(\mathbf{u}) q(X) \log \frac{p(\mathbf{u}|Z)}{q(\mathbf{u})} d\mathbf{u} dX.\end{aligned} \tag{49}$$

We know that $p(\mathbf{y}|\mathbf{f})$ and $p(\mathbf{f}|\mathbf{u}, X, Z)$ are both Gaussian, and then

$$\begin{aligned}&\int p(\mathbf{f}|\mathbf{u}, X, Z) \log p(\mathbf{y}|\mathbf{f}) d\mathbf{f} \\ &= \log \mathcal{N}(\mathbf{y}|\mathbf{K}_{\mathbf{f},\mathbf{u}}\mathbf{K}_{\mathbf{u},\mathbf{u}}^{-1}\mathbf{u}, \beta^{-1}I) - \frac{\beta}{2}\mathrm{Tr}(\mathbf{K}_{\mathbf{f},\mathbf{f}} - \mathbf{K}_{\mathbf{f},\mathbf{u}}\mathbf{K}_{\mathbf{u},\mathbf{u}}^{-1}\mathbf{K}_{\mathbf{u},\mathbf{f}}).\end{aligned} \tag{50}$$

Thus $\hat{\mathcal{L}}$ in (49) can be simplified as

$$\begin{aligned}\hat{\mathcal{L}} =& \int q(\mathbf{u})q(X) \log \frac{\mathcal{N}(\mathbf{y}|\mathbf{a}, B)p(\mathbf{u})}{q(\mathbf{u})} d\mathbf{u} dX \\ &- \int \frac{\beta}{2}\mathrm{Tr}(\mathbf{K}_{\mathbf{f},\mathbf{f}} - \mathbf{K}_{\mathbf{f},\mathbf{u}}\mathbf{K}_{\mathbf{u},\mathbf{u}}^{-1}\mathbf{K}_{\mathbf{u},\mathbf{f}})q(X)dX,\end{aligned} \tag{51}$$

where $\mathbf{a} = \mathbf{K}_{\mathbf{f},\mathbf{u}}\mathbf{K}_{\mathbf{u},\mathbf{u}}^{-1}\mathbf{u}$ and $B = \beta^{-1}I$. By changing the integration order, we get

$$\begin{aligned}\hat{\mathcal{L}} =& \int q(\mathbf{u})[\log \frac{e^{\langle \log \mathcal{N}(\mathbf{y}|\mathbf{a}, B)\rangle_{q(X)}} p(\mathbf{u})}{q(\mathbf{u})}] d(\mathbf{u}) \\ &- \frac{\beta}{2}\mathrm{Tr}(\langle \mathbf{K}_{\mathbf{f},\mathbf{f}}\rangle_{q(X)} - \langle \mathbf{K}_{\mathbf{f},\mathbf{u}}\mathbf{K}_{\mathbf{u},\mathbf{u}}^{-1}\mathbf{K}_{\mathbf{u},\mathbf{f}}\rangle_{q(X)}).\end{aligned} \tag{52}$$

We compute the optimal bound using the reserved Jensen's inequality as in Titsias and Lawrence (2010). This gives

$$\begin{aligned}\hat{\mathcal{L}} \leq& \log \int e^{\langle \log \mathcal{N}(\mathbf{y}|\mathbf{a}, B)\rangle_{q(X)}} p(\mathbf{u}) d\mathbf{u} \\ &- \frac{\beta}{2}\mathrm{Tr}(\langle \mathbf{K}_{\mathbf{f},\mathbf{f}}\rangle_{q(X)}) + \frac{\beta}{2}\mathrm{Tr}(\langle \mathbf{K}_{\mathbf{f},\mathbf{u}}\mathbf{K}_{\mathbf{u},\mathbf{u}}^{-1}\mathbf{K}_{\mathbf{u},\mathbf{f}}\rangle_{q(X)}).\end{aligned} \tag{53}$$

The optimal distribution $q(\mathbf{u})$ that gives rise to this lower bound is given by $q(\mathbf{u}) = e^{\langle \log \mathcal{N}(\mathbf{y}|\mathbf{K}_{\mathbf{f},\mathbf{u}}\mathbf{K}_{\mathbf{u},\mathbf{u}}^{-1}\mathbf{u}, \beta^{-1}I)\rangle_{q(X)}} p(\mathbf{u})$, which is analytically Gaussian

$$q(\mathbf{u}) \propto \mathcal{N}(\beta \mathbf{y}^{\top}\psi_1 \mathbf{K}_{\mathbf{u},\mathbf{u}}(\beta\psi_2 + \mathbf{K}_{\mathbf{u},\mathbf{u}})^{-1}\psi_1^{\top}\mathbf{y}, \mathbf{K}_{\mathbf{u},\mathbf{u}}(\beta\psi_2 + \mathbf{K}_{\mathbf{u},\mathbf{u}})^{-1}), \tag{54}$$

