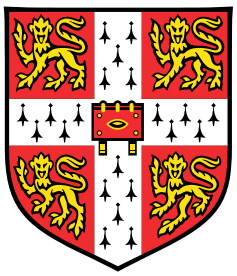
The Gaussian Process Latent Autoregressive Model



Rui Xia

Supervisors:
Dr. Richard E. Turner
Wessel Bruinsma
William Tebbutt

Department of Engineering
University of Cambridge

This dissertation is submitted for the degree of
Master of Philosophy in Machine Learning and Machine Intelligence

Pembroke College August 2020

I would like to dedicate this thesis to my loving parents, who made my studies in Cambridge possible.

Declaration

I, Rui Xia of Pembroke College, being a candidate of the MPhil in Machine Learning and Machine Intelligence, hereby declare that this report and the work described in it are my own work, unaided except as may be specified below, and that the report does not contain material that has already been used to any substantial extent for a comparable purpose.

This dissertation contains 13,391 words excluding declarations, bibliography, photographs and diagrams but including tables, footnotes, figure captions, appendices and abstract.

Software used: All experiments in this project are implemented in Python, utilizing Tensorflow 2.0, GPflow 2.0, built upon codes provided by doubly-stochastic DGP in https://github.com/ICL-SML/Doubly-Stochastic-DGP and modified to adapt to our model.

Rui Xia
August 2020

Acknowledgements

I would like to express my greatest gratitude to my supervisor, Dr. Richard E. Turner. Without your guidance and dedicated help every week, I would never accomplish this research project. I have learnt a lot about how to think comprehensively and broaden my mind like a real researcher and explore every possibility. I would also like to thank Wessel Bruinsma and William Tebbutt for all the intriguing discussions and feedback of experimental results, especially to Wessel for giving me a lot of datasets they used in their previous project on which this project is based. Despite the hard time, all my supervisors gave continuous help across the world which I will always be grateful for.

I would also like to thank the MLMI community and their invaluable friendships. Finally, I would give my greatest thanks to my parents who financially and mentally supported me during this hard time.

Abstract

Multi-output problems are now an extensively active area facing the rising need of transferring knowledge across related outputs. Multi-output Gaussian Processes are particularly important that utilize efficiencies and elegance of Gaussian Process and extend its modelling power from single-output to multi-output.

In this project, we focus on the Gaussian Process Autoregressive Regression (GPAR) model that explicitly exploit dependencies between outputs (Requeima et al., 2018). We extend the model to a fully Bayesian inference version which replaces the former denoising approximation used by GPAR and handles noisy outputs or missing data well by producing more robust results. The inference scheme also enables our model to deal with non-Gaussian likelihood and even combinations of different likelihoods, which allows for a tractable variational bounds that scales-up to large datasets. The expected advantages are validated through extensive experiments using synthetic and real datasets. We also put our novel model in an unifying framework of the multi-output Gaussian Processes literature, comparing existing state-of-art method with respect to modeling power, how latent process being shared and approximations required.

Table of contents

List of figures

List of tables

Chapter 1

Introduction

Traditional supervised learning has shown great power at solving single-output problems, such as binary classification of identifying spam among emails and regression problems such as to predict gross merchandise volume in the e-commerce platform. However, since the increasing trends of today's complex decision making, learning paradigms that simultaneously predict multiple outputs at once are at a pressing need. Due to frequent upgrading of technology and personalised system, there are many mechanisms that deal with multiple complex factors. For example, a mobile phone application that captures information about mobility, communications, and interactions in social media at the same time would need a good solution to learn these related tasks and infer future possible behaviors or missing recordings due to rare malfunction of devices. Moreover, medical signals measured of different body parts or weather conditions measured in different geographical locations would require a systematic way of learning that discovers similarities between patterns and help to leverage knowledge. Multi-outputs problems often appear in many different forms, such that they either differ in data types or ways how each output correlate and interact with each other. The diverse data types include real-valued multi-target regression (Borchani et al., 2015), multi-label classification (Zhang et al., 2013) where output variables are binary, and the heterogeneous case where a mix of continuous, categorical, or discrete variables are of interests (Moreno-Muñoz et al., 2018). For example, when human behavior is of interest, active use or non-use of social software and distance from home would correspond to the heterogeneous case where one output is binary and the other continuous. The correlation also appears in different ways, such that one output might depend quite simply on inputs but depend on certain other outputs in a complex way. On another aspect, outputs can either share similar marginal distribution, such as image or audio data, or they can be marginally heterogeneous and require separate modelling of "inter-" and "intra-" differences (Ma et al., 2020, Carlson et al., 2010). The sophisticated dependencies between these outputs need

structured inference and can be modeled by different methods.

Gaussian Process (GP) is a powerful model that defines probability distributions over functions, where Bayesian inference will be convenient to achieve or approximate in a wide range of tasks, including regression, classification, and state-space model (Wilk et al., 2020). Typically, GPs are designed for single-output problems that out-performs other methods in providing uncertainty over predictions but increasing research on multi-output GPs (MOGP) has also shown its popularity and generalization in adapting to the arisen multi-output fields. The first history of MOGP appears as co-kriging (Chiles et al., 2009) widely used in the geostatistic community and also evolves as multi-task learning or transfer learning within the machine learning group. The key focus of MOGP is to exploit the dependencies between outputs in a way that latent processes will share information and achieve better performance for all tasks. In this project, we would provide a brief comprehensive view of MOGP, compare and contrast existing methods. Particularly, one of the MOGPs that explicitly treats outputs as inputs, unlike in the co-kriging case where latent processes are implicitly combined using a matrix, is called the Gaussian Process Autoregressive Regression (GPAR) studied by Requeima et al. (2018). We focus on further generalising GPAR to deal with noisy or missing values in the output with more caution and enable it to model non-Gaussian likelihoods and even heterogeneous data as mentioned before. We utilize the approximation inference scheme raised by Salimbeni et al. (2017) which is motivated to solve the intractability introduced by non-Gaussian mappings in deep Gaussian Processes.

1.1 Thesis Contribution

We present the Gaussian Process Latent Autoregressive model (GPLAR), combining ideas in deep GPs and MOGPs literature. In order to solve the deficiency in the original GPAR such that noisy outputs are directly used as inputs and hence leads to larger noise in a subsequent stage, hidden variables are introduced corresponding to noiseless, unobserved but true latent function evaluations that require fully Bayesian inference. Since direct links between outputs could also be seen as a nested composition of GP priors, one can easily find similar structures in deep GPs. We utilize the doubly stochastic variational inference scheme in deep GPs, proposing a free-energy term by introducing inducing points at each output level.

- Firstly, we study and compare GPAR and GPLAR's performance over different levels of observation noise and our method successfully solves the problem of misbehavior of the original GPAR model when observation noise is large. We further extend GPLAR to deal with non-Gaussian likelihoods and show its superiority using real datasets.

- Secondly, we realize GPAR's poor performance when there are missing values in the first few output levels, as the original GPAR is sensitive to the sequence order of outputs. Analysis of possible reasons and a new version of GPLAR inspired by bi-directional Recurrent Neural Networks is proposed.
- Thirdly, we review and analyze the similarities and differences among the models proposed in the MOGPs' literature through the views of how latent processes are shared, linearity or non-linearity mappings between outputs.

1.2 Thesis Structure

This project is organized as follows. In Chapter 2, we first review the basics of Gaussian Process, including the widely-used sparse approximation inference scheme. Then we introduce ideas behind GPAR and extend it to the latent variable version, after which, approximation strategies utilized in deep GPs are reviewed and incorporated with our method. At the end of this chapter, Section. 2.5 summarizes related work in a unified framework, where clear advantages of our method compared with other works are presented. In Chapter 3, we discuss the theoretical details of the modeling process, including inducing points optimizations and kernel selections. We further interpret the uncertainty estimates and elaborate the fact that GPLAR and bi-directional approach can be combined together. In Chapter 4, we assess our methods by testing them on synthetic data sets and real data sets, comparing them with previous GP models dealing with multi-outputs. Lastly, conclusions, limitations of the proposed model, and comments over future research are made in Chapter 5.

Chapter 2

Gaussian Process Latent Autoregressive Model

This section provides a brief overview of the non-parametric model, GP, and its sparse approximation strategies which are widely used to deal with intractability and large computations. A complete description of the GPAR model and discussion of its deficiencies will be followed, and a new proposed model that deals with these deficiencies is then introduced. The new proposed model, GPLAR, requires a careful approximation scheme which we borrow ideas from deep GPs literature. We then put the new model into a comprehensive framework of multi-output GPs literature, listing flaws and advantages of different approaches.

2.1 Gaussian Process

A Gaussian Process is a generalization of a multivariate Gaussian distribution to infinitely many variables. One can also view a Gaussian Process as defining a distribution over functions, where inference and learning directly take place in the function space (Rasmussen, 2003). Due to the elegant nature of Gaussian, such that both the conditionals and the marginals of a joint Gaussian are again Gaussian, GP allows for specification on property of the concerned function at a finite number of points, ignoring the infinitely many other points without losing any information. Its combination of the Bayesian paradigm and non-parametric modelling makes it attractive in uniting a sophisticated and consistent view.

Considering a non-linear mapping, $f(\mathbf{x})$, from input $\mathbf{x}$, to a scalar real-valued output y, modelled by a GP, it is fully specified by its mean function, $m(\mathbf{x})$, and covariance function(kernel), $k(\mathbf{x}, \mathbf{x}')$ as follows,

$$f \sim \mathcal{GP}(m(\mathbf{x}), k(\mathbf{x}, \mathbf{x}'))$$
$$m(\mathbf{x}) = \mathbb{E}[f(\mathbf{x})]$$
$$k(\mathbf{x}, \mathbf{x}') = \mathbb{E}[(f(\mathbf{x}) - m(\mathbf{x}))(f(\mathbf{x}') - m(\mathbf{x}'))]$$

Following the definition, joint distribution of a finite collection of function values is,

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

where $\mathbf{X} = \{\mathbf{x}_n\}_{n=1}^N$ is the training inputs, $[m(\mathbf{X})]_n = m(\mathbf{x}_n)$ and $[\mathbf{K}(\mathbf{X}, \mathbf{X})]_{ij} = k(\mathbf{x}_i, \mathbf{x}_j)$. The mean functions and the kernels encapsulate the prior knowledge about the behaviour of the concerned function. The mean functions describe the average value, while the kernels specify how smooth, wiggly, periodic the function is. Squared-exponential (SE), Rational-quadratic (RQ) and Linear (Lin), and Periodic (Per) are kernels commonly used that determine different generalization properties (shown in Fig. 2.1) of the model.

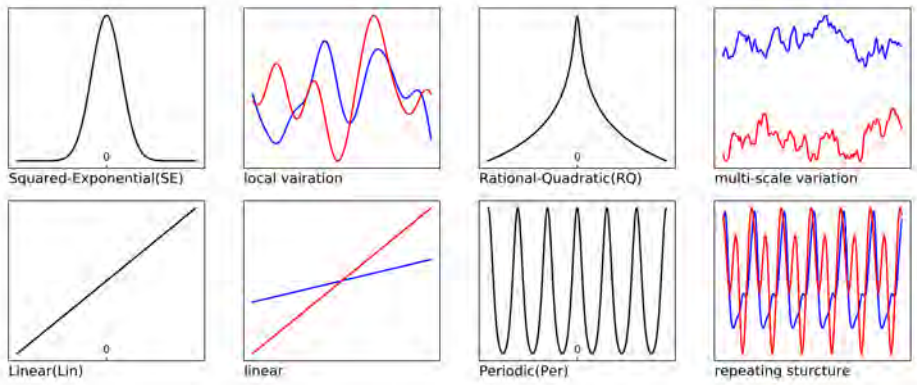


Fig. 2.1 First and Third columns: kernel $k(\cdot, 0)$. Second and Fourth columns: two examples draw from GP using corresponding kernels.

It is worth noticing that summing kernels can be seen as a superposition of independent GPs, such that suppose $f_1 \sim \mathcal{GP}(0, k_1)$ and $f2 \sim \mathcal{GP}(0, k_2)$ are two independent Gaussian Process, then $f_1 + f_2 \sim \mathcal{GP}(0, k_1 + k_2)$. With multi-dimensional input, sum of kernels discovers additive structures over dimensions while multiplication of kernels discovers interactive structures over dimensions. In views of AND-like and OR-like operations, summing kernels corresponds to OR-like operation since two locations are believed to have high covariance as

long as one of the kernel has a high value, while multiplying kernels corresponds to AND-like operation since similarity is assumed only when all kernels have high values (D. Duvenaud, Lloyd, et al., 2013).

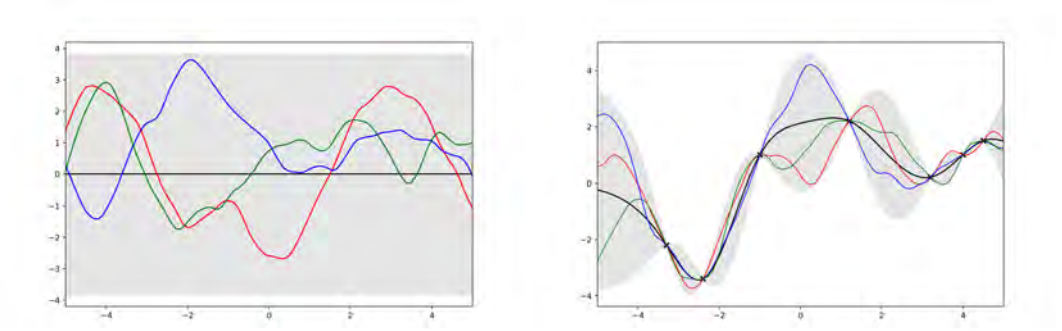
Incorporating knowledge into the prior after observing the training data is the primary interest of Bayesian models, and posterior of a GP model is another Gaussian Process. Suppose the N training observations $\mathbf{y}$ are noisy and modelled by Gaussian noise as follows, which can also be seen as a diagonal matrix added to the covariance,

$$p(\mathbf{y}|\mathbf{X},\sigma_y^2) = \prod_{n=1}^{N} \mathcal{N}(y_n; f(\mathbf{x}_n), \sigma_y^2)$$
$$cov(y_p, y_q) = k(\mathbf{x}_p, \mathbf{x}_q) + \sigma_y^2 \delta_{pq}, \text{ or, } cov(\mathbf{y}) = k(\mathbf{X},\mathbf{X}) + \sigma_y^2 \mathbf{I}$$

where $\delta_{pq} = 1$ iff $p = q$. The posterior distribution over the functions, $\mathbf{f}_*$, evaluated at some unseen N_* test points, $\mathbf{X}_*$, is equivalent to conditioning the joint Gaussian prior $p(\mathbf{y}, \mathbf{f}_*)$ on the observations. Graphically shown in Fig. 2.2, the generative process can be seen as drawing functions from the prior and rejecting those that disagree with the observations (Rasmussen, 2003). Analytical expressions for the posterior distribution and log-marginal likelihood for the varying hyperparameters are obtained as follows,

$$f|\mathbf{y} \sim \mathcal{GP}(\hat{m}(\mathbf{x}), \hat{k}(\mathbf{x},\mathbf{x}'))$$
$$\hat{m}(\mathbf{x}) = \mathbf{k}_{f\mathbf{f}}(\mathbf{K}_{\mathbf{ff}} + \sigma_y^2\mathbf{I})^{-1}\mathbf{y}$$
$$\hat{k}(\mathbf{x},\mathbf{x}') = \mathbf{k}_{ff'} - \mathbf{k}_{f\mathbf{f}}(\mathbf{K}_{\mathbf{ff}} + \sigma_y^2\mathbf{I})^{-1}\mathbf{k}_{\mathbf{f}f'}$$
$$\log p(\mathbf{y}|\theta,\sigma_y^2) = \log \mathcal{N}(\mathbf{y}; \mathbf{0}, \mathbf{K}_{\mathbf{ff}} + \sigma_y^2\mathbf{I}) = \frac{1}{2}\mathbf{y}^T(\mathbf{K}_{\mathbf{ff}} + \sigma_y^2\mathbf{I})^{-1}\mathbf{y} - \frac{1}{2}\log|\mathbf{K}_{\mathbf{ff}} + \sigma_y^2\mathbf{I}| - \frac{N}{2}\log 2\pi \tag{2.1}$$

where $\mathbf{k}_{f\mathbf{f}}$ and $\mathbf{K}_{\mathbf{ff}}$ are covariance vector and matrix between function values, θ are hyperparameters from mean and covariance functions specified by GP. The first term in the log-marginal likelihood is the only term which involves the observation $\mathbf{y}$ and hence encourages fitting and reduces bias. The second term penalises the complexity of the model, for example, small lengthscales would lead to large log-term value. As a result, selection of hyperparameters through maximisation of the marginal likelihood is robust to overfitting, despite the fact that the procedure can be trapped in local maximum (Rasmussen, 2003). Unfortunately, the cumbersome computational complexity resulting from the inversion of



(a) prior (b) posterior

Fig. 2.2 GP Bayesian Procedure: Shaded area in both figures are 95% confidence interval (i.e mean $\pm 2\times$ standard deviation) and black lines represent the mean; (a) shows three draws from the GP prior, (b) shows three draws from the posterior after observing seven noise-free points labeled as crosses

the matrix $\mathbf{K_{ff}} + \sigma_y^2\mathbf{I}$ in Eq. 2.1 which costs $\mathcal{O}(N^3)$ and the analytical intractability resulting from non-Gaussian likelihoods are the two main challenges of standard GPs. Many excellent approximation methods are developed to address these problems, one of which used variational inference (Titsias, 2009) and will be explained in detail in the following section.