where $\psi_0 = \mathrm{Tr}(\langle \mathbf{K}_{\mathbf{f},\mathbf{f}}\rangle_{q(X)})$, $\psi_1 = \langle \mathbf{K}_{\mathbf{f},\mathbf{u}}\rangle_{q(X)}$ and $\psi_2 = \langle \mathbf{K}_{\mathbf{u},\mathbf{f}}\mathbf{K}_{\mathbf{f},\mathbf{u}}\rangle_{q(X)}$. The closed-form of the lower bound of the approximated marginal log-likelihood defined as $\mathcal{L}$ is given by

$$\begin{aligned}\mathcal{L} =& \log \left[\frac{\beta^{\frac{ND}{2}} |\mathbf{K}_{\mathbf{u},\mathbf{u}}|^{\frac{1}{2}}}{|2\pi|^{\frac{ND}{2}} |\beta\psi_2 + \mathbf{K}_{\mathbf{u},\mathbf{u}}|^{\frac{1}{2}}} \exp\{-\frac{1}{2}\mathbf{y}^{\top}W\mathbf{y}\} \right] \\ &- \frac{\beta\psi_0}{2} + \frac{\beta}{2}\mathrm{Tr}(\mathbf{K}_{\mathbf{u},\mathbf{u}}^{-1}\psi_2) - \mathbf{KL}[q(X)||p(X|\mathbf{t})],\end{aligned} \tag{55}$$

where $W = \beta I - \beta^2\psi_1(\beta\psi_2 + \mathbf{K}_{\mathbf{u},\mathbf{u}})^{-1}\psi_1^{\top}$. Given the above, we can obtain the final formulation of the lower bound in (21) which has the similar formulation with Damianou et al. (2011). But actually they are not the same.

Appendix B. Gradients with Respect to the Parameters

The parameters involved in the proposed model include the model parameters $\{\beta, \boldsymbol{\theta}_x, \boldsymbol{\theta}_f\}$ and the variational parameters $\{\{\bar{\boldsymbol{\mu}}_q, \lambda_q\}_{q=1}^{Q}, Z\}$ after reparameterizing $\{\boldsymbol{\mu}_q, S_q\}_{q=1}^{Q}$. All the parameters are jointly optimized by maximizing the lower bound in (21) with the scaled conjugate gradient method. Here we give the detailed gradients of all the parameters. Note that in our model the dimensionality of the latent variable U is one (K=1). So the statistics such as $S_{d,k}$, Λ_k, W are changed to s_d, Λ, M here. In order to simplify the expressions, we define $\Sigma_{\psi_0} = 2P_d + \Lambda$, $\Sigma_{\psi_1} = P_d + \Lambda + S_n$, $\Sigma_{\psi_{21}} = 2(P_d + \Lambda)$, $\Sigma_{\psi_{22}} = \frac{P_d+\Lambda}{2} + S_n$, $C^{-1} = \beta^{-1}\mathbf{K}_{\mathbf{u},\mathbf{u}} + \psi_2$. $\mathbf{z}$ is equivalent to $\mathbf{z}_m$ and $\mathbf{z}'$ is equivalent to $\mathbf{z}_{m'}$. The symbol $|_q$ after a matrix means the qth column of the matrix.

Because of the reparameterization, we need to calculate the gradients of $\hat{\mathcal{L}}$ with respect to $\boldsymbol{\mu}_q$, S_q and then obtain the gradients of $\mathcal{L}$ with respect to $\bar{\boldsymbol{\mu}}_q$, $\boldsymbol{\lambda}_q$ using (28) and (29). Given that $\mathbf{KL}[q(X)||p(X|\mathbf{t})]$ does not involve the parameters $\boldsymbol{\theta}_f$, β and Z, its gradients with respect to $\boldsymbol{\theta}_f$, β and Z are zero. Therefore, $\frac{\partial\mathcal{L}}{\partial\boldsymbol{\theta}_f} = \frac{\partial\hat{\mathcal{L}}}{\partial\boldsymbol{\theta}_f}$, $\frac{\partial\mathcal{L}}{\partial\beta} = \frac{\partial\hat{\mathcal{L}}}{\partial\beta}$ and $\frac{\partial\mathcal{L}}{\partial Z} = \frac{\partial\hat{\mathcal{L}}}{\partial Z}$.

First, we give the gradients of $\hat{\mathcal{L}}$ with respect to $\boldsymbol{\mu}_q$, S_q, P_d, s_d through the formulation

$$\frac{\partial\hat{\mathcal{L}}}{\partial\theta} = -\frac{\beta}{2}\frac{\partial\psi_0}{\partial\theta} + \beta\mathrm{Tr}[\frac{\partial\psi_1^\top}{\partial\theta}\mathbf{y}\mathbf{y}^\top\psi_1 C] + \frac{\beta}{2}\mathrm{Tr}[\frac{\partial\psi_2}{\partial\theta}(\mathbf{K}_{\mathbf{u},\mathbf{u}}^{-1} - \frac{C}{\beta} - C\psi_1^\top\mathbf{y}\mathbf{y}^\top\psi_1 C)], \tag{56}$$

where θ represents μ_{nq}, S_{nq}, P_{dq} and s_d. The detailed derivatives of ψ_0, ψ_1 and ψ_2 with respect to μ_{nq}, S_{nq}, P_{dq} and s_d are different. The derivatives of ψ_0, $(\psi_1)_{vm}$ and $(\psi_2)_{mm'}$ with respect to μ_{nq} are

$$\begin{aligned}
\frac{\partial\psi_0}{\partial\mu_{nq}} &= 0,\\
\frac{\partial(\psi_1)_{vm}}{\partial\mu_{nq}} &= s_d\mathcal{N}(\mathbf{z}|\boldsymbol{\mu}_n, \Sigma_{\psi_1})((\mathbf{z}-\boldsymbol{\mu}_n)^\top\Sigma_{\psi_1}^{-1}|_q),\\
\frac{\partial(\psi_2)_{mm'}}{\partial\mu_{nq}} &= \sum_{d=1}^{D} {s_d}^2\mathcal{N}(\mathbf{z}|\mathbf{z}', \Sigma_{\psi_{21}})\mathcal{N}(\frac{\mathbf{z}+\mathbf{z}'}{2}|\boldsymbol{\mu}_n, \Sigma_{\psi_{22}})((\frac{\mathbf{z}+\mathbf{z}'}{2}-\boldsymbol{\mu}_n)^\top\Sigma_{\psi_{22}}^{-1}|_q).
\end{aligned} \tag{57}$$

The derivatives of ψ_0, $(\psi_1)_{vm}$ and $(\psi_2)_{mm'}$ with respect to S_{nq} are

$$\begin{aligned}
\frac{\partial\psi_0}{\partial S_{nq}} &= 0,\\
\frac{\partial(\psi_1)_{vm}}{\partial S_{nq}} &= s_d\mathcal{N}(\mathbf{z}|\boldsymbol{\mu}_n, \Sigma_{\psi_1})(-\frac{|\Sigma_{\psi_1}|^{-1}}{2}\frac{\partial|\Sigma_{\psi_1}|}{\partial S_{nq}} - \frac{1}{2}(\mathbf{z}-\boldsymbol{\mu}_n)^\top\frac{\partial\Sigma_{\psi_1}^{-1}}{\partial S_{nq}}(\mathbf{z}-\boldsymbol{\mu}_n)),\\
\frac{\partial(\psi_2)_{mm'}}{\partial S_{nq}} &= \sum_{d=1}^{D} {s_d}^2\mathcal{N}(\mathbf{z}|\mathbf{z}', \Sigma_{\psi_{21}})\mathcal{N}(\frac{\mathbf{z}+\mathbf{z}'}{2}|\boldsymbol{\mu}_n, \Sigma_{\psi_{22}})\\
&\qquad(-\frac{|\Sigma_{\psi_{22}}|^{-1}}{2}\frac{\partial|\Sigma_{\psi_{22}}|}{\partial S_{nq}} - \frac{1}{2}(\frac{\mathbf{z}+\mathbf{z}'}{2}-\boldsymbol{\mu}_n)^\top\frac{\partial\Sigma_{\psi_{22}}^{-1}}{\partial S_{nq}}(\frac{\mathbf{z}+\mathbf{z}'}{2}-\boldsymbol{\mu}_n)).
\end{aligned} \tag{58}$$

The derivatives of ψ_0, $(\psi_1)_{vm}$ and $(\psi_2)_{mm'}$ with respect to P_{dq} are