2.2 Sparse GP approximation: VFE

Most approximate methods in the literature utilize $M < N$ inducing-points $\mathbf{u}$ and allow time complexity reduce from $\mathcal{O}(N^3)$ to $\mathcal{O}(NM^2)$. A variational method first introduced by Titsias (2009) that works on finite variable sets, and then discussed in infinite-dimensional function space by A. G. d. G. Matthews et al. (2016), maximizes a lower bound to the exact marginal likelihood (ELBO) of the model by applying the Jensen's inequality:

$$\log p(\mathbf{y}|\theta) = \log \int p(\mathbf{y}, f|\theta) df \geq \int q(f) \log \frac{p(\mathbf{y}, f|\theta)}{q(f)} df = \mathbb{E}_{q(f)} \log \left[\frac{p(\mathbf{y}, f|\theta)}{q(f)} \right] = \mathcal{L}_{ELBO}$$

where f is the function. The difference between the exact log-marginal likelihood and $\mathcal{L}_{ELBO}$ is just the Kullback-Leibler (KL) divergence between the variational distribution and the true posterior:

$$\log p(\mathbf{y}|\theta) - \mathcal{L}_{ELBO} = KL[q(f) \| p(f|\mathbf{y}, \theta)]$$

The exact marginal likelihood is recovered when $q(f) = p(f|\mathbf{y},\theta)$, and maximising the ELBO is equivalent to minimizing the KL divergence. This avoids overfitting and obtains an approximation by explicitly minimizing the distance between the variational one and the truth. Making the inducing-points $\mathbf{u}$ explicit, the variational distribution is chosen to be of the form:

$$q(f) = q(f_{\neq \mathbf{u}}, \mathbf{u}|\theta) = p(f_{\neq \mathbf{u}}|\mathbf{u},\theta)q(\mathbf{u})$$

The true posterior is $p(f|\mathbf{y},\theta) = p(f_{\neq \mathbf{u}}|\mathbf{y},\mathbf{u},\theta)p(\mathbf{u}|\mathbf{y},\theta)$. In this approximation, the inducing points $\mathbf{u}$ act like a sufficient statistics "summarizing" all training observations $\mathbf{y}$, or like a bottleneck such that $y_{\neq \mathbf{u}}$ communicate through the inducing points indirectly with the data (Hensman, A. Matthews, et al., 2015). This particular form allows a cancellation of $p(f_{\neq \mathbf{u}}|\mathbf{u},\theta)$ which is the last remaining source of cubic time complexity.

$$\begin{aligned}
\mathcal{L}_{ELBO} &= \mathbb{E}_{q(f|\theta)}\left[\log \frac{p(\mathbf{y}|f,\theta)p(f_{\neq \mathbf{u}}|\mathbf{u},\theta)p(\mathbf{u}|\theta)}{p(f_{\neq \mathbf{u}}|\mathbf{u},\theta)q(\mathbf{u})}\right] \\
&= \mathbb{E}_{q(f|\theta)}\left[\log \frac{p(\mathbf{y}|f,\theta)p(\mathbf{u}|\theta)}{q(\mathbf{u})}\right] \\
&= \sum_{n=1}^{N} \mathbb{E}_{q(f|\theta)}[\log p(y_n|f_n,\theta)] - KL[q(\mathbf{u})\|p(\mathbf{u})] \qquad (2.2)
\end{aligned}$$

where $f_n = f(\mathbf{x}_n)$. When the variational distribution is of the form $q(\mathbf{u}) = \mathcal{N}(\mathbf{u},\mathbf{m},\mathbf{S})$, analytical solution that maximizes the ELBO with respect to parameter $\mathbf{m}$ and $\mathbf{S}$ can be found for regression with Gaussian observation noise (Titsias, 2009). However, to solve the intractability for non-Gaussian observations and enable stochastic optimisation via Monte-Carlo sampling method, the uncollapsed bound is used where an approximate posterior over the function evaluated at training points, $\mathbf{f}$, is obtained by analytically marginalising out the inducing points, $\mathbf{u}$ as follows,

$$\begin{aligned}
q(f) = \int p(f_{\neq \mathbf{u}}|\mathbf{u})q(\mathbf{u})d\mathbf{u} &\sim \mathcal{GP}(f;\tilde{m}(\mathbf{x}),\tilde{k}(\mathbf{x},\mathbf{x}')) \\
\tilde{m}(\mathbf{x}) &= \mathbf{k}_{f\mathbf{u}}\mathbf{K}_{\mathbf{uu}}^{-1}\mathbf{m} \\
\tilde{k}(\mathbf{x},\mathbf{x}') &= k_{ff'} - \mathbf{k}_{f\mathbf{u}}\mathbf{K}_{\mathbf{uu}}^{-1}(\mathbf{K}_{\mathbf{uu}} - \mathbf{S})\mathbf{K}_{\mathbf{uu}}^{-1}\mathbf{k}_{\mathbf{u}f'}
\end{aligned}$$

When non-Gaussian likelihood is placed, the expectation integrals in Eq. 2.2 becomes intractable and can be evaluated using quadrature (Hensman, A. Matthews, et al., 2015).

Evaluating the ELBO and $q(\mathbf{u})$ has time complexity $\mathcal{O}(NM^2)$ which results in a significant computational saving.

2.3 GPAR and GPLAR

As all previous sections are dealing with scalar output, we now consider the modelling of multiple outputs. In the multi-output scenario, assume $\mathbf{x}$ and $\mathbf{y} = \{\mathbf{y}_l\}_{l=1}^{M}$ are the training inputs and associated observations for M outputs. We assume all M outputs share the same input space, although the training sets can be heterotopic or isotopic in different situations, i.e. each output can have different or same training sets such that evaluations are obtained by separate or simultaneous simulations. We will use isotopic configurations from now on, but the discussed models can also be generalized to heterotopic situations which will be discussed in section. 2.5. Utilizing the product rule to decompose the joint distribution of multi-dimension into a set of univariate conditional distributions, Gaussian Process Autoregressive Regression (GPAR) (Requeima et al., 2018) model factorizes the distribution of M outputs $y_{1:M}(\mathbf{x}) = (y_1(\mathbf{x}), \ldots, y_M(\mathbf{x}))$ as,

$$p(y_{1:M}(\mathbf{x})) = p(y_1(\mathbf{x}))p(y_2(\mathbf{x})|y_1(\mathbf{x}))\ldots p(y_M(\mathbf{x})|y_{1:M-1}(\mathbf{x}))$$

such that $y_m(\mathbf{x})$ is generated from $y_{1:m-1}(\mathbf{x})$ according to some latent function f_m. GPAR models these latent function $f_{1:M}$ with GPs as follows,

$$\begin{aligned}
y_1(\mathbf{x}) &= f_1(\mathbf{x}), & f_1 &\sim \mathcal{GP}(0, k_1(\mathbf{x}, \mathbf{x}')) \\
y_2(\mathbf{x}) &= f_2(y_1(\mathbf{x}, \mathbf{x})), & f_2 &\sim \mathcal{GP}(0, k_2\left((y_1(\mathbf{x}), \mathbf{x}), (y_1(\mathbf{x}'), \mathbf{x}')\right) \\
&\vdots & & \\
y_M(\mathbf{x}) &= f_M(y_{1:M-1}(\mathbf{x}, \mathbf{x})), & f_M &\sim \mathcal{GP}(0, k_M\left((y_{1:M-1}(\mathbf{x}), \mathbf{x}), (y_{1:M-1}(\mathbf{x}'), \mathbf{x}')\right)
\end{aligned}$$

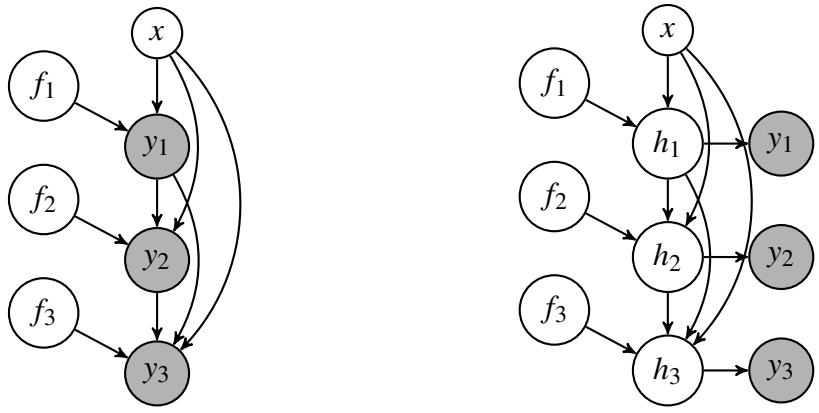
Kernel Selections. The choices of kernels $\{k_{1:M}\}$ are crucial since they determine nonlinear or linear dependencies to be modeled between outputs, input-dependent or input-independent relationships between noises that can be discovered. The original paper adopted the approach presented in Table 2.1, which will be the simple starting point of the upcoming variants of GPAR.

Deficiencies of GPAR. There are limitations in the current formulation of GPAR. An example graphical model of three-dimensional outputs are shown in Fig. 2.3a, where observation

Dependencies	Kernels
Linear	$k_{NL}(\mathbf{x},\mathbf{x}') + k_{Lin}(y(\mathbf{x}), y(\mathbf{x}'))$
+ dep. on input	$k_{NL}(\mathbf{x},\mathbf{x}') + k_{NL}(\mathbf{x},\mathbf{x}')k_{Lin}(y(\mathbf{x}), y(\mathbf{x}'))$
Nonlinear	$k_{NL}(\mathbf{x},\mathbf{x}') + k_{NL}(y(\mathbf{x}), y(\mathbf{x}'))$
+ dep. on input	$k_{NL}(\mathbf{x},\mathbf{x}') + k_{NL}((\mathbf{x}, y(\mathbf{x})), (\mathbf{x}', y(\mathbf{x}')))$
Linear + Nonlinear	$k_{NL}(\mathbf{x},\mathbf{x}') + k_{NL}(y(\mathbf{x}), y(\mathbf{x}')) + k_{Lin}(y(\mathbf{x}), y(\mathbf{x}'))$

Table 2.1 GPAR kernel $k_{1:M}$ for $f_{1:M}$, where k_{NL} denotes nonlinear kernels such as squared exponential or rational quadratic kernels, k_{Lin} denotes a linear kernel. Here, $y(\mathbf{x})$ are all preceding outputs and can be multi-dimensional.

y_1 is directly used as inputs to function f_2 and f_3 and the possible noises are not modelled. Noisy outputs from an earlier stage result in noisy inputs to a subsequent level. The original paper solved this by employing a denoising transformation, such that the posterior predictive mean of preceding outputs are used as inputs instead. Furthermore, when there are missing values in some levels of outputs, GPAR might fail to produce correct predictive distribution under some situation due to how the missing values are imputed. Although the inference and learning procedure remains valid for *closed-downwards* observations, i.e., for every observation $y_{mn} = y_m(\mathbf{x}_n)$, there are also observations $y_{(1:m-1)n}$, since the posterior and the evidence decompose like the prior as a product of conditionals, for not closed-downwards observations, imputation is required and the model uses posterior predictive mean. Experiments have shown that this imputation method and GPAR's layer-by-layer fitting procedure have poor performance on closed-upwards observations.



(a) GPAR (b) GPLAR

Fig. 2.3 : Graphical models of (a) GPAR and (b) GPLAR. Observed variables $y_{1:3}$ are shaded. $f_{1:3}$ denote latent function mappings.

A more principled approach is to do fully Bayesian inference. Instead of directly working on observations, we introduce latent variables, $\mathbf{h_{1:3}}$ for each output (graphically shown in Fig. 2.3b) and an approximation method is required to inference these variables. This method also enables non-Gaussian likelihoods, such as classifications, or non-negative data. We call this model Gaussian Process Latent Autoregressive (GPLAR) model. To leverage the fact that a similar idea is implemented in the inference schemes for deep GPs (Salimbeni et al., 2017), we review the basic ideas in the next section.

2.4 Approximate Inference for DGP and GPLAR

Deep Gaussian Process (DGPs) is a multi-layer generalisation of GPs combined in a hierarchical composition (Damianou et al., 2013). When GPAR's kernels $k_{1:M}$ depend non-linearly on previous outputs, one can construct a particular form of DGP (Requeima et al., 2018). For simplicity and easy comparison with GPLAR, we assume scalar real-valued output, single-dimensional intermediate layers and observations with Gaussian noise. The complete probabilistic representation of such a DGP comprising L layers can be written as follows,

$$
\begin{aligned}
p(f_l|\theta_l) &= \mathcal{GP}(f_l; \mathbf{m}_l, \mathbf{K}_l), l = 1, \ldots, L \\
p(\mathbf{h}_l|f_l, \mathbf{h}_{l-1}, \sigma_l^2) &= \prod_n \mathcal{N}(h_{l,n}; f_l(h_{l-1,n}), \sigma_l^2), h_{0,n} = \mathbf{x}_n \\
p(\mathbf{y}|f_L, \mathbf{h}_{L-1}, \sigma_L^2) &= \prod_n \mathcal{N}(y_n; f_L(h_{L-1,n}), \sigma_L^2)
\end{aligned}
$$

Similar to what is introduced to GPLAR, the inputs to each layer are noisy outputs from the previous layer, $\mathbf{h}_{l-1}$, which are referred to as "hidden variables". The probabilistic model of a GPLAR model with L-dimensional outputs can be represented as below,

$$
\begin{aligned}
p(f_l|\theta_l) &= \mathcal{GP}(f_l; \mathbf{m}_l, \mathbf{K}_l), l = 1, \ldots, L \\
p(\mathbf{h}_l|f_l, \mathbf{X}, \mathbf{h}_{1:l-1}, \sigma^2) &= \prod_n \mathcal{N}(h_{l,n}; f_l(\mathbf{x}_n, h_{1:l-1,n}), \sigma_l^2) \\
p(\mathbf{y}_l|\mathbf{h}_l) &= \prod_n \mathcal{N}(y_{l,n}; h_{l,n}, \sigma_{y_l}^2)
\end{aligned}
$$

Success of DGPs lies in the intermediate layers, which act like input, output wrappings, extracting important features. This leads to automatic learning of complex kernels which is more flexible compared to apprehend hand-chosen kernels. Hence, multi-dimensional

intermediate layers are important and crucial to DGPs. While in GPLAR, we use single-dimensional latent variables which might cause problems. However, since skip connections exit both from inputs and from previous outputs in GPLAR, pathology of oversimple latent representation can be alleviated (D. Duvenaud, Rippel, et al., 2014).

Posterior distributions over the latent function mappings, $f_{1:L}$, as well as over the intermediate hidden variables $\mathbf{h}_{1:L-1}$ are of interest. Taking a 3-layers DGP as an example, the joint density of all variables and the exact posterior of unobserved variables ($f_{1:3}$ and $\mathbf{h}_{1:2}$) have the following form,

$$p(\mathbf{y}, f_1, f_2, f_3, \mathbf{h}_1, \mathbf{h}_2) = p(f_1)p(f_2)p(f_3)\prod_{n=1}^{N}[p(h_{1n}|f_1, \mathbf{x}_n)p(h_{2n}|f_2, h_{1n})p(h_{3n}|f_3, h_{2n})]$$

$$p(f_1, f_2, f_3, \mathbf{h}_1, \mathbf{h}_2|\mathbf{y}) = \frac{p(\mathbf{y}, f_1, f_2, f_3, \mathbf{h}_1, \mathbf{h}_2)}{p(\mathbf{y})},$$

$$\text{where } p(\mathbf{y}) = \int p(\mathbf{y}, f_1, f_2, f_3, \mathbf{h}_1, \mathbf{h}_2) df_1 df_2 df_3 d\mathbf{h}_1 d\mathbf{h}_2$$

Here, $p(\mathbf{y})$ is the exact marginal likelihood for hyperparameter tuning. Unlike in the standard GPs, the inputs to each layer are stochastic or random caused by previous layers and the mapping is no longer Gaussian. Whilst this enables complex wrapping and build sophisticated relationships, the above posteriors and model evidence also become analytically intractable. Even though the computational complexity can be reduced by sparse approximation, the intractability due to non-Gaussian functionality is still not solved. Hence, more careful approximations are required. The sparse approximation method used in Damianou and Lawrence's original work (Damianou et al., 2013) introduce variational distributions over both the latent functions, $\{f_l\}$, and the hidden variables, $\{h_{ln}\}$. This was memory intensive as the space complexity scales linearly with the number of data points and rendering the model obsolete for large datasets (Thang Duc Bui, 2018). Furthermore, initialization of the variational parameters would be troublesome if the inducing locations and the latent GPs are mismatched (Richard et al., 2011). Instead, existing works (Salimbeni et al., 2017; T. Bui et al., 2016) used approximation schemes that only require variational distribution over latent functions. As noted by Thang Duc Bui (2018), since more importance is devoted to accurate predictions at test time, approximate posterior over non-linear GP mappings requires more effort instead of good approximations over training hidden variables. Hence, in these works, the conditional $p(h_{ln}|f_l, h_{(l-1)n})$ is retained, i.e. no approximation over the hidden variables are learnt, which also allows explicit dependencies within $\{\mathbf{h}_l\}$, and between $\{\mathbf{h}_l\}$ and $\{f_l\}$.

The resulting approximate posterior by introducing inducing points $\mathbf{u}_l$ to each layer are as follows,

$$\begin{aligned} q(f_1, f_2, f_3, \mathbf{h}_1, \mathbf{h}_2) =& p(f_{1\neq\mathbf{u}_1}|\mathbf{u}_1)p(f_{2\neq\mathbf{u}_2}|\mathbf{u}_2)p(f_{3\neq\mathbf{u}_3}|\mathbf{u}_3)q(\mathbf{u}_1)q(\mathbf{u}_2)q(\mathbf{u}_3)\times \\ & \prod_n p(h_{1n}|f_1, \mathbf{x}_n)p(h_{2n}|f_2, h_{1n}) \end{aligned}$$

Using the same idea, we introduce inducing points to every layer (w.r.t dimensions) of GPLAR, we have the approximate posterior and joint distribution (written with $\mathbf{u}$ explicitly) of the model as,

$$\begin{aligned} q(f_1, f_2, f_3, \mathbf{h}_1, \mathbf{h}_2, \mathbf{h}_3) =& p(f_{1\neq\mathbf{u}_1}|\mathbf{u}_1)p(f_{2\neq\mathbf{u}_2}|\mathbf{u}_2)p(f_{3\neq\mathbf{u}_3}|\mathbf{u}_3)q(\mathbf{u}_1)q(\mathbf{u}_2)q(\mathbf{u}_3)\times \\ & \prod_n \left[p(h_{1n}|f_1, \mathbf{x}_n)p(h_{2n}|f_2, h_{1n}, \mathbf{x}_n)p(h_{3n}|f_3, h_{2n}, h_{1n}, \mathbf{x}_n) \right] \\ p(\mathbf{y}, f_1, f_2, f_3, \mathbf{h}_1, \mathbf{h}_2, \mathbf{h}_3) =& p(f_{1\neq\mathbf{u}_1}|\mathbf{u}_1)p(f_{2\neq\mathbf{u}_2}|\mathbf{u}_2)p(f_{3\neq\mathbf{u}_3}|\mathbf{u}_3)p(\mathbf{u}_1)p(\mathbf{u}_2)p(\mathbf{u}_3)\times \\ & \prod_n \Big[p(h_{1n}|f_1, \mathbf{x}_n)p(h_{2n}|f_2, h_{1n}, \mathbf{x}_n)p(h_{3n}|f_3, h_{2n}, h_{1n}, \mathbf{x}_n) \\ & p(y_{1n}|h_{1n})p(y_{2n}|h_{2n})p(y_{3n}|h_{3n}) \Big] \end{aligned}$$

Evidence Lower Bound. As the procedure stated in section 2.2, the lower bound to the log-marginal likelihood is as follows,

$$\mathcal{L}_{ELBO} = \mathbb{E}_q \left[\log \frac{p(\mathbf{y}, f_1, f_2, f_3, \mathbf{h}_1, \mathbf{h}_2, \mathbf{h}_3)}{q(f_1, f_2, f_3, \mathbf{h}_1, \mathbf{h}_2, \mathbf{h}_3)} \right] \tag{2.3}$$

$$= -\sum_{l=1}^{3} KL[q(\mathbf{u}_l)\|p(\mathbf{u}_l)] + \sum_{l,n} \mathbb{E}_q [\log p(y_{ln}|h_{ln})] \tag{2.4}$$

The difference between the exact log-marginal likelihood and ELBO is the KL divergence between the approximate posterior and the true one:

$$\log p(\mathbf{y}) - \mathcal{L}_{ELBO} = KL[q(f_1, f_2, f_3, \mathbf{h}_1, \mathbf{h}_2, \mathbf{h}_3)\|p(f_1, f_2, f_3, \mathbf{h}_1, \mathbf{h}_2, \mathbf{h}_3|\mathbf{y})]$$

Maximizing the ELBO w.r.t to the variational parameters and the hyperparameters is equivalent to minimizing the KL-divergence. We jointly obtain approximations to the model evidence and the posterior. The latent function distribution conditioning on inducing points, $p(f_{l\neq\mathbf{u}_l}|\mathbf{u}_l)$ and the hidden variables conditionals, $p(h_{ln}|f_l, h_{(1:l-1)n}, \mathbf{x}_n)$, are cancelled out in the variational expectation, which leads to greatly simplified form in Eq. 2.4. The second term in Eq. 2.4 decomposes along the training instance and along output dimensions (i.e. along inputs and outputs which enables use of stochastic optimization method), such that one layer output only requires one hidden variables, which can be rewritten as,

$$
\begin{aligned}
\sum_{l,n} \mathbb{E}_q \left[\log p(y_{ln}|h_{ln})\right] = &\sum_n \int_{h_{1n}} q(h_{1n}|\mathbf{x}_n) \log p(y_{1n}|h_{1n}) \\
&+ \int_{h_{1n},h_{2n}} q(h_{1n}|\mathbf{x}_n) q(h_{2n}|h_{1n},\mathbf{x}_n) \log p(y_{2n}|h_{2n}) \\
&+ \int_{h_{1n},h_{2n},h_{3n}} q(h_{1n}|\mathbf{x}_n) q(h_{2n}|h_{1n},\mathbf{x}_n) q(h_{3n}|h_{2n},h_{1n},\mathbf{x}_n) \log p(y_{3n}|h_{3n}) \\
= &\sum_{l,n} \int_{h_{ln}} q(h_{ln}) \log p(y_{ln}|h_{ln}) \\
\text{where } q(h_{ln}) = &\int_{h_{(1:l-1)n}} q(h_{ln}|h_{(1:l-1)n},\mathbf{x}_n) \ldots q(h_{2n}|h_{1n},\mathbf{x}_n) q(h_{1n}|\mathbf{x}_n)
\end{aligned}
$$