$$\begin{aligned}
\frac{\partial\psi_0}{\partial P_{dq}} =&\sum_{d=1}^{D}-\frac{N}{2}\frac{s_d s_d}{|2\pi|^{\frac{Q}{2}}|\Sigma_{\psi_0}|^{\frac{3}{2}}}\frac{\partial|\Sigma_{\psi_0}|}{\partial P_{dq}},\\
\frac{\partial(\psi_1)_{vm}}{\partial P_{dq}} =&s_d\mathcal{N}(\mathbf{z}|\boldsymbol{\mu}_n,\Sigma_{\psi_1})(-\frac{|\Sigma_{\psi_1}|^{-1}}{2}\frac{\partial|\Sigma_{\psi_1}|}{\partial P_{dq}}-\frac{1}{2}(\mathbf{z}-\boldsymbol{\mu}_n)^\top\frac{\partial\Sigma_{\psi_1}^{-1}}{\partial P_{dq}}(\mathbf{z}-\boldsymbol{\mu}_n)),\\
\frac{\partial(\psi_2)_{mm'}}{\partial P_{dq}} =&\sum_{n=1}^{N}{s_d}^2\mathcal{N}(\mathbf{z}|\mathbf{z}',\Sigma_{\psi_{21}})\mathcal{N}(\frac{\mathbf{z}+\mathbf{z}'}{2}|\boldsymbol{\mu}_n,\Sigma_{\psi_{22}})\\
&(-\frac{|\Sigma_{\psi_{21}}|^{-1}}{2}\frac{\partial|\Sigma_{\psi_{21}}|}{\partial P_{dq}}-\frac{1}{2}(\mathbf{z}-\mathbf{z}')^\top\frac{\partial\Sigma_{\psi_{21}}^{-1}}{\partial P_{dq}}(\mathbf{z}-\mathbf{z}')\\
&-\frac{|\Sigma_{\psi_{22}}|^{-1}}{2}\frac{\partial|\Sigma_{\psi_{22}}|}{\partial P_{dq}}-\frac{1}{2}(\frac{\mathbf{z}+\mathbf{z}'}{2}-\boldsymbol{\mu}_n)^\top\frac{\partial\Sigma_{\psi_{22}}^{-1}}{\partial P_{dq}}(\frac{\mathbf{z}+\mathbf{z}'}{2}-\boldsymbol{\mu}_n)).
\end{aligned}\tag{59}$$

The derivatives of ψ_0, $(\psi_1)_{vm}$ and $(\psi_2)_{mm'}$ with respect to s_d are

$$\begin{aligned}
\frac{\partial\psi_0}{\partial s_d} &= \frac{2Ns_d}{|2\pi|^{\frac{Q}{2}}|\Sigma_{\psi_0}|^{\frac{1}{2}}},\\
\frac{\partial(\psi_1)_{vm}}{\partial s_d} &= \mathcal{N}(\mathbf{z}|\boldsymbol{\mu}_n,\Sigma_{\psi_1}),\\
\frac{\partial(\psi_2)_{mm'}}{\partial s_d} &= \sum_{n=1}^{N}2s_d\mathcal{N}(\mathbf{z}|\mathbf{z}',\Sigma_{\psi_{21}})\mathcal{N}(\frac{\mathbf{z}+\mathbf{z}'}{2}|\boldsymbol{\mu}_n,\Sigma_{\psi_{22}}).
\end{aligned}\tag{60}$$

Then, we give the gradients of $\mathcal{L}$ with respect to Λ and Z through the formulation

$$\begin{aligned}
\frac{\partial\mathcal{L}}{\partial\theta} =& -\frac{\beta}{2}\frac{\partial\psi_0}{\partial\theta}+\beta\mathrm{Tr}[\frac{\partial\psi_1^\top}{\partial\theta}\mathbf{y}\mathbf{y}^T\psi_1 C]+\frac{\beta}{2}\mathrm{Tr}[\frac{\partial\psi_2}{\partial\theta}(\mathbf{K}_{\mathbf{u},\mathbf{u}}^{-1}-C\psi_1^\top\mathbf{y}\mathbf{y}^\top\psi_1 C-\beta^{-1}C)]\\
&+\frac{1}{2}\mathrm{Tr}[\frac{\partial\mathbf{K}_{\mathbf{u},\mathbf{u}}}{\partial\theta}(\mathbf{K}_{\mathbf{u},\mathbf{u}}^{-1}-C\psi_1^\top\mathbf{y}\mathbf{y}^\top\psi_1 C-\beta^{-1}C-\beta\mathbf{K}_{\mathbf{u},\mathbf{u}}^{-1}\psi_2\mathbf{K}_{\mathbf{u},\mathbf{u}}^{-1}],
\end{aligned}\tag{61}$$

where θ represents Λ_q, Z_{mq}. The detailed derivatives of the ψ_0, ψ_1, ψ_2 and $\mathbf{K}_{\mathbf{u},\mathbf{u}}$ with respect to Λ_q, Z_{mq} are given separately. The derivatives of ψ_0, $(\psi_1)_{vm}$, $(\psi_2)_{mm'}$ and $(\mathbf{K}_{\mathbf{u},\mathbf{u}})_{mm'}$