Positing a Gaussian form for each variational distribution $q(\mathbf{u}_l) = \mathcal{N}(\mathbf{u}_l; \mathbf{m}_l, \mathbf{S}_l)$, we notice that the latent function, f_l, can be analytically marginalized out at each layer, as mentioned in section 2.2,

$$
\begin{aligned}
q(h_{ln}|\mathbf{x}_n, h_{(1:l-1)n}) &= \int_{f_l} p(h_{ln}|f_l, \mathbf{x}_n, h_{(1:l-1)n}) p(f_{l\neq\mathbf{u}_l}|\mathbf{u}_l) q(\mathbf{u}_l) \\
&= \mathcal{N}(h_{ln};\ \mu_{h_l|h_{1:l-1}}(\hat{\mathbf{x}}_{ln}),\ \sigma^2_{h_l|h_{1:l-1}}(\hat{\mathbf{x}}_{ln})) \\
\text{where } \mu_{h_l|h_{1:l-1}}(\hat{\mathbf{x}}_{ln}) &= \mathbf{k}_l(\hat{\mathbf{x}}_{ln}, \mathbf{Z}_l) \mathbf{K}^{-1}_{\mathbf{u}_l\mathbf{u}_l} \mathbf{m}_l, \\
\sigma^2_{h_l|h_{1:l-1}}(\hat{\mathbf{x}}_{ln}) &= k_l(\hat{\mathbf{x}}_{ln}, \hat{\mathbf{x}}_{ln}) - \mathbf{k}_l(\hat{\mathbf{x}}_{ln}, \mathbf{Z}_l) \mathbf{K}^{-1}_{\mathbf{u}_l\mathbf{u}_l} (\mathbf{K}_{\mathbf{u}_l\mathbf{u}_l} - \mathbf{S}_l) \mathbf{K}^{-1}_{\mathbf{u}_l\mathbf{u}_l} \mathbf{k}_l(\mathbf{Z}_l, \hat{\mathbf{x}}_{ln}) + \sigma_l^2
\end{aligned}
$$

where $\hat{\mathbf{x}}_{ln} = (\mathbf{x}_n, h_{(1:l-1)n})$ is concatenation of input and previous hidden variables, and $\mathbf{K}_{\mathbf{u}_l\mathbf{u}_l} = \mathbf{K}_l(\mathbf{Z}_l, \mathbf{Z}_l)$ is the covariance between each layer's inducing points. Notice that for $q(h_{1n})$, the distribution is not conditioned on any hidden variables but only input $\mathbf{x}_n$, and hence is a Gaussian predictive distribution. However, for $q(h_{2n}|\mathbf{x}_n) = \int_{h_{1n}} q(h_{2n}|h_{1n},\mathbf{x}_n) q(h_{1n}|\mathbf{x}_n)$, since conditional GP predictive distribution is non-linear w.r.t h_{1n}, and there exists random-

ness in h_{1n}, the resulting $q(h_{2n})$ can be seen as a complicated mixture of infinite number of Gaussian densities (Thang Duc Bui, 2018), which is likely to be multi-modal or heavy-tailed.

Simple Monte Carlo Sampling. The nested simple Monte Carlo method is adopted by Salimbeni et al. (2017) to propagate a Gaussian distribution through a GP posterior, where $q(h_{2n}|\mathbf{x}_n)$ is approximated by a mixture of finite number of Gaussian densities sampled from $q(h_{1n}|\mathbf{x}_n)$ as follows,

$$q(h_{2n}|\mathbf{x}_n) \approx \frac{1}{R}\sum_r q(h_{2n}|h_{1nr},\mathbf{x}_n),\ \text{s.t. } h_{1nr} \sim q(h_{1n}|\mathbf{x}_n)$$

This sampling-based approximation is unbiased and exact when $R \to \infty$. When further propagating h_{2n} and h_{1n} to posterior of h_{3n}, samples are now drawn from a uniformly weighted mixture of Gaussian and we have,

$$\begin{aligned}
q(h_{3n}|\mathbf{x}_n) &= \int_{h_{1n},h_{2n}} q(h_{3n}|h_{2n},h_{1n},\mathbf{x}_n)q(h_{2n}|h_{1n},\mathbf{x}_n)q(h_{1n}|\mathbf{x}_n) \\
&\approx \frac{1}{R}\sum_r \int_{h_{2n}} q(h_{3n}|h_{2n},h_{1nr},\mathbf{x}_n)q(h_{2n}|h_{1nr},\mathbf{x}_n), && h_{1nr} \sim q(h_{1n}|\mathbf{x}_n) \\
&\approx \frac{1}{R}\sum_r \frac{1}{M}\sum_m q(h_{3n}|h_{2nm},h_{1nr},\mathbf{x}_n), && h_{2nm} \sim q(h_{2n}|h_{1nr},\mathbf{x}_n) \\
&\approx \frac{1}{R}\sum_r q(h_{3n}|h_{2nr},h_{1nr},\mathbf{x}_n), && h_{1nr} \sim q(h_{1n}|\mathbf{x}_n) \\
& && h_{2nr} \sim q(h_{2n}|h_{1nr},\mathbf{x}_n)
\end{aligned}$$

In DGPs, the marginals in each layer only depend on output from the last layer, for example, h_{Ln} only depends on $h_{(L-1)n}$ which in turn only depends on $h_{(L-2)n}$. Therefore, every h_{ln} is propagated only to the next layer. While in GPLAR, the marginals in each layer depend on outputs from all previous layers, hence h_{ln} is propagated to all following layers from $(l+1)$-th to the last. To obtain a low variance gradient of the variational expectation w.r.t the variational parameters, which in here corresponds to $\mathbf{m}_l$ and $\mathbf{S}_l$, we apply the reparametrisation trick invented by Kingma and Welling (2013), and recursively draw samples $h_{lnr} \sim q(h_{ln}|h_{(1:l-1)n},\mathbf{x}_n)$ as,

$$h_{lnr} = \mu_{h_l|h_{1:l-1}}(\hat{\mathbf{x}}_{ln}) + \varepsilon_{lnr} \times \sigma^2_{h_l|h_{1:l-1}}(\hat{\mathbf{x}}_{ln}),\ \varepsilon_{lnr} \sim \mathcal{N}(0,1) \tag{2.5}$$

When general likelihood rather than Gaussian observation is placed, $\log p(y_{ln}|h_{ln})$ requires additional approximations such as another simple Monte Carlo sampling, which also enables

sub-sampling the data and achieve scalability.

Predictions. To make predictions over test locations $\mathbf{X}_*$, we can obtain an approximate predictive distribution as a mixture of Gaussian densities by propagating S samples through the variational posterior as,

$$q(h^*_{ln}|\mathbf{x}^*_n) \approx \frac{1}{R}\sum_r q(h^*_{ln}|h^*_{(1:l-1)nr}, \mathbf{x}^*_n)$$

where samples are drawn using Eq. 2.5, replacing input $\mathbf{x}$ by test input $\mathbf{x}^*$.

2.5 Related Work

Many existing works have been done to explore dependencies between outputs, similarly to GPAR or GPLAR that utilize Gaussian Process to model non-linearities in data, i.e. MOGPs. The difference lies in how the latent process relates to each other, and how information, containing inputs or hidden variables, flow through latent process, either shared in an implicit mixture form, or explicitly Markov-chained or fully-connected, which will be thoroughly discussed in this section. A unifying framework presenting connections between different methods of MOGPs are shown in Figure. 2.4.

2.5.1 Linear and Nonlinear Variants.

When kernels of GPLAR depend linearly on previous outputs, a multi-output GP model where latent processes are combined linearly using a lower-triangular matrix is recovered (Requeima et al., 2018). Defining a suitable cross-covariance function between multiple outputs has been the main focus of MOGPs. One classical way to define such a property shared between tasks are the linear coregionalization model (LCM) (Wackernagel, 2013), which is equivalent to putting a GP prior on the multiple tasks that share a set of independent latent functions linearly combined by a matrix as follows,

$$f_l(\mathbf{x}) = \sum_{j=1}^{Q} a_{lj} u_j(\mathbf{x}), \text{a matrix formulation:} \mathbf{f} = \mathbf{A}\mathbf{u}$$

where $\mathbf{f} = \{f_l\}_{l=1}^{M}$ denotes functions modeling each task, $\mathbf{A} = \{a_{lj}\}_{l=1,j=1}^{M,Q}$ is the matrix discovering inter-task correlations and $\mathbf{u} = \{u_j \sim \mathcal{GP}(0, k_j)\}$ is the set of independent GPs.

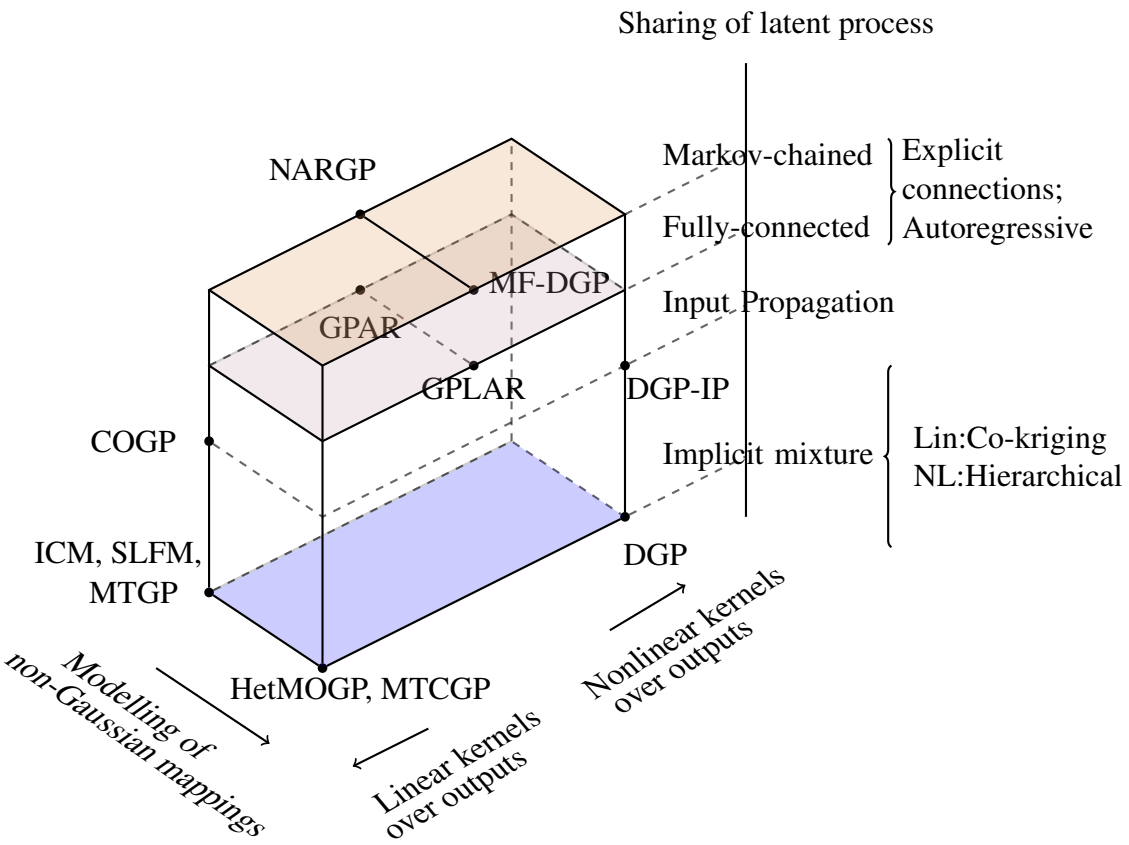


Fig. 2.4 A unifying framework of multi-output GPs, corresponding models and references of every abbreviation are presented in Table. 2.2.

Abbreviation	Model	Reference
ICM	Intrinstic Coregionalisation Model	Wackernagel (2013)
SLFM	Semi-Parametric Latent Factor Model	Gryparis et al. (2007)
MTGP	Multi-Task Gaussian Processes	Yu et al. (2005),Bonilla et al. (2008)
COGP	Collaborative Multi-Output GPs	Nguyen et al. (2014)
HetMOGP	Heterogeneous Multi-Output GPs	Moreno-Muñoz et al. (2018)
MTCGP	Multitask classification GPs	Skolidis et al. (2011)
DGP	Deep Gaussian Process	Damianou et al. (2013),T. Bui et al. (2016)
	with input propagation	D. Duvenaud, Rippel, et al. (2014)
NARGP	Nonlinear Multi-fidelity Model	Perdikaris et al. (2017)
MF-DGP	Multi-fidelity DGP	Cutajar et al. (2019)

Table 2.2 Models and references

Within the MOGPs literature, many existing works are variants of LCM which differ on the extent of how the latent process $\mathbf{u}$ are shared. Works of Yu et al. (2005) achieve inter-task tying by drawing functions from one same process and never uses the mixing matrix. Semi-parametric latent factor model (Gryparis et al., 2007) uses $P < M$ (M= number of outputs) latent processes with P distinct covariance functions while Bonilla et al. (2008) uses P latent process with one same covariance function. After introducing pseudo-points for sparse approximations to enable scalable inference, collaborative multi-output GP model (COGP) (Nguyen et al., 2014) achieves sharing of latent process by sharing "sparsity strucuture", i.e. the inducing variables. Works of Moreno-Muñoz et al. (2018) further generalize by sharing the same set of inducing inputs, $\mathbf{Z}$. Although these methods are involving and becoming more and more flexible to inference dependencies across outputs, the kernels over outputs are still linear, while more complex relationships can be discovered if nonlinear kernels over outputs are used. One advantage of LCM is that any ordering of outputs combination is achievable since the mixing matrix, $\mathbf{A}$ is not constrained, while one particular ordering of outputs modelled by GPAR or GPLAR would restrict the mixing matrix to be lower-triangular.

When kernels of GPLAR depend non-linearly on previous outputs, a particular structured DGP with input propagation is recovered (Requeima et al., 2018). Since the single-layer GPs, which is just independent GPs (sharing covariance functions and inducing inputs in settings of (Salimbeni et al., 2017), is limited by assumptions of Gaussian marginals, and the requirement of defining suitable priors in terms of mean and covariance functions, deep structures solve these using a hierarchical combination of latent processes. However, since this input and output wrappings are complex and implicit, either inference or interpretation over missing values after observing other outputs would be difficult. The leveraged version where each intermediate layer is directly observed as corresponding output would be much more helpful and will be discussed in the next subsection.

There are also works that consider more direct connections between outputs or inputs and gain greater flexibility. Apart from a weighted sum of shared latent function $\mathbf{u}$, COGP added an individual GP h_l which is unique to each output, y_l. The observation likelihood model is given by,

$$p(\mathbf{y}|u_{1:Q}, h_{1:M}) = \prod_{l=1}^{M}\prod_{n=1}^{N} \mathcal{N}(y_{ln}; \sum_{j=1}^{Q} a_{lj}u_j(\mathbf{x}_n) + h_l(\mathbf{x}_n), \sigma_{y_l}^2)$$

where a_{ij} is the combination weights, i.e. the mixing matrix. These unique functions over inputs can be seen as input propagation in the linear case. In the nonlinear situation, problem of invariance of all modeled directions but one when repeated composition of GPs are used, is alleviated by propagating the input to each layer (D. Duvenaud, Rippel, et al., 2014). Apart from input propagation, the multi-view DGP model (MvDGP) presented by Zhu et al. (2020) considered modeling data coming from different sources that has different structured features that need to be treated separately. Specifically, if the data are from two different sources, two DGPs are run over the two input spaces respectively. After certain depths, the last layers of two views are concatenated and passed to a common DGP which models shared information from both views. The concatenation step is similar to the concatenation of inputs and previous outputs in GPLAR, where GPs at layer l can be seen as a multi-view problem with views from both the input and previous outputs.

2.5.2 Explicit sharing of latent process: Autoregressive

As mentioned above, implicit mixtures of latent processes, either in an LCM form that leads to linear kernels over outputs or in a hierarchical way that leads to nonlinear kernels over outputs, is limiting and difficult to interpret. Explicit connections with other outputs using kernels in "free-form" as shown in Table. 2.1 would provide more complicated modelling of dependencies and benefit dealing with arbitrarily missing data. Multi-fidelity model is closely related to multi-output predictions and is interested in fusions of cheaply-obtained information with low-fidelity into limited high-fidelity data where relationships between observations with variant fidelity levels are discovered. The nonlinear multi-fidelity model (NARGP) raised by Perdikaris et al. (2017) leveraged the structure in deep GPs and made the connections between outputs explicit by passing the outputs from previous fidelity layer to next layer in nested GPs, just like GPAR. However, unlike the fully autoregressive connections in GPAR and GPLAR, Perdikaris et al. (2017) assumed a Markov property in multi-fidelity situations such that, given the evaluations at nearest fidelity functions, nothing more can be learnt from earlier fidelities as follows,

$$f_t(\mathbf{x}) = g_t(\mathbf{x}, f_{t-1}(\mathbf{x}))$$

where g_t is a function with a GP prior, f_t and f_{t-1} are functions modelling data at fidelity level t and $t-1$. The original design of kernels in the NARGP model also captures both nonlinear mappings between fidelities and their correlation with input. However, Perdikaris et al. (2017) assumed nonlinear kernels for every component which is not appropriate when

the correlations between fidelities are linear. Hence, Cutajar et al. (2019) who studied a model based on NARGP, use the following covariance function which is nearly identical to our discussions (f^*_{l-1} denotes the function of previous fidelity at previous layer)

$$k_l = k_l^{\rho}(\mathbf{x},\mathbf{x}')[\sigma_l^2 f^*_{l-1}(\mathbf{x})^T f^*_{l-1}(\mathbf{x}') + k_l^{f-1}(f^*_{l-1}(\mathbf{x}), f^*_{l-1}(\mathbf{x}')] + k_l^{\delta}(\mathbf{x},\mathbf{x}')$$

where k_l^{ρ} is a space-dependent scaling factor which corresponds to the kernel in GPLAR who captures outputs' dependence on input, σ_l^2 is the variance of linear kernel over fidelities, k_l^{f-1} take charges of non-linear correlations over fidelities, and k_l^{δ} models bias at that level.

From another perspective, as mentioned in (Liu et al., 2018), a significant difference between the linear and nonlinear variants mentioned in last subsection and the autoregressive models is that the outputs are treated with unequal importance. This asymmetric characteristic is also revealed as a special form of the mixing matrix in the LMC framework where it is constrained to a lower triangular matrix. In the multi-fidelity scenario, the datasets usually has a natural ordering since the goal is to improve the predictions of expensive outputs with high-fidelity by transferring knowledge learned from inexpensive outputs with low-fidelity. Similarly, in GPAR and GPLAR, the data also admits an ordering. For example, if inferring missing values in one single output is of interest, this output should be placed last so that dependencies with all previous channels are modeled and transfer learning from all previous tasks are achieved. This may exhibit some disadvantages since GPLAR becomes sensitive to the selection of one particular ordering. Alternatives of GPLAR that deal with this problem are discussed in Chapter 3. There are models which also take outputs from previous tasks as inputs but treat all outputs equally, such as the stacked single target model (Wolpert, 1992) and ensemble of regressor chains (Read et al., 2011). The stacked single-target model raised by Wolpert (1992) predicts each output using independent GPs at the first stage and then augment the input with predictions from the first stage to learn a new function for each output at the second stage. However, such transformation of the multi-output modeling process into a series of successive modeling of single-output problems (Borchani et al., 2015) can be problematic if outputs observed latter fail to feedback its effect on modelings of previous tasks. This problem is also present in GPAR but solved in GPLAR as discussed in the following subsection.

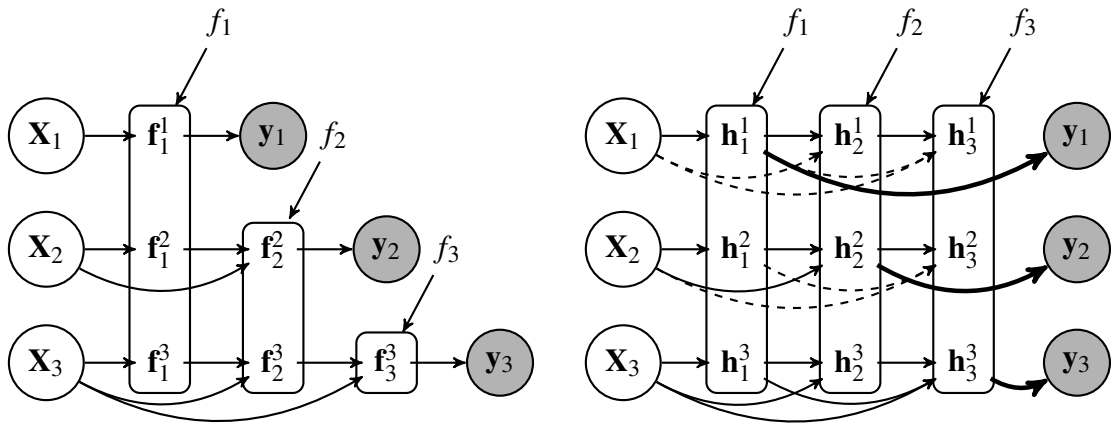
2.5.3 Modelling for non-Gaussian mappings

When non-Gaussian likelihoods are placed, computing posterior distribution becomes intractable and different schemes are used to offer approximations. The heterogeneous multi-output GP(HetMOGP) (Moreno-Muñoz et al., 2018) draws latent functions from LCM to model heterogeneous outputs, where every distribution is completely specified by a set of parameters to be fitted. For example, a Bernoulli distribution is fully specified by the probability of success, and a Gaussian distribution is fully specified by mean and variance. The intractability is solved by variational inference and applying stochastic variational inference which is similar to GPLAR's inference scheme. While Skolidis et al. (2011) used probit likelihoods to model several binary variables, and used expectation-propagation to approximate the posterior. Additive multi-output GPs presented by Vanhatalo et al. (2018) utilized the Laplace approximation.

Apart from non-Gaussian likelihoods, nested GPs also come at a price with intractability although they allow for greater flexibility through non-Gaussian marginal densities. Authors of NARGP chose to maintain the computational cost by replacing the function values with a GP prior by the posterior mean from the previous inference level, which is similar in the case of GPAR. However, this setting reduces the problem to sequential maximum-likelihood estimation questions which are equivalent to fitting GPs in an isolated hierarchical way such that GPs at lower fidelities will not be updated even when more observations from higher fidelities arrive. Similar in GPAR, GPs of foregoing outputs will be fixed once the fitting is finished, which leads to its poor performance in closed-upwards missing observations. To allow communications of information (such as uncertainty) between fidelities (equivalent to channels in GPLAR), Cutajar et al. (2019) presented the multi-fidelity DGP (MF-DGP) which trains end-to-end. A graphical example with three fidelity levels is presented in Fig. 2.5a, where each fidelity level t has a different set of inputs $\{\mathbf{X}_t\}_{t=1}^{3}$. GPLAR can also have different input locations for different outputs (i.e. heterotopic data where evaluations are obtained by separate simulations), shown in Fig. 2.5b. The only modeling difference between GPLAR and MF-DGP is that each layer in MG-DGP is conditioned on inputs and evaluations only from the immediate previous fidelity. While in GPLAR, each layer is conditioned on evaluations from all preceding layers. Namely, one fidelity level in MF-DGP only communicates to other fidelities through its immediate former level, while a channel in GPLAR communicates to all previous channels explicitly and directly.

Although MF-DGP also uses variational inference and simple Monte Carlo method to propagate information, the treatments of inducing inputs are different. Since the intermediate

layers in MF-DGP has true meanings and unlike intermediate layers in original DGP where hidden variables are free to represent any extracted features, MF-DGP selects inducing points from available observations where previous fidelities are also observed and fixes inducing points during optimization. However, in GPLAR, more principled approach is used to enable the optimization of inducing locations and will be discussed in detail in section 3.1.



(a) MF-DGP (b) GPLAR

Fig. 2.5 (a): Graphical model of MF-DGP. $\mathbf{f}_l^t$ denotes evaluations of latent function in layer l at inputs with fidelity t. (b): Graphical model of GPLAR with different inputs for each output. $\mathbf{h}_l^t$ denote specific evaluations of latent function in output level l at inputs $\mathbf{X}_t$. For $\mathbf{h}_l^t$ whose l is larger than t, output is not observed and will be regarded as missing value. $\log p(y_{ln}|h_{ln})$ for such missing value will not be included while calculating the variational expectation (second term in Eq. 2.2). Passing of samples to missing values is marked by dashed lines as they are omitted.

Chapter 3

Theoretical Details, Alternatives and Extensions of GPLAR

3.1 Treatment of Inducing Inputs

Selection and optimization of the inducing inputs location are less explicit parts of the model. Recall that for the first layer corresponding to the GP mappings for first output y_1, there is only one single input, $\mathbf{x}$. Treatment of inducing inputs is standard and similar to other sparse approximation situations. The original GPAR model used fixed and regularly-spaced inducing inputs locations over time, since they are known to perform well for time-series datasets (Thang D Bui et al., 2014). However, for higher layers, the selections are less straightforward. As noted by Cutajar et al. (2019), since both points in the original input space (temporal space) and their **corresponding** evaluations of functions at previous output level are passed through the next layer, the inducing points of higher layers should also be intrinsically linked due to these correspondences. For example, suppose $\mathbf{z}_m$ is the inducing location of one of the inducing inputs of first layer, the following vector should be passed to layer l after previous $l-1$ propagation,

$$\begin{bmatrix} \mathbf{z}_n & f_1(\mathbf{z}_n) & f_2(\mathbf{z}_n, f_1(\mathbf{z}_n)) & \dots & f_{l-1}(\mathbf{z}_n, f_{1:l-2}(\mathbf{z}_n)) \end{bmatrix}$$

Since the inducing inputs are associated across layers, free optimization of inducing inputs at each layer is no longer appropriate in contrast to the case in DGPs. The original GPAR used the posterior predictive means of previous layers evaluated at inducing locations $\mathbf{Z}$ (which are fixed and evenly-spaced over time) as "optimized" inducing points. While MF-DGP selected inducing points from available observations which were fixed afterwards as mentioned in section 2.5.

Choose from Observations. Firstly, we used the same strategy employed in MF-DGP, where a subset of training data $\{\mathbf{x}_m, \mathbf{y}_m\}$ are selected as inducing points and fixed afterwards. Mean of the variational distribution over inducing inputs, $\mathbf{m}_l$, at each layer is also initialized to the corresponding observed output at dimension l, $\{y_{lm}\}$, while variances are initialized to be near zero, meaning start from being deterministic. There is some problem with this strategy, since inducing points can only be selected at where observations and also their previous levels are available, no inducing points will be placed when the observations are not closed-downwards. Taking a three-dimensional output with temporal inputs in interval $[0, 1]$ as an example, if y_1 and y_2 has missing values on interval $[0.4, 0.6]$, even if the corresponding values are present for y_3, there is no inducing points over $[0.4, 0.6]$ for the third layer, since evaluations of first and second latent functions are missing. This would unnecessarily introduce uncertainty at the third layer for interval $[0.4, 0.6]$.

Using GPAR Predictive Mean. To solve the problem mentioned above, we then tried to use the posterior predictive mean from GPAR. This allows us to select inducing locations anywhere regardless of observations being closed-downwards or upwards. One concern might be whether a large bias in GPAR's posterior predictive mean will pass false information to GPLAR, since GPAR performs poorly if there are missing values in the first few channels. Experiments on synthetic data have shown that even if inducing inputs are initialized incorrectly, parameters of variational distributions, especially the mean parameter, will correct them after observing outputs from latter channels. The experimental results will be further discussed in section. **??**. This again shown that GPAR's sequential optimization prevents communication of information through channels, while GPLAR improved this and allow GPs at preceding layers to be updated in order to have predictions at latter channels "closer" to observations (w.r.t to higher variational expectation). However, although predicted mean has been improved, uncertainty are not well-calibrated. We would expect the model to produce high uncertainty to reflect lack of data on missing areas, but current GPLAR can some times give an incorrect result with high confidence in channels whose following outputs are all observed (closed-upwards). This problem will be further discussed in the next section.

Enable Optimization of Inducing Locations. To make the optimization of inducing locations possible, we need to relate inducing inputs over output dimensions to inducing inputs over the original input space. Inspired from the last strategy that uses posterior predictive mean of GPAR, we can use posterior mean of GPLAR. Since $q(\mathbf{u}_l)$ is taken to approximate the posterior $q(\mathbf{u}_l|\mathbf{y})$, summarizing sufficient statistics from the training observations $\mathbf{y}$, we

could take mean of $p(\mathbf{u}_l)$ as inducing inputs to the next layer $l+1$. The correspondence is also clear as $q(\mathbf{u}_1)$ is the posterior distribution over inducing locations $\mathbf{Z}$, such that,

$$
\begin{aligned}
q(\mathbf{u}_1)_m &= q(u_{1m}), && \text{where } u_{1m} = f_1(\mathbf{z}_m) \\
q(\mathbf{u}_2)_m &= q(u_{2m}), && \text{where } u_{2m} = f_2(\mathbf{z}_m, \mathbb{E}[q(u_{1m})]) \\
&\vdots \\
q(\mathbf{u}_L)_m &= q(u_{Lm}), && \text{where } u_{Lm} = f_L(\mathbf{z}_m, \mathbb{E}[q(u_{1m})], \ldots, \mathbb{E}[q(u_{(L-1)m})])
\end{aligned}
$$

The resulting inducing inputs to each layer l is as follows,

$$\begin{bmatrix} \mathbf{Z} & \mathbf{m}_1 & \ldots & \mathbf{m}_{l-1} \end{bmatrix}$$

where $\mathbf{m}_l$ denotes mean of each variational distribution. In this setting, inducing inputs are "automatically" optimized since they are variational parameters themselves, except for $\mathbf{Z}$ which are inducing inputs over the original input space. Experiments have shown little overhead in computation after enabling optimization over inducing locations. Optimizing the inducing locations are beneficial in high-dimensional problems (Nguyen et al., 2014). More comparison and discussions on performance over real datasets with higher dimensional inputs are in Chapter. 4.

3.2 Bi-directional GPLAR

The current settings of GPLAR has difficulties and deficiencies while dealing with real datasets with longer dimensional outputs. As mentioned before, although the end-to-end training across all channels enables update of GPs for outputs that are first fed to the model after observing data from outputs arrived at a later stage, when outputs' dimension becomes larger, e.g. 10 outputs, backpropagation of errors at the 10^{th} level to the 1^{st} channel becomes difficult. For example, when there is strong dependencies between the 10^{th} output and the 8^{th} output, then the 8^{th} output and the 5^{th} output, and finally the 5^{th} output and the 1^{st} output, updating variational parameters of the first output would be difficult after observing data points at the 10^{th} channel. However, an update in the first layer would be easy to show effect on predictions at the 10^{th} layers. Since the outputs are processed according to a particular order, each channel tends to have most effects on the previous channel. On the other hand, in the real world, correlation between two variables is often asymmetric, for example, the number of people suffering from lung cancer is greatly potentially correlated

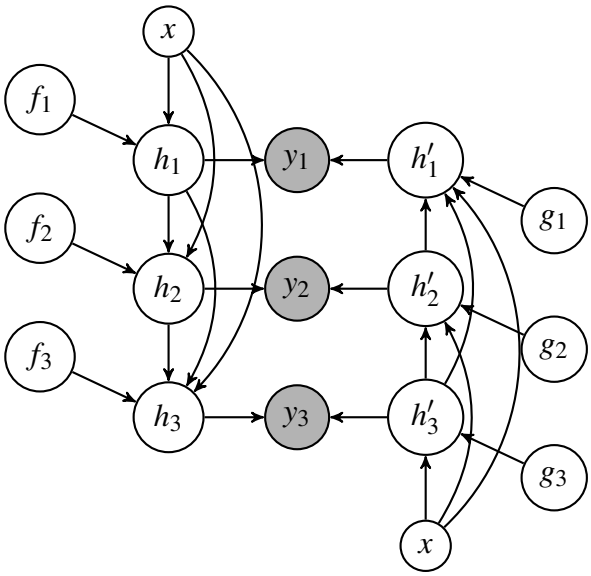


Fig. 3.1 Graphical model of Bi-directional GPLAR

to the number of people smoking, while the number of people smoking would have a more complex dependency on other factors, instead of also having a significant correlation with people with lung cancer. Hence, a particular ordering of outputs in GPLAR would restrict the modelling from discovering dependencies in both directions.

3.2.1 Inspired by Bi-directional RNN

To solve the problem, we take inspiration from the bi-directional RNN model. RNN is specially and elegantly designed to deal with sequential data that embodies correlations between points over the sequence. Original RNN can be modified to use input information from future by delaying the output for certain time frames. However, adding too many frames from the future might distract RNN's modeling power. The basic idea of bidirectional recurrent neural nets (BRNNs) raised by Schuster et al. (1997), is splitting each state neuron to take charge of the forward and the backward direction separately, both of which are connected to the same output. This enables every point to have complete and sequential information from both the past and the future.

Taking the bi-directional form of GPLAR, since the sequential character is along the output dimension instead of the temporal space as in the standard RNN situation, we run another GPLAR in reverse order whose graphical verison is shown in Fig. 3.1. The hidden variables from both directions at each layer are aggregated and added with noise to produce observations if in regression problem. The complete probabilistic model is as follows,

$$
\begin{aligned}
p(f_l|\theta_l) &= \mathcal{GP}(f_l;\mathbf{m}_l,\mathbf{K}_l), l = 1,\ldots,L \\
p(g_l|\theta'_l) &= \mathcal{GP}(g_l;\mathbf{m}'_l,\mathbf{K}'_l), l = 1,\ldots,L \\
p(\mathbf{h}_l|f_l,\mathbf{X},\mathbf{h}_{1:l-1},\sigma^2) &= \prod_n \mathcal{N}(h_{ln};f_l(\mathbf{x}_n,h_{(1:l-1)n}),\sigma_l^2) \\
p(\mathbf{h}'_l|g_l,\mathbf{X},\mathbf{h}'_{l+1:L},\sigma^2) &= \prod_n \mathcal{N}(h'_{ln};g_l(\mathbf{x}_n,h'_{(l+1:L)n}),\sigma_l^2) \\
p(\mathbf{y}_l|\mathbf{h}_l,\mathbf{h}'_l) &= \prod_n \mathcal{N}(y_{ln};h_{ln}+h'_{ln},\sigma_{y_l}^2)
\end{aligned}
$$

Now, GP in each layer would have kernels on all other channels. This structure not only makes the model richer by enabling asymmetric correlations but also makes learning easier since information flows from both directions and source of updates is not restricted to back-propagation from channels at a distance. Apart from better predictive mean, the uncertainty estimates should also be better-calibrated as discussed below.

3.2.2 Uncertainty in Former Channels

Firstly, although experiment results do confirm better predictive mean at missing areas in the first few channels when the number of outputs is low using simple GPLAR than GPAR, predictive uncertainty is sometimes not well-calibrated and is especially over-estimated. We looked into a case when the model did give predictions with low-confidence to reflect lack of data in layers where following channels are observed. In this case, a simple DGP with one-dimensional intermediate layers are added in order to propagate information from last output to each previous output. In Figure 3.2, electrodes FZ, F1, and F2 are placed in the first three rows, whose measurements for time from 0.6 to 0.8 are missing. The observations in this time area are called closed-upwards, since measurements from F3 onwards are observed. The observations in time interval $[0.8, 1.0]$ are neither closed-upwards nor closed-downwards, and dependencies between (FZ,F1,F2) and (F3,F4) are still possible to be discovered from GPAR, hence the predictive mean and variance of GPAR are well-fitted. While for GPLAR, all predictions are over-confident except for electrode F2. To understand why uncertainty for $F2$ is correctly reflected, we looked into the learnt variance parameter of kernels at each layer in GPLAR. Large variance would indicate strong dependencies between variables and when the kernels between outputs have zero variances one would recover independent GPs.

As shown in Table 3.1, variances of kernels between F2 and all its following electrodes are of low values. However, there are electrodes who depend quite significantly on either F1 or

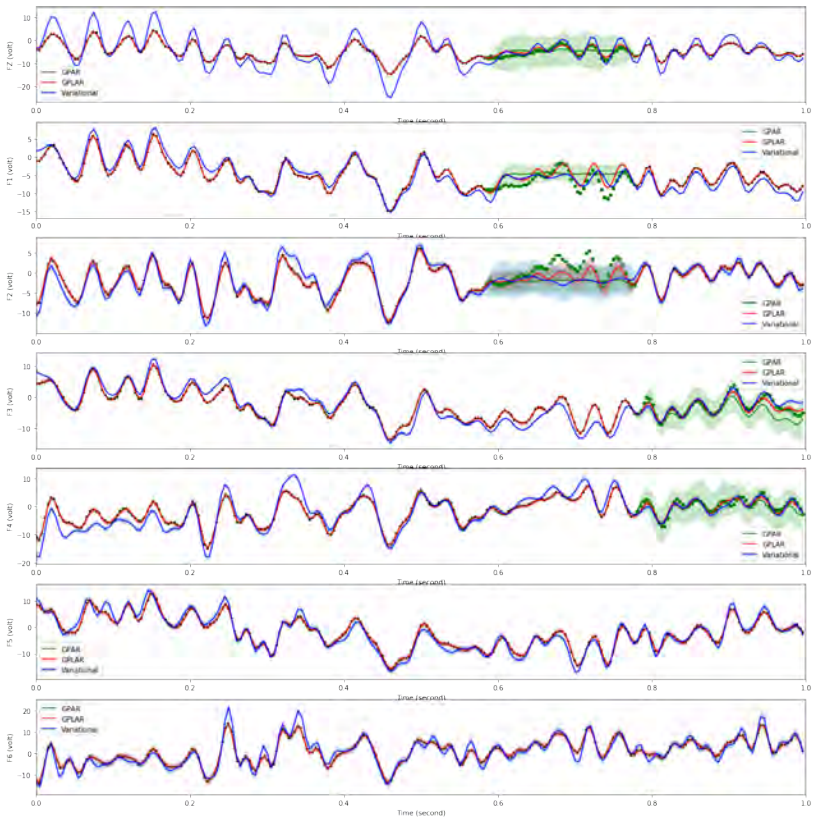


Fig. 3.2 This is a real dataset example consisting of measurements (in voltage) over time (in seconds) from 7 electrodes placed on a certain patient's scalp. Black dots denote training points fed to the model, while green dots represent missing values taken away. Red and green lines are predictive mean from GPAR and GPLAR respectively, while blue lines are variational mean (labeled as "q_mu") of GPLAR evaluated at inducing points in the forward direction. The shaded area corresponds to 95% confidence interval.