with respect to Λ_q are

$$
\begin{aligned}
\frac{\partial \psi_0}{\partial \Lambda_q} =& \sum_{d=1}^{D} -\frac{N}{2}\frac{s_d s_d}{|2\pi|^{\frac{Q}{2}}|\Sigma_{\psi_0}|^{\frac{3}{2}}}\frac{\partial |\Sigma_{\psi_0}|}{\partial \Lambda_q},\\
\frac{\partial (\psi_1)_{vm}}{\partial \Lambda_q} =& s_d \mathcal{N}(\mathbf{z}|\boldsymbol{\mu}_n, \Sigma_{\psi_1})(-\frac{1}{2}|\Sigma_{\psi_1}|^{-1}\frac{\partial |\Sigma_{\psi_1}|}{\partial \Lambda_q} - \frac{1}{2}(\mathbf{z}-\boldsymbol{\mu}_n)^\top \frac{\partial \Sigma_{\psi_1}^{-1}}{\partial \Lambda_q}(\mathbf{z}-\boldsymbol{\mu}_n)),\\
\frac{\partial (\psi_2)_{mm'}}{\partial \Lambda_q} =& \sum_{n=1}^{N}\sum_{d=1}^{D} {s_d}^2 \mathcal{N}(\mathbf{z}|\mathbf{z}', \Sigma_{\psi_{21}})\mathcal{N}(\frac{\mathbf{z}+\mathbf{z}'}{2}|\boldsymbol{\mu}_n, \Sigma_{\psi_{22}})\\
& \qquad (-\frac{\partial |\Sigma_{\psi_{21}}|}{2|\Sigma_{\psi_{21}}|\partial \Lambda_q} - \frac{1}{2}(\mathbf{z}-\mathbf{z}')^\top \frac{\partial \Sigma_{\psi_{21}}^{-1}}{\partial \Lambda_q}(\mathbf{z}-\mathbf{z}')\\
& \qquad - \frac{\partial |\Sigma_{\psi_{22}}|}{2|\Sigma_{\psi_{22}}|\partial \Lambda_q} - \frac{1}{2}(\frac{\mathbf{z}+\mathbf{z}'}{2}-\boldsymbol{\mu}_n)^\top \frac{\partial \Sigma_{\psi_{22}}^{-1}}{\partial \Lambda_q}(\frac{\mathbf{z}+\mathbf{z}'}{2}-\boldsymbol{\mu}_n)),\\
\frac{\partial (\mathbf{K}_{\mathbf{u},\mathbf{u}})_{mm'}}{\partial \Lambda_q} =& \mathcal{N}(\mathbf{z}|\mathbf{z}', \Lambda)(-\frac{1}{2}|\Lambda|^{-1}\frac{\partial |\Lambda|}{\partial \Lambda_q} - \frac{1}{2}(\mathbf{z}-\mathbf{z}')^\top \frac{\partial \Lambda^{-1}}{\partial \Lambda_q}(\mathbf{z}-\mathbf{z}')).
\end{aligned}
\tag{62}
$$

The derivatives of ψ_0, $(\psi_1)_{vm}$, $(\psi_2)_{mm'}$ and $(\mathbf{K}_{\mathbf{u},\mathbf{u}})_{mm'}$ with respect to z_{mq} are

$$
\begin{aligned}
\frac{\partial \psi_0}{\partial z_{mq}} =& 0,\\
\frac{\partial (\psi_1)_{vm}}{\partial z_{mq}} =& -s_d \mathcal{N}(\mathbf{z}|\boldsymbol{\mu}_n, \Sigma_{\psi_1})((\mathbf{z}-\boldsymbol{\mu}_n)^\top \Sigma_{\psi_1}^{-1}|_q),\\
\frac{\partial (\psi_2)_{mm'}}{\partial z_{mq}} =& \sum_{d=1}^{D} {s_d}^2 \mathcal{N}(\mathbf{z}|\mathbf{z}', \Sigma_{\psi_{21}})\mathcal{N}(\frac{\mathbf{z}+\mathbf{z}'}{2}|\boldsymbol{\mu}_n, \Sigma_{\psi_{22}})\\
& \qquad (-\frac{1}{2}(\frac{\mathbf{z}+\mathbf{z}'}{2}-\boldsymbol{\mu}_n)^\top \Sigma_{\psi_{22}}^{-1}|_q + (\mathbf{z}-\mathbf{z}')^\top \Sigma_{\psi_{21}}^{-1}|_q),\\
\frac{\partial (\mathbf{K}_{\mathbf{u},\mathbf{u}})_{mm'}}{\partial z_{mq}} =& -\frac{1}{\Lambda_q}\mathcal{N}(\mathbf{z}|\mathbf{z}', \Lambda)(z_{mq} - z_{m'q}).
\end{aligned}
\tag{63}
$$

Finally, we give the gradients of $\mathcal{L}$ with respect to β and $\boldsymbol{\theta}_x$ as follows.