FZ. For example, F3 depends similarly on time and F1; F4 depends both significantly on F1 and FZ. These strong dependencies will have effects such that variance of predictions in the preceding channels will be squeezed in order to have more certain predictions in latter channels since data points are only observed in latter channels. For example, if we denote one particular prediction of hidden variables in F1 as h_{2n}, corresponding to input $\mathbf{x}_n \in [0.6, 0.8]$, variational expectation of the corresponding prediction in F3 that depends on h_{2n} are as follows,

$$\frac{1}{S}\sum_r \int \log p(y_{4n}|h_{4n}) p(h_{4n}|\mathbf{x}_n, h_{1nr}, h_{2nr}, h_{3nr})$$

where $(h_{1nr}, h_{2nr}, h_{3nr})$ are sampled according to Eq. 2.5. Since y_{4n} is observed, predictive mean of h_{4n} is learnt to be closer to y_{4n}, variance of h_{4n} will also be pushed to zero after

Output	Temporal kernel variance	Linear kernel variances over output	Non-linear kernel variance over output
FZ	4.8E+00		
F1	3.6E-01	FZ:7.1E-01	1.9E-06
F2	3.1E-01	FZ:2.4E+00 F1:2.1E-01	4.1E-06
F3	6.0E-01	FZ:5.9E-07 F1:6.3E-01 F2:7.4E-07	6.5E-07
F4	1.9E+00	FZ: 3.4E+00 F1:1.3E+00 F2:8.2E-06 F3:7.9E-05	1.0E-06
F5	2.8E-01	FZ:6.0E-07 F1:3.3E-05 F2:4.6E-07 F3:5.1E-01 F4:5.8E-07	1.9E-06
F6	5.9E-01	FZ:6.1E-04 F1:2.9E-05 F2:5.5E-06 F3:4.1E-07 F4:1.9E-01 F5:3.1E-04	2.3E-06

Table 3.1 Parameter values of variances of kernels learnt training on models and datasets from Figure. 3.2. Kernels between F2 and other electrodes are highlighted by red color.

predictive mean gets closer to the observation as likelihoods are maximized. If variance of h_{2n} remains to be high, h_{2nr} would take on several values, and "high" correlation between F1 and F3 would leads to varying predictive mean of h_{4n} which brings down likelihood. Hence, variance of h_{2n} is pulled down, ignoring the fact that no enough data is observed around the area. Another interpretation would be that y_{4n} are treated as direct observations of F1, while model should be able to separate direct observations in the same channel from observations of latter channels who only utilize the hidden variables at the current layer as inputs. As for the hidden variable in F2, h_{3n}, since correlation with any latter channel is low, no latter observations are treated as direct observations. Regularization term in Eq. 2.2,

$$-KL[q(\mathbf{u}_3)\|p(\mathbf{u}_3)]$$

would pull the variational distribution along the missing area towards prior distribution. DGP in another direction has corrected the predictive mean as correlation between F3 and F2 is discovered, while variance from the forward direction remains. In conclusion, since some hidden variables h are combining information from all previous layers, but is modeled by a single function, the variance is directly modified by observations arrived at a later stage. If later outputs' observations only partially update or partially depend on previous hidden variables, where the lack of data can still be retained, the model can give better-calibrated uncertainty estimates. Bi-directional GPLAR is expected to achieve this by decomposing the hidden variables into additive sum of GPs from two directions. If there is missing value in the first output, whose hidden variables are h_{1n} and h'_{1n}, even if variance of h_{1n} is reduced by latter observations y_{2n} and y_{3n}, variance of h'_{1n} would still be pushed to prior. Because h'_{1n} is the last hidden variable in the reverse direction, and is never used as input to any other layer. Hence, bi-directional GPLAR can have better uncertainty estimates in the first few channels, but might still fail for missing outputs in the middle channels. If dependencies are discovered both with latter outputs in the forward direction and with previous outputs in the backward direction, variance of both hidden variables will be eliminated.

3.2.3 Correlated/Repeated Kernels

However, although theoretically, bi-directional GPLAR should produce better-calibrated uncertainty, in experiments they still can perform badly. We suspect is another problem spotted from the graphical model itself, which might be the cause of the aforementioned behavior. If one directly write down the probabilistic formula of the last layer in Fig. 3.1 combing two directions as follows,

$$p(y_{3n}|\mathbf{x}_n, h_{1n}, h_{2n}) = \mathcal{N}(y_{3n}; f_3(\mathbf{x}_n, h_{1n}, h_{2n}) + g_3(\mathbf{x}_n), \sigma^2_{y_3})$$

It is easily observed that there are repeated kernels over temporal space. The kernels in f_3 are additive which contains a distinct kernel over input and g_3 is over the same temporal space. Although sum of squared exponential kernels can represent discovering of different characteristic length-scales, correlated kernels are also possible to cause over-confidence.

3.3 Additive GPLAR

We first summarise the deficiencies of simple bi-directional GPLAR and raise possible direction of solutions,

- Since the kernels over outputs in GPLAR are multi-dimensional and previous channels evaluated at $\mathbf{x}_n$ are passed to later channels which also depends on $\mathbf{x}_n$ locally, this intrinsic linkage requires inducing points to be also intrinsically linked. The methods mentioned in section. 3.1 either disables the optimization over inducing locations or only has freedom to optimize over the temporal space and is restricted over output space. More flexibility can be achieved if the inducing points over outputs are not internally connected to inducing points over input.

- When there are missing values in the middle channel which has strong dependencies with latter and previous channels whose data points over the same input are observed, predictions at the current layer will have under-estimated uncertainty resulted from failure of model to distinguish direct observations from current output and those indirect.

- Although bi-directional version allows for flows of information from all outputs, repeated using of input can hurt model's complexity, making it more susceptible to overfitting.

An alternative of GPLAR is to make the combined hidden variables explicit by decomposing them into additive components, each of which is individually modelled by a distinct GP. The decomposition can appear in many forms with different depth.

Combined	GPLAR	Additive	Fully Additive
h_1	$f_1(\mathbf{x})$	$g_1(\mathbf{x})$	$g_1(\mathbf{x})$
h_2	$f_2(\mathbf{x}, h_2)$	$g_2(\mathbf{x})$ $+h_{21}(h_2)$	$g_2(\mathbf{x})$ $+h_{21}(g_1)$
h_3	$f_3(\mathbf{x}, h_2, h_3)$	$g_3(\mathbf{x})$ $+h_{31}(h_1)$ $+h_{32}(h_2)$	$g_3(\mathbf{x})$ $+h_{31}(g_1)$ $+h_{32}(g_2)$ $+h_{321}(h_{21})$

Table 3.2 Decomposition of single GP on hidden variable. First column is the original hidden variable which is directly connected to observation through likelihood. The second column is the original setting of GPLAR. The third column decomposes the single GP into a temporal function and functions explicitly on previous hidden variables. The forth column further decomposes where there is a function on every previous function outputs

In the original GPLAR, a single GP is used to model the output whose input is multi-dimensional. The additive version breaks the multi-dimension to multi-GP processes, which is equivalent to changing a product of kernels to a first-order additive kernel in the non-linear kernel case. Functions drawn from a high-order kernel (ARD SE) have less long-range structure, while draws from additive low-order kernel tends to have more global trends (D. K. Duvenaud et al., 2011). Since sum of orthogonal single-dimensional kernels represents low-order correlations, in the fully additive version which is the fourth column in Table. 3.2, deeper-order dependencies are also modeled, for example, $h_{321}(h_{21}(g_1(x)))$ is a GP modelling correlation between the third output and the part in second output explained by first output.

The fully additive transformation breaks the combination and places a GP on every component. This breakage removes constraints on internally linked inducing points, such that in the example presented in Table. 3.2, the inducing points for g_3 are over time, and the inducing points for h_{31} are over the output space of g_3. These two sets of inducing points do not necessarily have one-to-one correspondence. However, this breakage also introduces new sets of variational parameters which significantly increase the number of variables that require more careful initialization strategies. g_i is unique to each output i that corresponds to the temporal correlation in the original model. Hence, this separation also enables generalisation to bi-directional version without using repeated kernels over input as shown in Fig. 3.3. It is obvious that temporal kernel only exists in function g individually in each channel, and every output has at least one kernel with g in all other channels. Since now the hidden variables come from addition of multiple GPs, the source of variance also becomes multiple. When observation in the third output is present but missing in the first output, variance of hidden variables h_{1n} will only decrease in part represented by g_1, while variance of h_{12}, h_{13}, h_{123} will remain. However, lack of data in g_1 is still unable to be reflected through higher-order functions.

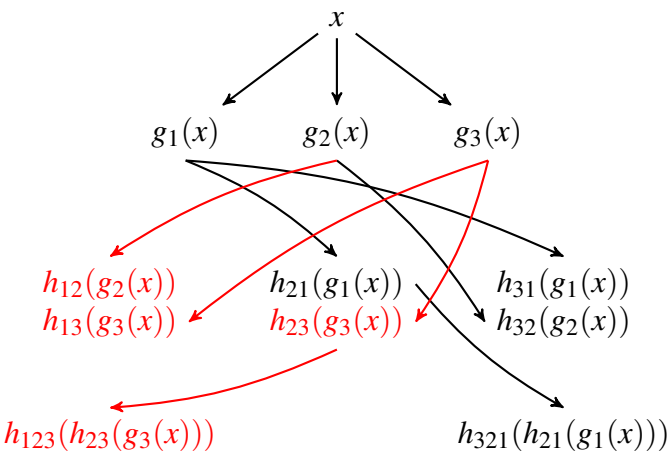


Fig. 3.3 Graphical model of Bi-directional Additive GPLAR. The red part denote additive GPLAR in backward direction.

Chapter 4

Experiments

4.1 Overview

All experiments implemented in this chapter utilize the library GPflow 2. (De G. Matthews et al., 2017), which takes advantage of the benefits of TensorFlow 2.0. In all cases, we evaluate the held-out predictive log-likelihood (HLL) and the standardised mean squared error (SMSE) on a held-out test datasets for each output. As the predictions of GPLAR are mixtures of Gaussian by propagating S samples through all layers as described in Eq. 2.4, the log-likelihood is approximated by,

$$\text{HLL} = \sum_n \log \frac{1}{R} \sum_r \int p(y_{lnr}|h_{lnr}) \mathcal{N}(h_{lnr}; \mu_{h_l|h_{1:l-1}}, \sigma^2_{h_l|h_{1:l-1}})$$

where μ and σ^2 follow Eq. 2.5 and R denotes number of samples which is usually 100. The SMSE is simply the mean squared error normalised by the variance of the truth:

$$\text{SMSE} = \frac{\sum_n (\sum_r y_{lnr}/R - y_{in})^2}{N \times (\bar{y}_i - y_{in})^2}$$

where $\bar{y}_i = \frac{1}{N} \sum_n y_{in}$. During training, we use Adam (Kingma and Ba, 2014) optimizer, with default parameter values and an exponentially decaying learning rate initialized as 0.01. Different strategies of initialization and optimization of inducing points locations as discussed in section. 3.1 are compared in detail. Lengthscales and variances of nonlinear kernels over outputs are initialized to 1., while variances of linear kernels over outputs are initialized to 10.. Initialization of variances of kernels are important, such that if variances of kernels over outputs are much smaller than that of kernels over input, model would have difficulties in finding the expected correlations between outputs.

The techniques of using stochastic variational inference (Hoffman et al., 2013; Hensman, Fusi, et al., 2013) require careful strategies when optimizing hyperparameters of kernels, variational parameters and inducing locations and noise variance of likelihoods all together. In the implementation of DGPs, variances of the variational parameters of intermediate layers are initialized to very small values, which is equivalent to starting as a single layer GP. Similar to this idea, variances of the variational parameters of each layer of GPLAR are initialized to be close to zero, which is giving the initial inducing points function value (either from observations or GPAR posterior mean) full confidence to ensure stability during the early iterations. The noise variances of likelihoods are also kept small at the beginning. Since the first term in Eq. 2.2 is a sum of independent terms, its computation can be distributed and unbiased noisy estimations of the objective and the gradients can be obtained by sub-sampling with a scaling factor, $N/|\mathcal{B}|$, where $|\mathcal{B}|$ denotes the minibatch size.

Normalization is important as multi-outputs regression usually has very distinct distributions for different outputs and it is necessary that every output is normalised before being fed into the model. Otherwise, distribution of one output need to be learnt each time it is passed to the following layers, and slows down learning. Hence, we applied whitening on every output for the training data, and kept records of mean and variance of training samples. During prediction time, these mean and variance are added or multiplied back to perform easy "un-normalization".

4.2 Synthetic Data Experiments

4.2.1 Synthetic Data from Functions

We first tested GPLAR on the same set of data produced by synthetic functions as follows, which was also used in the original GPAR paper to demonstrate model's ability of discovering dependencies between outputs,

$$
\begin{aligned}
y_1(x) &= -\frac{sin(10\pi(x+1))}{2x+1} - x^4 + \varepsilon_1 \\
y_2(x) &= \cos^2(y_1(x) + \sin(3x)) + \varepsilon_2 \\
y_3(x) &= y_2(x)y_1^2(x) + 3x + \varepsilon_3
\end{aligned}
$$

To begin with, variances of noises $\varepsilon_{1:3} \overset{\text{iid}}{\sim} \mathcal{N}(0, 0.05)$ are fixed to be small values. Since y_2 and y_1 is combined and used in y_3 in a much simpler form as compared to the function with only x as input to predict y_3 directly, GPAR and GPLAR can exploit these dependencies between (y_1, y_2) and y_3 while independent GPs would struggle with this complicated structure between x and y_3. Fig. 4.1 shows GPAR and GPLAR fit to 100 data points randomly drawn from the above functions. It is easy to observe that GPLAR has well-calibrated uncertainty such that observations are mostly covered by the shaded area, while GPAR tends to overfit and fail to reflect the noise which accumulates along the outputs. It is also surprised to discover that GPAR becomes more and more unstable when noise level increases. As shown in Fig. 4.2, with same level of noise variance but only different random seeds for drawing random data points from the functions, the left figure shows severe over-fitting in the third output such that the model is trying to fit to noise, while the right figure shows significant under-fitting in both y_2 and y_3 where GPAR performs similarly to independent GPs. GPLAR performs more stable and produce similar results with similar held-out log-likelihood even with different datasets produced by different seeds.

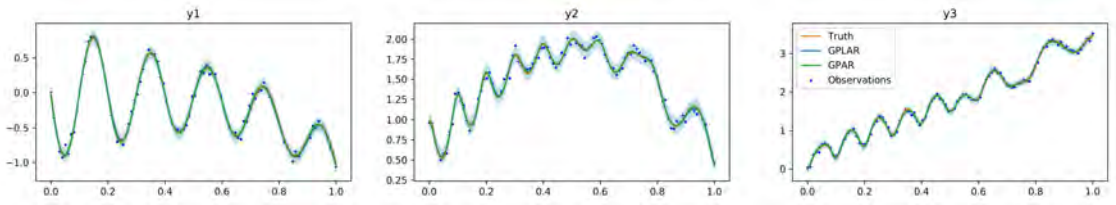
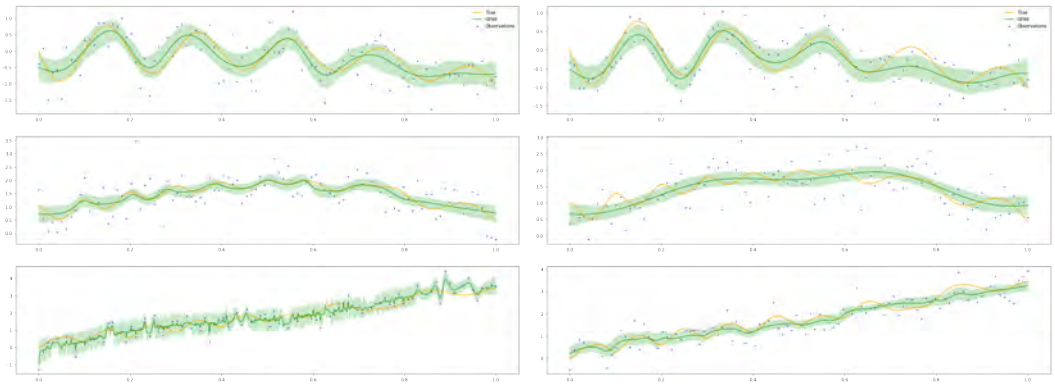


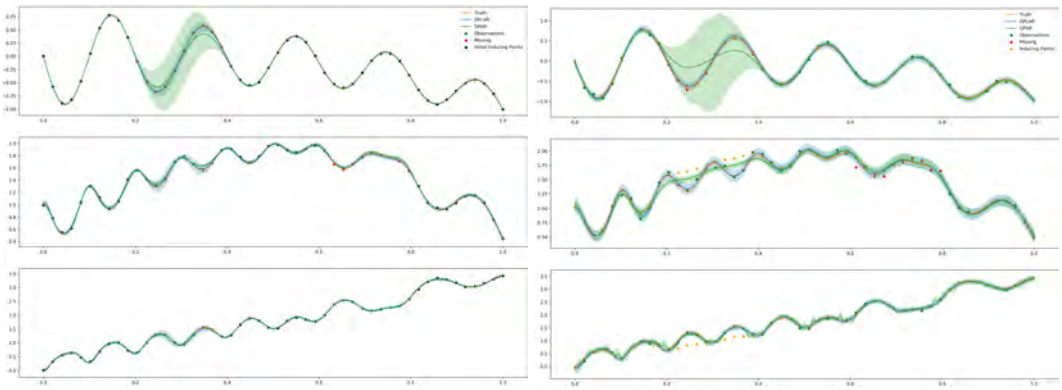
Fig. 4.1 Dataset from synthetic functions: GPAR vs GPLAR predictions. Blue dots are observations. Shaded area represents 95% condifence interval



(a) Overfitting Example (b) Underfitting Example

Fig. 4.2 Overfitting and underfitting examples for GPAR

We then test the ability of handling missing values to see whether GPLAR can produce predictions on missing output after observing other outputs and applying the discovered dependencies. As mentioned in section 3.1, if inducing points are chosen from observations, all its previous outputs should be available. This leads to unnecessary uncertainty as shown in Fig. 4.3a. Since observations with input ranging from 0.2 to 0.4 are missing for y_1, no inducing points are located in this area for all following outputs. The uncertainty estimates shown by blue shaded area is larger for y_2 and y_3 with $x \in [0.2, 0.6]$ than other locations, despite the fact that observations are present. If inducing points are initialized using GPAR's posterior mean of evenly-spaced inducing locations, aforementioned problem will be solved. When noise variance is as high as 0.05, even if posterior mean from GPAR has high bias, as shown in Fig. 4.3b for y_1 and y_2 with $x \in [0.2, 0.6]$, GPLAR successfully updates its variational mean after observing data points from y_3. GPAR would fail to do so since every layer is fitted sequentially. However, it is clear that uncertainty estimates from GPLAR for both the first and the second output in the missing area are under-estimated.



(a) Initialize from observations (b) Initialize from GPAR posterior mean

Fig. 4.3 Handling missing values with two inducing points strategies. Missing values are placed in $[0.2, 0.4]$ for first output and in $[0.6, 0.8]$ for second output, labelled using red dots. Noise level for the two datasets are 0.01 and 0.05 respectively. Black dots in the left figure represent inducing points from observations. Yellow dots in the right figure represent the final variational mean parameter value for each inducing location.