$$
\begin{aligned}
\frac{\partial \mathcal{L}}{\partial \beta} =& \frac{1}{2}[\mathrm{Tr}(\mathbf{K}_{\mathbf{u},\mathbf{u}}^{-1}\psi_2) + (V-M)\beta^{-1} - \psi_0 - \mathrm{Tr}(\mathbf{y}\mathbf{y}^\top) + \mathrm{Tr}(C\psi_1^\top \mathbf{y}\mathbf{y}^\top \psi_1)\\
& + \beta^{-2}\mathrm{Tr}(\mathbf{K}_{\mathbf{u},\mathbf{u}}C) + \beta^{-1}\mathrm{Tr}(\mathbf{K}_{\mathbf{u},\mathbf{u}}C\psi_1^\top \mathbf{y}\mathbf{y}^\top \psi_1 C)],
\end{aligned}
\tag{64}
$$

$$
\begin{aligned}
\frac{\partial \mathcal{L}}{\partial \boldsymbol{\theta}_x} =& \sum_{q=1}^{Q}\mathrm{Tr}[[-\frac{1}{2}(\hat{B}_q\mathbf{K}_{\mathbf{t},\mathbf{t}}\hat{B}_q + \bar{\boldsymbol{\mu}}_q\bar{\boldsymbol{\mu}}_q^T) + (I-\hat{B}_q\mathbf{K}_{\mathbf{t},\mathbf{t}})\frac{\partial \hat{\mathcal{L}}}{\partial S_q}(I-\hat{B}_q\mathbf{K}_{\mathbf{t},\mathbf{t}})^\top]\frac{\partial \mathbf{K}_{\mathbf{t},\mathbf{t}}}{\partial \boldsymbol{\theta}_x}]\\
& + \left(\frac{\partial \hat{\mathcal{L}}}{\partial \boldsymbol{\mu}_q}\right)^\top \frac{\partial \mathbf{K}_{\mathbf{t},\mathbf{t}}}{\partial \boldsymbol{\theta}_x}\bar{\boldsymbol{\mu}}_q,
\end{aligned}
\tag{65}
$$

where $\hat{B}_q = \Lambda_q^{\frac{1}{2}}(I + \Lambda_q^{\frac{1}{2}}\mathbf{K}_{\mathbf{t},\mathbf{t}}\Lambda_q^{\frac{1}{2}})^{-1}\Lambda_q^{\frac{1}{2}}$.

Appendix C. Derivations of Prediction with Only Time

With the parameters as well as time $\mathbf{t}$ and $\mathbf{t}_*$ omitted, the posterior density for prediction is given by

$$p(Y_*|Y) = \int p(Y_*|F_*)p(F_*|X_*, Y)p(X_*|Y)dF_*dX_*, \tag{66}$$

where $F_* \in \mathbb{R}^{N_* \times D}$ denotes the set of latent variables (the noise-free version of Y_*) and $X_* \in \mathbb{R}^{N_* \times Q}$ represents the latent variables in the low dimensional space.

The distribution $p(F_*|X_*, Y)$ in (66) is approximated by the variational distribution

$$p\left(F_*|X_*, Y\right) \approx q(\mathbf{f}_*|X_*) = \int p(\mathbf{f}_*|\mathbf{u}, X_*)q(\mathbf{u})d\mathbf{u}, \tag{67}$$

where $\mathbf{f}_*^\top = [\mathbf{f}_{*1}^\top, ..., \mathbf{f}_{*D}^\top]$, and $p(\mathbf{f}_*|\mathbf{u}, X_*)$ is Gaussian with the formulation

$$p(\mathbf{f}_*|\mathbf{u}, X_*) = \mathcal{N}(\mathbf{f}_*|\mathbf{K}_{\mathbf{f}_*,\mathbf{u}}\mathbf{K}_{\mathbf{u},\mathbf{u}}^{-1}\mathbf{u}, \mathbf{K}_{\mathbf{f}_*,\mathbf{f}_*} - \mathbf{K}_{\mathbf{f}_*,\mathbf{u}}\mathbf{K}_{\mathbf{u},\mathbf{u}}^{-1}\mathbf{K}_{\mathbf{u},\mathbf{f}_*}). \tag{68}$$

Since the optimal setting for $q(\mathbf{u})$ in our variational framework is also found to be Gaussian, $q(\mathbf{f}_*|X_*)$ is Gaussian that can be computed analytically

$$\begin{aligned} q(\mathbf{f}_*|X_*) =& \mathcal{N}(\beta\mathbf{K}_{\mathbf{f}_*,\mathbf{u}}(\mathbf{K}_{\mathbf{u},\mathbf{u}} + \beta\psi_2)^{-1}\psi_1^\top\mathbf{y}, \\ & \mathbf{K}_{\mathbf{f}_*,\mathbf{u}}(\mathbf{K}_{\mathbf{u},\mathbf{u}} + \beta\psi_2)^{-1}\mathbf{K}_{\mathbf{f}_*,\mathbf{u}}^\top + \mathbf{K}_{\mathbf{f}_*,\mathbf{f}_*} - \mathbf{K}_{\mathbf{f}_*,\mathbf{u}}, \mathbf{K}_{\mathbf{u},\mathbf{u}}^{-1}\mathbf{K}_{\mathbf{f}_*,\mathbf{u}}^\top). \end{aligned} \tag{69}$$

The distribution $p\left(X_*|Y\right)$ in (66) is approximated by the variational distribution $q\left(X_*\right)$ which is Gaussian and can be explicitly formulated as

$$q(X_*) = \mathcal{N}(\boldsymbol{\mu}_{X_*}, \boldsymbol{\Sigma}_{X_*}), \tag{70}$$

where $\boldsymbol{\mu}_{X_*}$ is composed of column vector $\boldsymbol{\mu}_{\mathbf{x}_{*q}}$ and block-diagonal matrix $\boldsymbol{\Sigma}_{X_*}$ has diagonal element $\boldsymbol{\Sigma}_{\mathbf{x}_{*q}}$ with

$$\begin{aligned} \boldsymbol{\mu}_{\mathbf{x}_{*q}} &= \mathbf{K}_{\mathbf{t}_*,\mathbf{t}}\mathbf{K}_{\mathbf{t},\mathbf{t}}^{-1}\boldsymbol{\mu}_q, & (71) \\ \boldsymbol{\Sigma}_{\mathbf{x}_{*q}} &= \mathbf{K}_{\mathbf{t}_*,\mathbf{t}_*} - \mathbf{K}_{\mathbf{t}_*,\mathbf{t}}\mathbf{K}_{\mathbf{t},\mathbf{t}}^{-1}(\mathbf{K}_{\mathbf{t},\mathbf{t}_*} - S_q\mathbf{K}_{\mathbf{t},\mathbf{t}}^{-1}\mathbf{K}_{\mathbf{t},\mathbf{t}_*}). & (72) \end{aligned}$$

Given $p(F_*|X_*, Y)$ approximated by $q(\mathbf{f}_*|X_*)$ and $p(X_*|Y)$ approximated by $q(X_*)$, the posterior density of $\mathbf{f}_*$ (the noise-free version of $\mathbf{y}_*$) is now approximated by

$$p(\mathbf{f}_*|Y) = \int q(\mathbf{f}_*|X_*)q(X_*)dX_*. \tag{73}$$

So far, following Damianou et al. (2011), the expectation of $\mathbf{f}_*$and its element-wise autocovariance are given in (33) and (34).

References

M. A. Álvarez and N. D. Lawrence. Computationally efficient convolved multiple output Gaussian processes. *Journal of Machine Learning Research*, 12:1459–1500, 2011.

M. A. Álvarez, D. Luengo, and N. D. Lawrence. Latent force models. In *Proceedings of the 12th International Conference on Artificial Intelligence and Statistics*, pages 9–16, 2009.

M. A. Álvarez, D. Luengo, M. K. Titsias, and N. D. Lawrence. Efficient multioutput Gaussian processes through variational inducing kernels. In *Proceedings of the 13th International Conference on Artificial Intelligence and Statistics*, pages 25–32, 2010.

M. A. Álvarez, J. Peters, B. Schölkopf, and N. D. Lawrence. Switched latent force models for movement segmentation. *Advances in Neural Information Processing Systems*, 23: 55–63, 2011.

M. A. Álvarez, L. Rosasco, and N. D. Lawrence. Kernels for vector-valued functions: a review. *Foundations and Trends® in Machine Learning*, 4:195–266, 2012.

M. A. Álvarez, D. Luengo, and N. D. Lawrence. Linear latent force models using Gaussian processes. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 35:2693–2705, 2013.

C. Archambeau, D. Cornford, M. Opper, and J. Shawe-Taylor. Gaussian process approximations of stochastic differential equations. *Journal of Machine Learning Research Workshop and Conference Proceedings*, 1:1–16, 2007.

C. M. Bishop. *Patten Recognition and Machine Learning.* Springer, New York, 2006.

E. V. Bonilla, K. M. A. Chai, and C. K. I. Williams. Multi-task Gaussian process prediction. *Advances in Neural Information Processing Systems*, 20:153–160, 2007.

P. Boyle. *Gaussian Process for Regression and Optimisation.* PhD thesis, Victoria University of Wellington, 2007.

K. M. A. Chai, C. K. I. Williams, S. Klanke, and S. Vijayakumar. Multi-task Gaussian process learning of robot inverse dynamics. *Advances in Neural Information Processing Systems*, 21:265–272, 2009.

L. Csató and M. Opper. Sparse representation for Gaussian process models. *Advances in Neural Information Processing Systems*, 13:444–450, 2001.

A. C. Damianou, M. K. Titsias, and N. D. Lawrence. Variational Gaussian process dynamical systems. *Advances in Neural Information Processing Systems*, 24:2510–2518, 2011.

A. C. Damianou, M. K. Titsias, and N. D. Lawrence. Variational inference for uncertainty on the inputs of Gaussian process models. `http://arxiv.org/abs/1409.2287`, 2014.

M. P. Deisenroth and S. Mohamed. Expectation propagation in Gaussian process dynamical systems. *Advances in Neural Information Processing Systems*, 25:2618–2626, 2012.

N. Gamage, T. C. Kuang, R. Akmeliawati, and S. Demidenko. Gaussian process dynamical models for hand gesture interpretation in sign language. *Pattern Recognition Letters*, 32: 2009–2014, 2011.