Output	HLL	
	GPAR	GPLAR
y_1	-200.290	-175.794
y_2	-314.689	-490.632
y_3	-353.180	-408.538
y_4	-448.052	-347.413
y_5	**-54226.6**	**-939.300**

Table 4.1 Held-out log-likelihood for every output: GPAR vs GPLAR

4.2.2 Synthetic Data from GPs

A more theoretical approach is to draw synthetic data directly from a GPLAR model. Suppose we have,

$$\begin{aligned} k_1(\mathbf{x},\mathbf{x}') &= k_{SE}(\mathbf{x},\mathbf{x}') \\ k_2((x,h_1(\mathbf{x})),(\mathbf{x}',h_1(\mathbf{x}'))) &= k_{SE}(\mathbf{x},\mathbf{x}')+k_{SE}(h_1(\mathbf{x}),h_1(\mathbf{x}')) \\ k_3((x,h_{1:2}(\mathbf{x})),(\mathbf{x}',h_{1:2}(\mathbf{x}'))) &= k_{SE}(\mathbf{x},\mathbf{x}')+k_{SE}(h_{1:2}(\mathbf{x}),h_{1:2}(\mathbf{x}')) \end{aligned}$$

where k_{SE} denotes squared-exponential kernel. With zero mean function in each layer, we randomly draw samples layer by layer and the results are shown in Figure. 4.4. Many test points and even some training observations falls outside GPAR's 95% confidence interval. If linear kernels over outputs are used instead of SE kernel, and with large variance in each dimension, a small change in the first output would lead to large fluctuations in following outputs. A five outputs example is presented in Fig. 4.5. If we set noise variance for the former four outputs to be large and small for the last output, GPAR would fail to consider noise from previous outputs that leads to variance in the last output and give severely over-confident predictions. To numerically compare performance of the two models, we use the held-out log-likelihood. The held-out log-likelihoods (HLL) corresponding to Fig. 4.5 are presented in Table. 4.1. It is observed that HLL of GPAR for last output exploded negatively, while GPLAR successfully propagated uncertainty through layers and gave moderately confident predictions near training observations and predictions with larger uncertainty far away from observations.

To better assess performance of two models against the noise level, we calculate the held-out log-likelihood on test datasets versus the variance of noise ε (The datasets has same form as show in Fig. 4.4). As mentioned in the previous section, results of GPAR are unstable and can give different predictions using different seeds of random sampling. Hence, we run

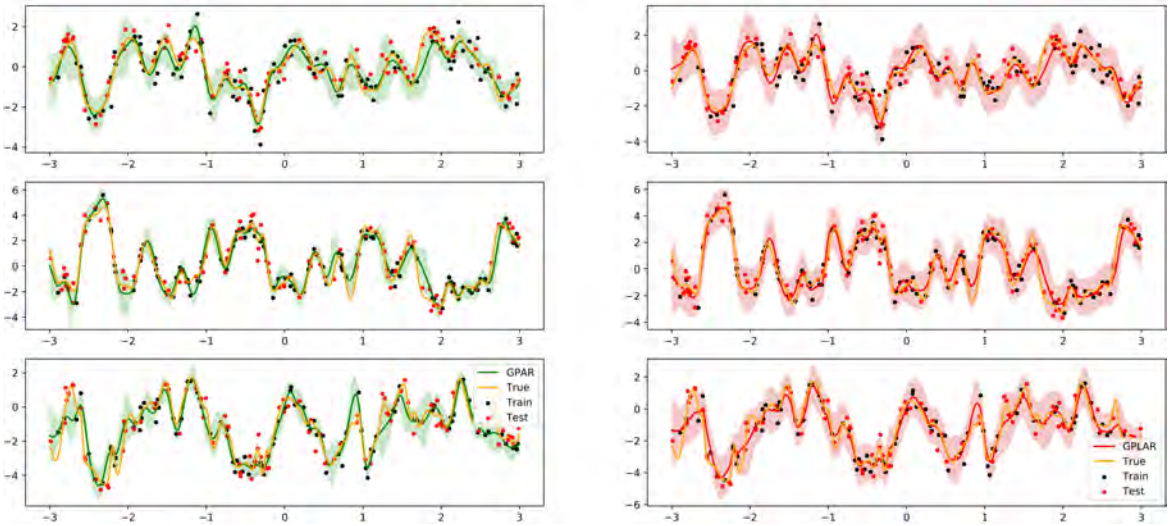


Fig. 4.4 Dataset from synthetic GPs with non-linear kernel over outputs: GPAR(left) vs GPLAR(right) predictions. Noise level is high(variance=0.5). Black and red dots denote training observations and test points respectively. Green lines and shaded area in left figure is GPAR's results, red lines and shaded area in right figure is GPLAR's results.

GPAR 100 times independently using different seeds for each variance level. It is observed from Fig. 4.6 that the instability of GPAR increases along outputs, and the lower bound of held-out log-likelihood also decreases along outputs, which suggests that using noisy outputs from previous layer harms GPAR's predictions for next output. Since GPLAR always gives same predictions after convergence, only one run is performed and its held-out log-likelihood is observed to always overlap or locate higher than the upper bounds of GPAR. Strange behavior is spotted in the third output of GPAR in Fig. 4.6, such that held-out log-likelihood is extremely negative when noise variance is close to zero. It turns out that predictive variance of the third output are all near zero (in $1e-7$ level), making the likelihood explode in the negative direction if the true observation is only slightly away from the predictive mean. While predictive variances of GPLAR are at an appropriate level. These reults all indicate that GPLAR handles noise from observations more carefully, and produce predictions that are more robust to under-fitting or over-fitting.

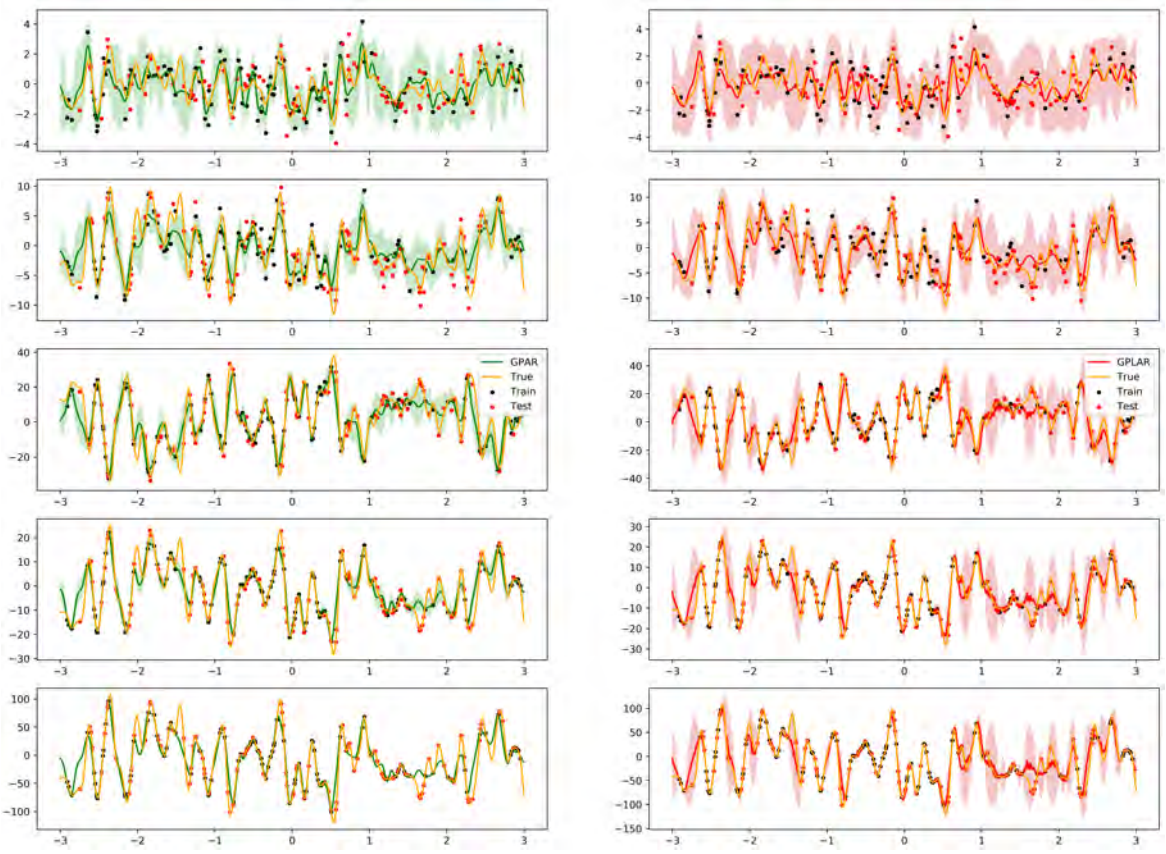


Fig. 4.5 Dataset from synthetic GPs with linear kernel over outputs: GPAR(left) vs GPLAR(right) predictions. The first four outputs has high noise and the last with low noise.

4.3 Real-World Data Experiments

4.3.1 Base model comparison

In this section, we evaluate GPLAR's performance on two standard datasets commonly used to evaluate multi-output modelling power, and compare GPLAR against GPAR.

Electroencephalogram (EEG) dataset. [1] As mentioned in section 3.2.2 as an example, these are 256 measurements in voltage in one second from 7 electrodes mounted on a patient's scalp when the patient is presented with a certain image. We took the measurements from patient number 337, and use full 256 observations from electrodes F3-F6 and first 156 signals from electrodes, FZ, F1, and F2 as training points, and last 100 observations of FZ, F1, and F2 as the test points to predict. Fig. 4.7 visualize predictions for the three electrodes

[1] The EEG datset is available at https://archive.ics.uci.edu/ml/datasets/eeg+database

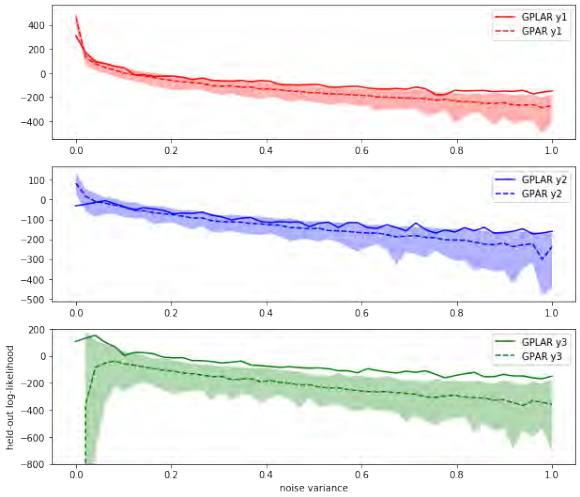


Fig. 4.6 Held-out log-likelihood vs Noise level for GPAR and GPLAR. Only one run is performed by GPLAR which is denoted by solid lines. 100 runs are performed by GPAR, whose 95% CI over log-likelihood is shaded and the median is labeled by dashed lines.

Output	SMSE		HLL	
	GPAR	GPLAR	GPAR	GPLAR
FZ	0.1340	0.1273	-135.7	-141.3
F1	0.3285	0.3130	**-663.1**	**-183.1**
F2	0.1536	0.1317	-132.4	-136.6

Table 4.2 SMSE and HLL for last three outputs: GPAR vs GPLAR for the EEG datasets

by only using non-linear kernels over outputs, and it is observed that predictions of GPAR over $F1$ are over-confident which leads to large HLL in Table. 4.2. While uncertainty over F1 from GPLAR is well-calibrated and the 95% confidence interval covers nearly every point except those in time $[0.9, 1.0]$. The SMSE for every output is also lower in results provided by GPLAR. As FZ, F1 and F2 are last three outputs fed to the model, posterior mean from GPAR already has high accuracy. Hence, fixed inducing inputs do not introduce any problems, and second method of optimizing inducing locations from section. 3.1 would produce similar results.

If we fit standard DGPs to this dataset, only one small modification is needed to deal with missing data, which is separating the calculations of the variational expectation terms in the final layer. Missing values are identified and skipped during the calculations. Since every output in such a multi-output task is correlated to input to some extend, skip connection is

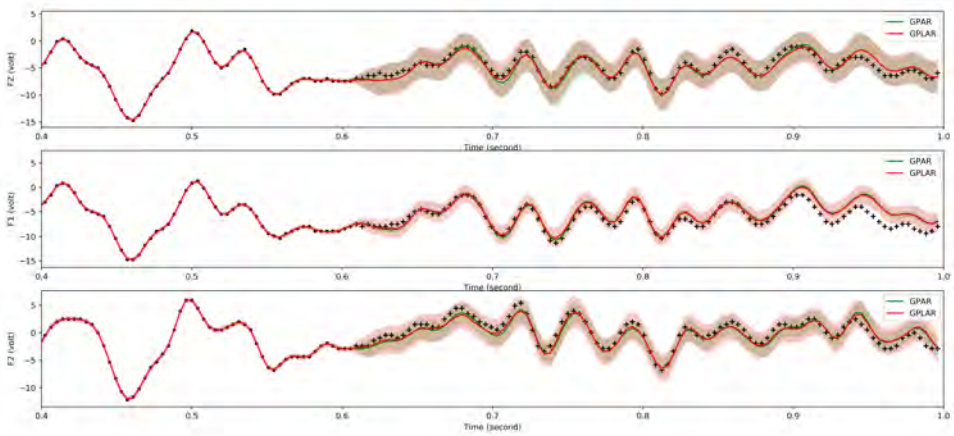


Fig. 4.7 Predictions for electordes FZ, $F1$, and $F2$ from the EEG datasets by GPAR(green) and GPLAR(red). Dots and crosses denote training and test points.

important which propagates the input to every intermediate layer and the final layer. As an alternative, Salimbeni et al. (2017) introduced an identity mean function at each intermediate layer, however, explicit propagation of input is validated to perform better although still worse than autoregressive models as shown in Fig. **??**. Despite that DGPs can discover non-Gaussian dependencies between inputs and outputs since all multi-output layers use independent outputs with shared covariance functions, such input and output wrappings are implicit and independent outputs prevent DGPs from exploiting dependencies between outputs, limiting their predictive performance on highly correlated data.

Exchange Rates Dataset. [2] The Pacific Exchange Rates Service keeps records of exchange rates of all currencies against US dollars every day. We extract exchange rates of ten international currencies and three metals in the year 2007, and take $50-100$th days for "USD/CAD", $50-150$th days for "USD/JPY" and $50-200$th days for "USD/AUD" as missing values to be predicted, and take information on all other days and full-year observations for all other currencies as training points. By using both linear and non-linear kernels over outputs, Fig. 4.8 presents predictions of GPAR and GPLAR for the three currencies with missing values. Although as shown in Table. 4.3, only SMSE of GPLAR over "USD/AUD" is significantly lower than that of GPAR, it is also observed that GPLAR give predictions with more uncertainty even outside the missing area, while GPAR would have high confidence

[2] The exchange rates dataset is available at http://fx.sauder.ubc.ca.

and model with more wigglings.

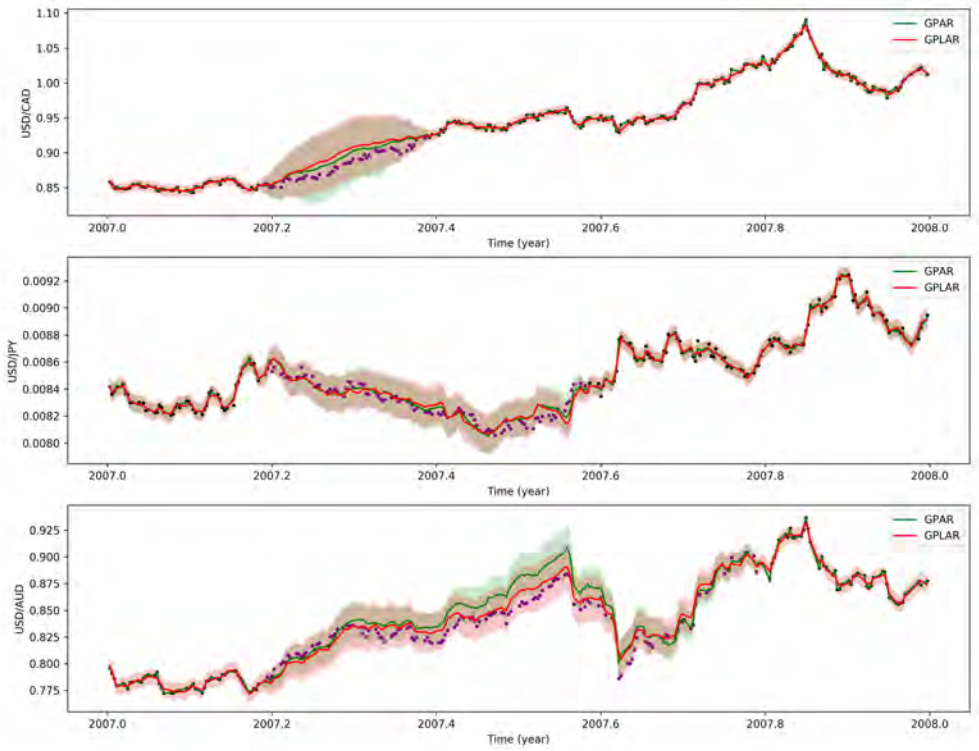


Fig. 4.8 Predictions for "USD/CAD", "USD/JPY" and "USD/AUD" from the exchange rates datasets by GPAR(green) and GPLAR(red). Black dots are training observations, purple dots are test points.

4.3.2 Bi-directional GPLAR on real-world data

Failure of one-directional GPLAR and ways to improve it. In this section, we evaluate the bi-directional GPLAR on multi-output datasets and evaluate the prediction power regardless of where the missing values are located, either closed-upwards, closed-downwards, or in the middle. To begin with, we first show misjudgement of one-directional GPLAR on the EEG dataset. As shown in Fig. 4.9a, observations in time $[0.6, 0.8]$ are closed-downwards since down-to electrode FZ, all its previous outputs are also observed. While observations in time $[0.4, 0.6]$ are closed-upwards since up-to output electrode F5, all its following outputs

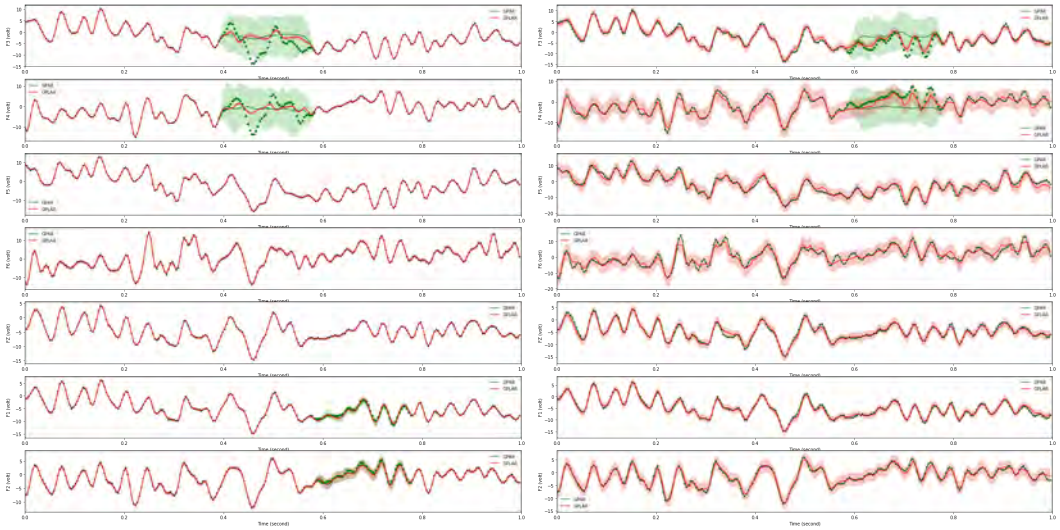
Output	SMSE		HLL	
	GPAR	GPLAR	GPAR	GPLAR
USD/CAD	0.0215	0.0439	148.60	153.95
USD/JPY	0.0170	0.0234	843.18	860.95
USD/AUD	**0.2089**	**0.0685**	523.97	464.58

Table 4.3 SMSE and HLL for outputs: GPAR vs GPLAR for the Exchange datasets

are observed. If simple one-directional GPLAR is run, it would produce neither correct predictive mean nor well-calibrated uncertainty on closed-upwards observations. It is suspected that F4, F5, FZ, F1, F2 (the later five) electrodes can perform perfectly only given input time, such that the variational parameters of the first two outputs are not updated. Hence, if the kernels over inputs for those later outputs are removed, GPLAR can produce better predictive mean, but the uncertainty outside the missing area would also by unnecessarily high. Forcing the outputs to learn through cross-channel can benefit predictions on missing values. Another method that would help is initializing mean of variational distributions from zero. This would also force later outputs to learn correct dependencies between itself and the former channel, because the intentional zero variational mean from previous layers would propagate same values and "flatten" the predictions which is not desirable. While starting from posterior mean of GPAR, although dependencies are found, since the kernels over outputs are not input-dependent, the discovered function relationships are not bijective, and model could be easily trapped in local minimums.