J. Hartikainen and S. Särkkä. Sequential inference for latent force models. `http://arxiv.org/abs/1202.3730`, 2012.

G. E. Henter, M. R. Frean, and W. B. Kleijn. Gaussian process dynamical models for nonparametric speech representation and synthesis. In *Proceedings of IEEE International Conference on Acoustics, Speech and Signal Processing*, pages 4505–4508, 2012.

N. D. Lawrence. Gaussian process latent variable models for visualisation of high dimensional data. *Advances in Neural Information Processing Systems*, 17:329–336, 2004.

N. D. Lawrence. Probabilistic non-linear principal component analysis with Gaussian process latent variable models. *Journal of Machine Learning Research*, 6:1783–1816, 2005.

N. D. Lawrence. Learning for larger dataset with the Gaussian process latent variable model. In *Proceedings of the 11th International Workshop on Artificial Intelligence and Statistics*, pages 243–250, 2007.

M. Lázaro-gredilla. Bayesian warped Gaussian processes. *Advances in Neural Information Processing Systems*, 25:1628–1636, 2012.

J. Luttinen and A. Ilin. Efficient Gaussian process inference for short-scale spatio-temporal modeling. In *Proceedings of the 15th International Conference on Artificial Intelligence and Statistics*, pages 741–750, 2012.

V. T. Nguyen and E. Bonilla. Collaborative multi-output Gaussian processes. In *Proceedings of Uncertainty in Artificial Intelligence*, pages 1–10, 2014.

M. Opper and A. Archambeau. The variational Gaussian approximation revisited. *Neural Computation*, 21:786–792, 2009.

H. Park, S. Yun, S. Park, J. Kim, and C. D. Yoo. Phoneme classification using constrained variational Gaussian process dynamical system. *Advances in Neural Information Processing Systems*, 25:2015–2023, 2012.

J. Quiñonero-Candela and C. E. Rasmussen. A unifying view of sparse approximate Gaussian process regression. *Journal of Machine Learning Research*, 6:1939–1959, 2005.

C. E. Rasmussen and C. K. I. Williams. *Gaussian Process for Machine Learning.* MIT Press, 2006.

L. Sciavicco and B. Vijayakumar. *Modelling and Control of Robot Manipulators.* Springer, 2000.

E. Snelson and Z. Ghahramani. Sparse Gaussian process using pseudo-inputs. *Advances in Neural Information Processing Systems*, 18:444–450, 2006.

E. Snelson, Z. Ghahramani, and C. Rasmussen. Warped Gaussian processes. *Advances in Neural Information Processing Systems*, 16:337–344, 2003.

S. Sun, C. Zhang, and G. Yu. A Bayesian network approach to traffic flow forecasting. *IEEE Transactions on Intelligent Transportation Systems*, 7:124–132, 2006.

G. W. Taylor, G. E. Hinton, and S. Roweis. Modeling human motion using binary latent variables. *Advances in Neural Information Processing Systems*, 19:1345–1352, 2006.

M. E. Tipping and C. M. Bishop. Probabilistic principal component analysis. *Journal of the Royal Statistical Society*, 61:611–622, 1999.

M. K. Titsias. Variational learning of inducing variables in sparse Gaussian processes. In *Proceedings of the 12th International Conference on Artificial Intelligence and Statistics*, pages 567–574, 2009.

M. K. Titsias and N. D. Lawrence. Bayesian Gaussian process latent variable model. In *Proceedings of the 13th International Conference on Artificial Intelligence and Statistics*, pages 844–851, 2010.

R. Urtasun, D. J. Fleet, and P. Fua. 3D people tracking with Gaussian process dynamic models. In *Proceedings of IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, pages 238–245, 2006.

S. Vijayakumar and S. Schaal. Locally weighted projection regression: An O(n) algorithm for incremental real time learning in high dimensional space. In *Proceedings of the 17th International Conference on Machine Learning*, pages 1079–1086, 2000.

J. M. Wang, D. J. Fleet, and A. Hertzmann. Gaussian process dynamical models. *Advances in Neural Information Processing Systems*, 19:1441–1448, 2006.

J. M. Wang, D. J. Fleet, and A. Hertzmann. Gaussian process dynamical models for human motion. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 30:283–398, 2008.

B. M. Williams. *Modeling and forecasting vehicular traffic flow as a seasonal stochastic time series process*. PhD thesis, University of Virginia, 1999.

A. G. Wilson, D. A. Knowles, and Z. Ghahramani. Gaussian process regression networks. In *Proceedings of the 29th International Conference on Machine Learning*, pages 599–606, 2012.

J. Zhao and S. Sun. Variational dependent multi-output Gaussian process dynamical systems. In *Proceedings of the 17th International Conference on Discovery Science*, pages 350–361, 2014.