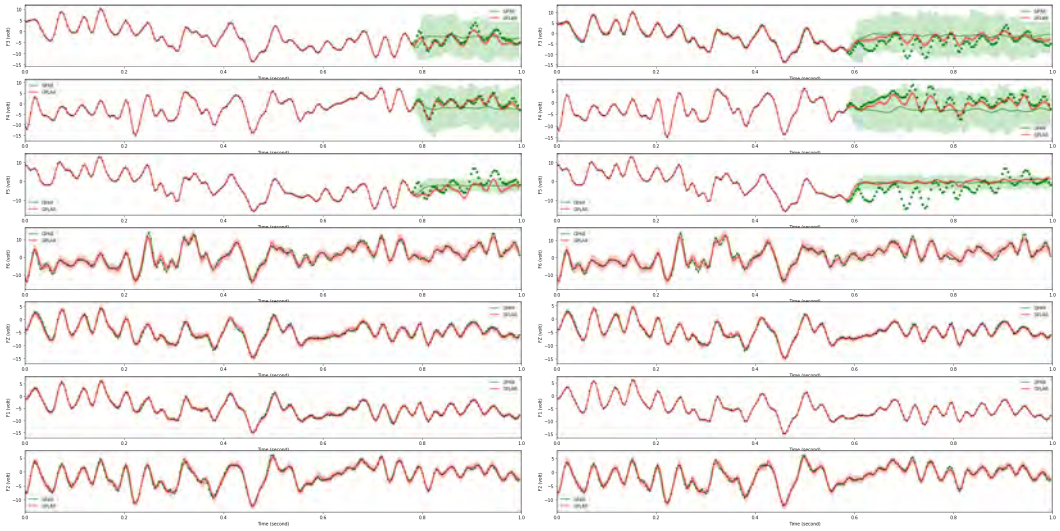
Fig. 4.9b shows the result after applying the first strategy where uncertainty is higher than usual everywhere since temporal kernels are removed. Fig. 4.10a shows the result after applying the second method, and it is clear that the uncertainty is over-estimated as the reason stated in section. 3.2.2. The predictive mean also becomes more flattened and has high bias, if the missing period is longer, shown in Fig. 4.9b.

Bi-directional GPLAR. As mentioned in section. 3.2, since information flows from both directions, dependencies between each output and all other outputs can be modeled. Furthermore, outputs are no longer modeled by single hidden variables but by two, one from the original direction and one from backwards, predictive variance from former outputs would not be entirely shrunk in order to have larger log-likelihood on observations from later outputs. Hence, bi-directional GPLAR should give both better predictive means and better predictive



(a) Wrong predictions: simple GPLAR (b) Better predictive mean: method 1

Fig. 4.9 The **left** figure shows wrong predictions on closed-upwards observations on the EEG datsets by one-directional GPLAR. Notice that the order of electrodes is different from previous experiments. The **right** figure shows better predictive mean after using the technique of learning only from cross-channels.



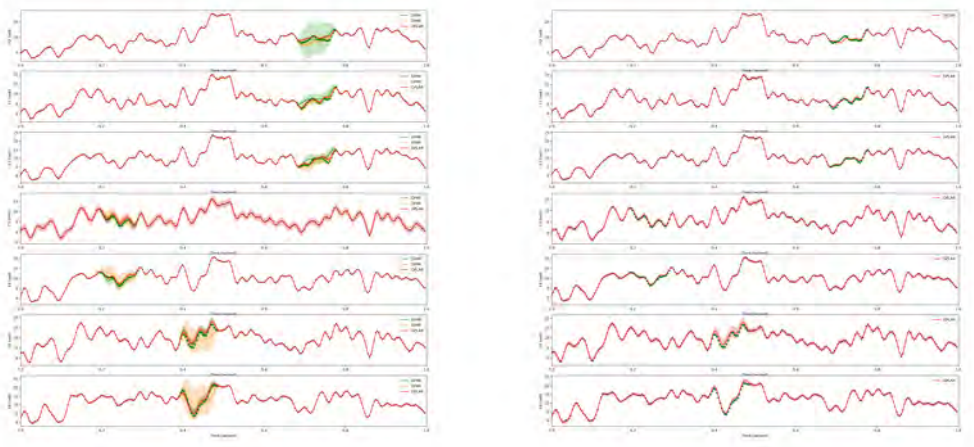
(a) Better predictive mean: method 2 (b) Longer predictions

Fig. 4.10 The **left** figure shows better predictive mean after using the technique of initializing the variational mean from zero. The **right** figure shows predictions on longer time period.

variances. We run the bi-directional GPLAR on the EEG and exchange-rate dataset, and evaluate the performance on a larger range with different patients for EEG dataset and different years for exchange-rate dataset. It is more impartial to compare performance of bi-directional

GPLAR with both GPAR running in a forward direction and also in a backward direction. It is shown in Figure. 4.11a that without using any method as mentioned in previous subparagraph, bi-directional GPLAR can learn better predictive mean. Although the uncertainty is larger as compared to the simple version, it is still too small to cover the missing observations.

The dependencies are so much harder for GPLAR to learn is because, for such situations, the model is expected to learn both dependencies between channels and also correlations over the input space. Simple data as EEG datasets can already achieve decent results by only considering functions over the input space, however, when missing data is of importance, cross-channel dependencies would play a much more important role than temporal functions. Hence, the model needs to choose carefully and balance the contributions of inputs and other outputs. On the other hand, maximizing free-energy, or lower bound of the evidence, with respect to such amount of parameters is sensitive to initialization and is more likely to be trapped by local minimum. Some experiments have shown that when model has been trapped inside local minimum, it would give prior distribution on missing areas which is equivalent to independent GPs as if variances of cross-channel kernels have been pushed to zero. A example of entirely learning through cross-channel kernels are shown in Fig. 4.11b, and it is clear that better predictive mean are accomplished.



(a) Bi-directional GPLAR (b) Remove temporal kernel

Fig. 4.11 The **left** figure shows predictions of bi-directional GPLAR (red) against GPAR running forward (green) and backward (orange) on EEG datasets of patient no.345. The **right** figure shows predictions of bi-directional GPLAR on the same data after removing kernels on temporal space.

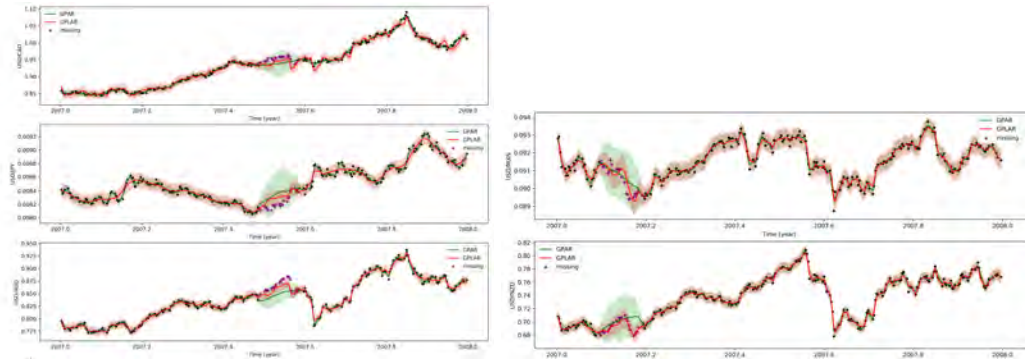
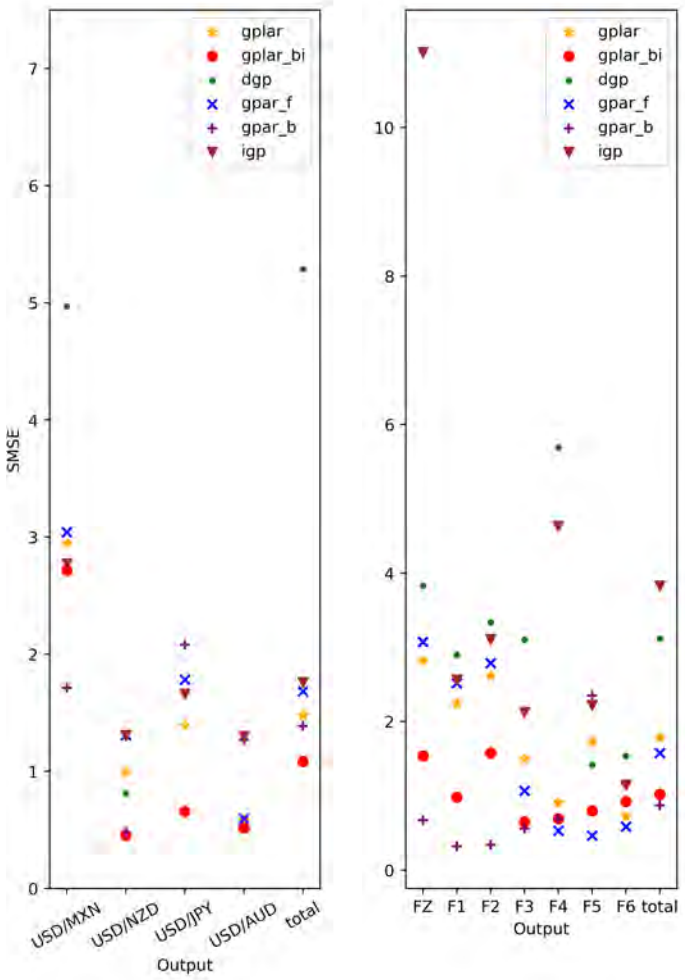


Fig. 4.12 Predictions of bi-directional GPLAR on exchange rate data set of year 2007. The shown predictions are first few outputs fed to the model, while the remaining outputs are not shown.

The Same test is run on the exchange rate datasets as shown in Fig. 4.12, where different areas of outputs and different ordering of outputs are tried. To make sure we are not getting the results by luck, we compare the performances of GPLAR, bi-directional GPLAR with independent GPs, GPAR in a forward direction, GPAR in a backward direction, and DGPs on the EEG data set averaged over 5 different patients and on exchange rate data set averaged over 5 years. The results are shown in Fig. 4.13. Since the EEG datasets contains almost noiseless measurements, GPAR in the forward direction performs better over closed-downwards observations. However, they perform significantly worse on closed-upwards observations. Bi-directional GPLAR would give the average performance between GPARs in two directions but it is not worth since GPLAR would take much more time. Hence, situations when noise is not dominant, one should use GPAR with natural outputs ordering. As for exchange rate datasets, since the observations are much noisier and sometimes even contain severe outliers, bi-directional GPLAR gives the best results for all four outputs (except for "USD/MXN", as correlations between inputs and outputs are sometimes hard to find for all models). Bi-directional GPLAR also gives the best results averaged over all outputs with closed-upwards and closed-downwards observations. It is obvious that bi-directional GPLAR is better than deep GPs and independent GPs, and most of the time, independent GPs or deep GPs perform worse than using mean of the test value.



(a) Exchange rate data set (b) EEG data set

Fig. 4.13 SMSE over missing values of independent GPs (igp), GPAR in forward direction (gpar_f), GPAR in backward direction (gpar_b), DGPs (dgp), GPLAR (gplar), and bi-directional GPLAR (gplar_bi). The comparison is made separately on different outputs, such that "'FZ,F1,F2' and "USD/MXN,USD/NZD" are **closed-upwards**, "F5,F6" and "USD/JPY, USD/AUD" are **closed-downwards**, and "F3,F4" are **closed-between**. The total column denotes averaged performance over all kinds of outputs.

4.4 GPLAR on Heterogeneous Outputs

4.4.1 Overview & Synthetic data

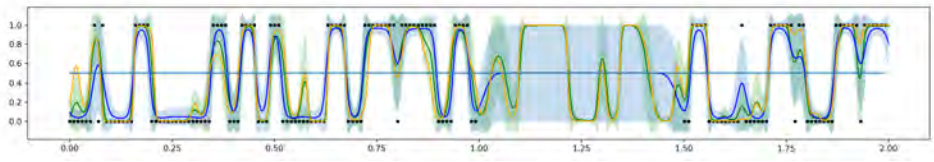
So far, we have only focus on the regression case where likelihoods are restricted to be Gaussian. However, GPLAR can be extended to non-Gaussian likelihoods and even heterogeneous outputs where a combination of continuous, categorical, binary, or discrete likelihood functions is presented. Since the likelihood expectation term in Eq .2.2 only requires the variational marginals, the final log-likelihood term can be computed analytically using quadrature (Hensman, A. Matthews, et al., 2015) or approximated by Monte Carlo sampling (Gal et al., 2015). In all following experiments, we first initialize the inducing points value from GPAR posterior predictive mean where binary data is treated as continuous outputs from 0 to 1, and categorical data is first one-hot encoded and then treated as continuous outputs likewise in the binary case. We show that GPLAR is able to correct any bad behaviors of prediction caused by this classification-regression conversion, and provide more informative results than independent GPs.

We first draw data from 4 synthetic GPs similarly in section. 4.2.2, where relations with previous outputs are made explicit by linear or non-linear kernels. In this experiment, the last two outputs are converted to binary outputs by first transforming samples of evaluations of the latent process to valid probability values using the sigmoid function, i.e. logistic probability, and then labels are generated from a Bernoulli distribution. The process of drawing the third output is shown as follows,

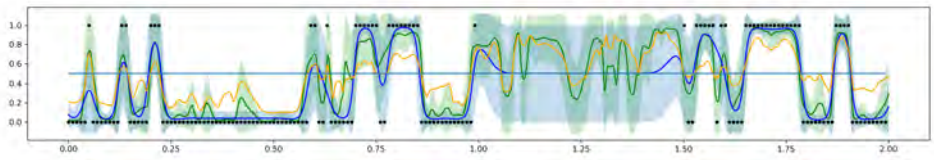
$$
\begin{aligned}
p(f_3|\theta_3) &= \mathcal{GP}(f_3; \mathbf{0}, k(\mathbf{x},\mathbf{x}') + k(h_{1:2}(\mathbf{x}), h_{1:2}(\mathbf{x}'))) \\
p(\mathbf{h}_3|f_3, \mathbf{X}, \mathbf{h}_{1:2}, \sigma^2) &= \prod_n \mathcal{N}(h_{3,n}; f_3(\mathbf{x}_n, h_{1:2,n}), \sigma_3^2) \\
p(y_{3n} = 1|h_{3n}) &= \sigma(h_{3n}) \text{ where, } \sigma(x) = 1/(1+\exp(x))
\end{aligned}
$$

The training inputs are uniformly drawn ranging from $[0.0, 2.0]$, with $N = 200$. $N_{missing} = 50$ observations for the last binary output are deliberately removed from interval $[0.5, 1.0]$. The remaining points are fed to the GPLAR model and independent GPs. The results are shown in Fig. 4.14. It is observed that information learned from previous tasks (either continuous or binary) has helped predictions at the last labeling task in GPLAR. The predictive mean nearly recovered the true underlying process, and the uncertainty is greatly reduced. As expected, the independent GPs failed to capture the dependencies and prior distribution was

given in the missing area. Moreover, it is obvious that independent GP is only fitting to the current binary output, while GPLAR would sense the change of latent process although there is no change in the labeling sequence. For example, labels in the interval $[0.55, 0.60]$ in Fig. 4.14a are all zeros, and independent GP gives consistent extreme predictive mean being over-confident about the output. While the true underlying probability of being labeled as 0 is actually not close to 0. GPLAR successfully learnt from previous tasks and reflect this uncertainty, which pattern is also shown elsewhere when the true latent process has a probability close to 0.5. The same conclusion can be drawn from the nonlinear case that GPLAR recovers dependencies between outputs even when the outputs are a mixture of binary and continuous values. It is also noticed that when the true dependency is drawn from nonlinear kernels, a GPLAR with only linear kernels between outputs would fail to capture the correlations.



(a) Linear kernel between outputs



(b) Nonlinear kernel between outputs

Fig. 4.14 GPLAR vs IGP on synthetic heterogeneous data. GPLAR predictions are green, Independent GPs predictions are blue, and the true underlying process values are orange. Observation points are denoted as black dots.

4.4.2 Real-data: Multi-label classification

In the multi-label classification framework, every single instance over the same input space has multiple labeling tasks. GPLAR can be used to achieve transfer learning between the multi-label tasks, and we used a real-data set to validate performance of our model against the simple independent GPs. The data is collected from 29 various landmine fields, and each point is consist of a 9-dimensional input feature vector extracted from radar images

taken at that field and a binary label indicating whether it is a mine or not[3]. Unlike previous experiments, where input is isotropic and 1-dimensional, this dataset has multi-dimensional input and is heterotopic such that each task has a different set of inputs. The original sensing problem aims to find landmines with minimum false alarms, we take $1-10$ tasks, all collected at foliated regions and hence share similar patterns of radar images.

For independent GPs and GPLAR, a randomly selected subset of data of various sizes is used as training data, while the remaining is used for test. The area-under-curve (AUC), which equals to the probability a randomly chosen positive instance is ranked higher than a randomly chosen negative instance, is used as performance metrics. We run 100 and 50 independent trials for independent GPs and GPLAR respectively, and the trend of AUC averaged over 10 tasks as the number of training data increases is shown in Fig. 4.15. It is observed that both independent GPs and GPLAR are sensitive to the amount of training data, and both have a large variance when the number of training data is low. However, on average GPLAR significantly outperforms independent GPs and has a smaller variance compared to independent GPs as the number of observations increases. The superiority of GPLAR suggests that kernels between outputs have great benefits in predictions of all tasks.

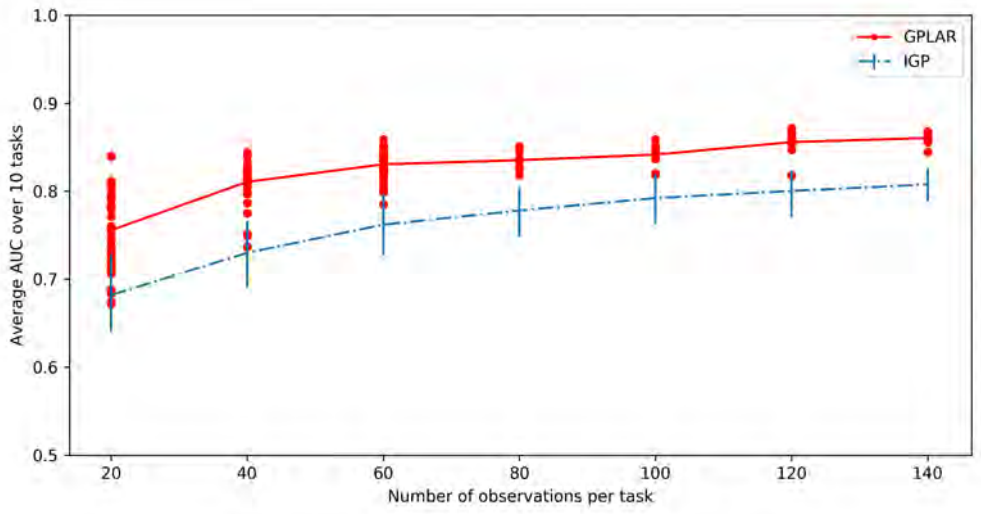


Fig. 4.15 Averaged AUC over 10 tasks as a function of number of training data between Independent GPs and GPLAR. Each curve connects mean of all trials, while the error bars of IGP curve represent standard deviation, and each scatter point in GPLAR curve represents each independent GPLAR trial.

[3]The Landmine data set is available at http://www.ee.duke.edu/~lcarin/LandmineData.zip.

4.4.3 Real-data: Heterogeneous Output

To compare with the large-scale experiments in Hensman, Fusi, et al. (2013) and Moreno-Muñoz et al. (2018), we test GPLAR on the complete records of house properties sold in the Greater London area in 2017[4]. Each record contains the postcode of the property and is transformed into a latitude-longitude 2-dimensional spatial point as input. We take two observations, one multi-class and one continuous. Unlike the experiments done in Moreno-Muñoz et al. (2018) where the first output only distinguish flat or non-flat properties (binary), the first observation in our case is multi-class indicating whether the property is flat, terraced, or semi-detached. The second output is the logarithm transformed sale price of the house. It is possible that multiple records exist with the same postcode and property type, for example, flats in one building, or properties sold multiple times in one year. Hence, prices of these records are averaged, making observations of each spatial point distinct. The complete datasets containing distinct records are shown in Fig. 4.16. A training set of randomly selected $20,000$ points is used with 200 inducing points, and the remaining $5,286$ are for test predictions.

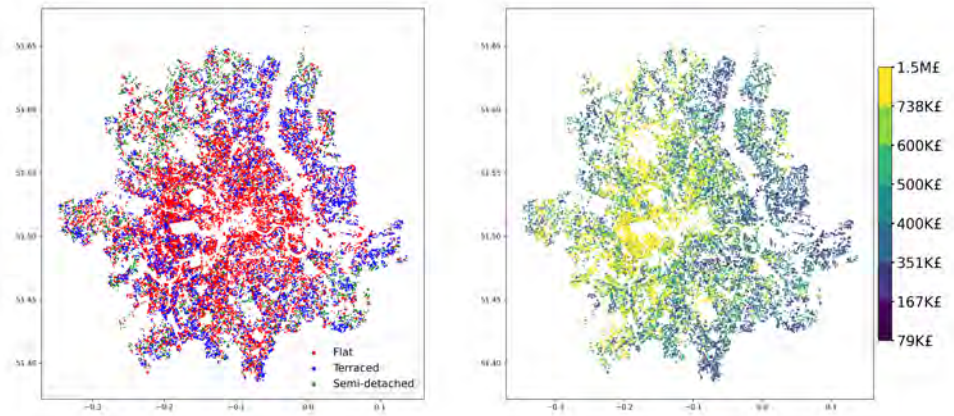


Fig. 4.16 London House Price dataset: property type (left) and sale price (right), presented on a longitude-latitude map.

As mentioned in section. 4.4.1, we initialize inducing points of GPLAR from the posterior predictive mean of GPAR acting as if the multi-class labels are a set of three continuous outputs ranging from 0 to 1. GPLAR then imposes a robust maximum likelihood with these

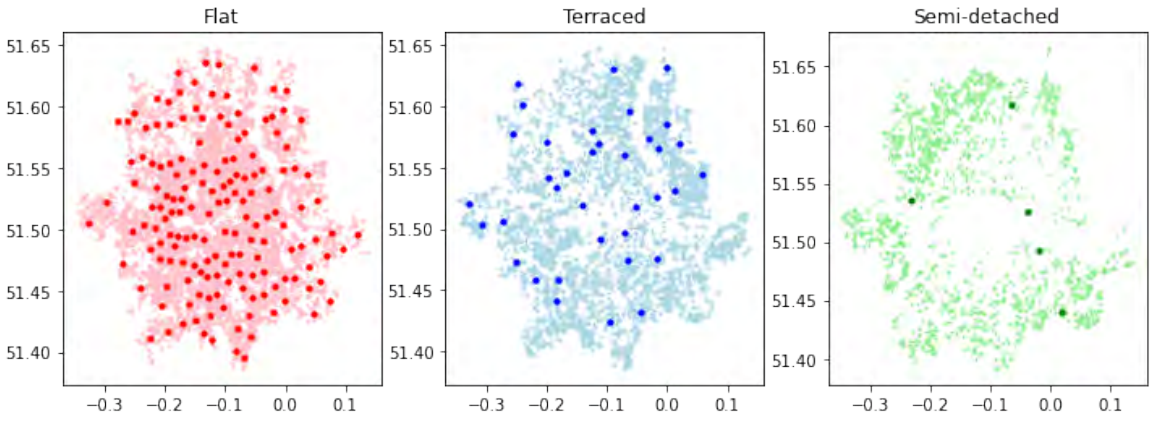
[4]The London House Price data is available at https://www.gov.uk/government/statistical-data-sets/price-paid-data-downloads.

	Input Kernel		Output Linear	Output Nonlinear	
Output	Variance	Lengthscales	Variance	Variance	Lengthscales
Flat	0.1884	[0.021, 0.026]			
Terraced	0.3489	[0.026, 0.030]	[4.388]	0.3186	[0.921]
Semi-detached	**0.0001**	**[1000., 1000.]**	[0.890 1.547]	0.0001	**[10000, 9714.]**

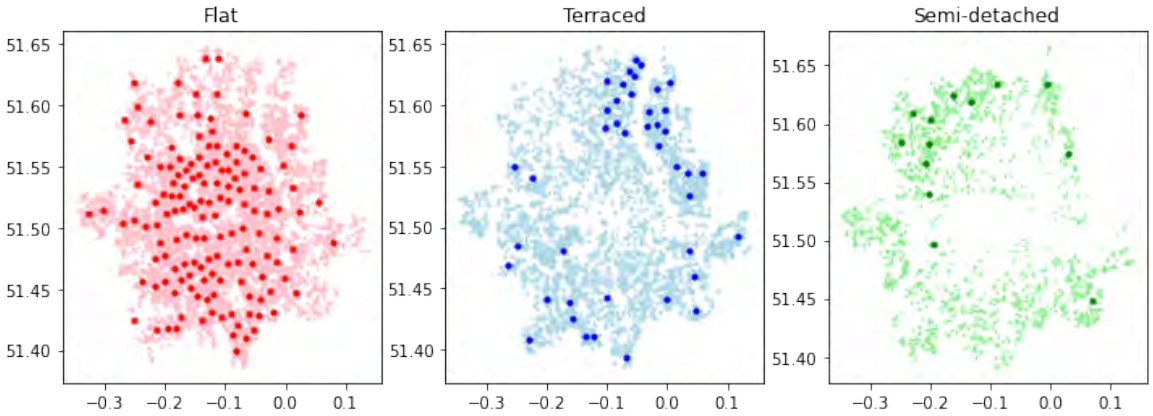
Table 4.4 Hyperparameter values of kernels learnt by GPAR on London House Price datasets

three latent processes. As shown in Fig. 4.17, the inducing points after optimization has a better representation or summary of the overall observations (The meaning of color is explained in the captions of Fig. 4.17). For properties of type flat, more inducing points are located in the centre of London. For properties of type terraced, more inducing points are moved to the northeast, or spread out in the southern part. As for semi-detached properties, more inducing points are located in the northwest. All the inducing points after optimization gain a more reasonable spatial meaning reflecting the true distribution of houses. This suggests that our optimization strategy of inducing points has corrected the error brought by treatments of multi-class labels as continuous values in GPAR. An interesting and unexpected finding of GPAR is that when the first three latent processes are treated as continuous values from 0 to 1, GPAR still finds the particular relationship between the three outputs such that property type can only be one of them. If one looks at the hyperparameter values of kernels of the third output in Table. 4.4, variance of temporal kernel is pushed to zero and lengthscales along both longitude and latitude are pushed to large numbers. The same phenomenon can be observed with the nonlinear kernel between the third and first two outputs, indicating the third output is learnt to completely depend linearly on the first two outputs.

During test predictions, GPLAR would produce $S = 100$ samples for each test point. For each sample, we take the corresponding class with the maximum latent process value. Finally, the modal class over all samples will give the predicted class of that test point. Accuracy of multi-class property type, SMSE of the real-valued house sale price, and log-density of both outputs are presented in Table. 4.5. It is observed that GPLAR has better performance than independent GPs evaluated by all metrics, indicating improvement of performance after adding kernels between outputs in large-scale datasets and heterogeneous real datasets, such that property type of a house has information for predicting the sale price of the house, and vise versa. Fig. 4.18 shows that GPLAR has successfully recovered the distribution of type and sales-price and with well-calibrated uncertainty. For example, the middle area has a lighter color (indicating high uncertainty) compared to darker colors on the periphery. Because the



(a) GPAR



(b) GPLAR

Fig. 4.17 The x-axis is longitude and the y-axis is latitude. The inducing points locations are labeled as red dots if the inducing values of first three latent process has largest value corresponding to flat, as blue dots if it is terraced, and as green dots if it is semi-detached. Figure (a) shows inducing points in GPAR and Figure (b) shows the optimized inducing points learnt by GPLAR. Both figures have the lighter and smaller scatter points denoting true observations in the background.

type is more mixed-up in the centre part, while more separated away from centre as observed from Fig. 4.16. Similarly, the predicted price distribution also matches with true observations.

Output	Binary		Continuous	
	Accuracy	HLL	SMSE	HLL
IGP	0.6416	-2.655	0.6122	-0.8820
GPLAR	**0.6672**	**-2.444**	**0.5949**	**-0.8566**

Table 4.5 SMSE/accuracy and HLL for heterogeneous outputs: IGP vs GPLAR for the London House Price datasets

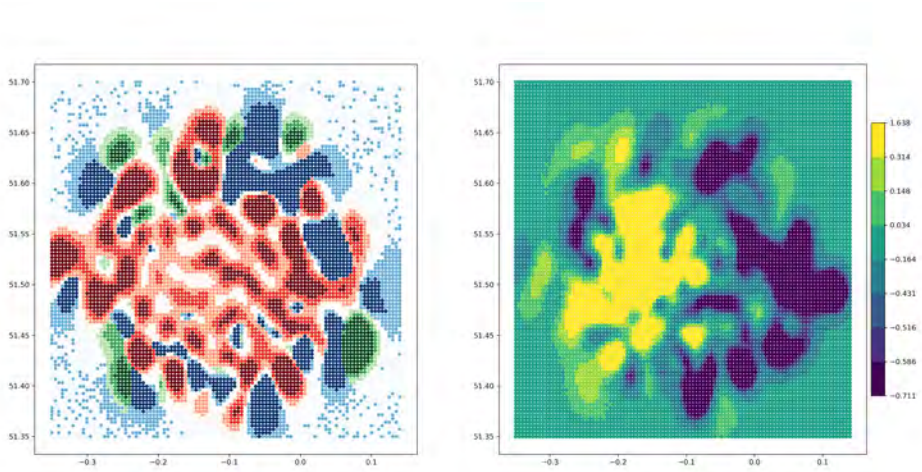


Fig. 4.18 GPLAR predictions of London House property type (left) and easy-normalized log-sale price (right) evaluated at 100×100 grid points over longitude-latitude space. The left figure shows the color corresponding to the latent process having the largest value (Flat, terraced or semi-detached), and the darkness of color denotes uncertainty, the darker the more certain.

Chapter 5

Conclusion

Our work mainly extended the Gaussian Process Autoregressive Regression model raised by Requeima et al. (2018) to deal with noisy outputs using a fully Bayesian approach, which also enables the resulting model to work with non-Gaussian or even heterogeneous likelihoods.

We utilized the inference scheme applied to the DGPs, where variational distributions are introduced over the non-linear GP mappings to deal with the intractability raised after hidden variables are made explicit in the new GPLAR model. GPLAR with efficient sparse approximation is proved to be more robust to noisy observations than GPAR with denoising effect, when the latter can be unstable and give under-fitting or over-fitting outcomes. A significant difference with DGPs in the inference scheme which requires careful treatment is the initialization and optimization of inducing points and locations. We made optimization of inducing points and locations possible and effective which are assumed to have the ability to correct the mistakes resulted from the posterior predictive mean of GPAR. This ability is further validated by experiments over synthetic data and real datasets.

We further extended the GPLAR model to bi-directional version, since the original settings of GPLAR would largely depend on the ordering of outputs which restricts the modeling power. It is realized that bi-directional GPLAR has better predictive mean, since information that flows in both directions empowers asymmetric correlations between outputs and makes learning easier than single ordering where updates of variational parameters in the first few channels would be hard when outputs dimension becomes larger and chain of GPs becomes longer. Experiments on real datasets verified the superiority of bi-directional GPLAR on insensitivity to locations of missing values, either in preceding or later part of the outputs. Real datasets with heterogeneous outputs containing both continuous and multi-class type are run with GPLAR that showed prominent improvement over independent GPs

indicating GPLAR's ability of knowledge transfer between regression and classification tasks.

Apart from the novel GPLAR model, we also made a comprehensive overview of the current literature over multi-output Gaussian Process models. Differences lie in how latent processes are being shared, either implicitly combined using a matrix, wrapped in a hierarchical way, or explicitly using outputs as inputs in an autoregressive way. The richness of autoregressive models in specifying any combination of linear or nonlinear kernels between outputs or between outputs and inputs makes them more powerful than linear coregionalization model or DGPs.

5.1 Limitations

Although the results have shown great success of GPLAR and its alternatives in predictions and inference, especially when the observations have large noises or when there are missing values, it still has some drawbacks and limitations that require careful human intervenes.

Sensitivity of initialization of hyperparameters As it is long realized, methods that use local stochastic optimization steps on non-convex objective functions can easily be trapped in local optimum. Apart from sensitivity to initialization of inducing locations which is solved by using k-means clustering when input dimension is high or initialized from the posterior predictive mean of GPAR for preceding channels, GPLAR is particularly sensitive to hyperparameter initial values and design of kernels. For example, the EEG datasets are much smoother than exchange rate datasets, and GPLAR would perform better if squared exponential kernel and rational quadratic kernel are used over the temporal space for these two tasks respectively. Furthermore, the landmine datasets have extremely imbalanced labels such that positive instances are far fewer than the negative ones, and hence requiring a careful selection of values of lengthscales and variances. Both choices of hyperparameters' initial value and design of kernels would need expert knowledge which is prohibitive when one wishes to build models that easily generalize to all conditions without much human intervention.

Balancing contribution of input and other outputs As mentioned before in section. 4.3.2, by specifying both kernels over inputs and over outputs, autoregressive model would need to balance the contributions. If the mapping can already achieve decent results by only utilizing the input information, model would struggle when the test points are far away from the

training input space and observations of other output would have played a crucial role that requires dependencies between outputs already discovered. This balance is also sensitive to initialization of hyperparameters such that if variances of kernels between outputs are much smaller than variances of input kernels, GPLAR would start like independent GPs and be trapped inside such local optimum.

Large training time as output dimension increases Although the model can scale-up with large-scale datasets, the training time still grows as the dimension of outputs increases. Since the GP modelling of later outputs depend on outputs of previous GPs, the training process is sequential which cannot be paralyzed or distributed to utilize the efficient calculations brought by GPU.

5.2 Future work

Additive GPLAR One of the alternatives of GPLAR discussed in section. 3.3, additive GPLAR, is designed to deal with poorly calibrated uncertainty of GPLAR, however, the number of latent processes and their corresponding number of variational parameters increase significantly as the number of outputs increases and hence requires more careful treatments. The performance over real datasets also did not show its advantages over original GPLAR or bi-directional GPLAR. However, its explicit separation of hidden variables should provide better interpretation ability, and more flexibility should be achieved by not internally connecting the inducing points over input. Further work can be done to explore the additive GPLAR and improve its inference scheme.

GPLAR with Bayesian Network GPLAR can be extended to work with the Bayesian network, where uncertainty and structures of correlations between outputs and inputs can also be modeled instead of directly using fully connected graphs. Gaussian Process Networks raised by Friedman et al. (2013) only puts a Gaussian Process prior over the network and allows for structural learning. It can be extended to combine with GPLAR and then kernels between some unrelated outputs can be relaxed. For example, the landmine detection tasks in section. 4.4.2 contains labels collected from foliated regions and regions that are bare earth or desert. If networks are learnt to discover the distinction between the two regions which share no common knowledge, outputs of foliated regions would not be fed to regions on desert and as a result, benefits the modeling.

References

[1] Edwin V Bonilla et al. "Multi-task Gaussian process prediction". In: *Advances in neural information processing systems*. 2008, pp. 153–160.

[2] Hanen Borchani et al. "A survey on multi-output regression". In: *Wiley Interdisciplinary Reviews: Data Mining and Knowledge Discovery* 5.5 (2015), pp. 216–233.

[3] Thang Duc Bui. "Efficient Deterministic Approximate Bayesian Inference for Gaussian Process models". PhD thesis. University of Cambridge, 2018.

[4] Thang D Bui et al. "Tree-structured Gaussian process approximations". In: *Advances in Neural Information Processing Systems*. 2014, pp. 2213–2221.

[5] Thang Bui et al. "Deep Gaussian processes for regression using approximate expectation propagation". In: *International conference on machine learning*. 2016, pp. 1472–1481.

[6] Andrew Carlson et al. "Coupled semi-supervised learning for information extraction". In: *Proceedings of the third ACM international conference on Web search and data mining*. 2010, pp. 101–110.

[7] Jean-Paul Chiles et al. *Geostatistics: modeling spatial uncertainty*. Vol. 497. John Wiley & Sons, 2009.

[8] Kurt Cutajar et al. "Deep gaussian processes for multi-fidelity modeling". In: *arXiv preprint arXiv:1903.07320* (2019).

[9] Andreas Damianou et al. "Deep gaussian processes". In: *Artificial Intelligence and Statistics*. 2013, pp. 207–215.

[10] Alexander G De G. Matthews et al. "GPflow: A Gaussian process library using TensorFlow". In: *The Journal of Machine Learning Research* 18.1 (2017), pp. 1299–1304.

[11] David K Duvenaud et al. "Additive gaussian processes". In: *Advances in neural information processing systems*. 2011, pp. 226–234.

[12] David Duvenaud, James Lloyd, et al. "Structure discovery in nonparametric regression through compositional kernel search". In: *International Conference on Machine Learning*. 2013, pp. 1166–1174.

[13] David Duvenaud, Oren Rippel, et al. "Avoiding pathologies in very deep networks". In: *Artificial Intelligence and Statistics*. 2014, pp. 202–210.

[14] Nir Friedman et al. "Gaussian process networks". In: *arXiv preprint arXiv:1301.3857* (2013).

[15] Yarin Gal et al. "Latent Gaussian processes for distribution estimation of multivariate categorical data". In: (2015).

[16] Alexandros Gryparis et al. “Semiparametric latent variable regression models for spatiotemporal modelling of mobile source particles in the greater Boston area”. In: *Journal of the Royal Statistical Society: Series C (Applied Statistics)* 56.2 (2007), pp. 183–209.

[17] James Hensman, Nicolo Fusi, et al. “Gaussian processes for big data”. In: *arXiv preprint arXiv:1309.6835* (2013).

[18] James Hensman, Alexander Matthews, et al. “Scalable variational Gaussian process classification”. In: (2015).

[19] Matthew D Hoffman et al. “Stochastic variational inference”. In: *The Journal of Machine Learning Research* 14.1 (2013), pp. 1303–1347.

[20] Diederik P Kingma and Jimmy Ba. “Adam: A method for stochastic optimization”. In: *arXiv preprint arXiv:1412.6980* (2014).

[21] Diederik P Kingma and Max Welling. “Auto-encoding variational bayes”. In: *arXiv preprint arXiv:1312.6114* (2013).

[22] Haitao Liu et al. “Remarks on multi-output Gaussian process regression”. In: *Knowledge-Based Systems* 144 (2018), pp. 102–121.

[23] Chao Ma et al. “HM-VAEs: a Deep Generative Model for Real-valued Data with Heterogeneous Marginals”. In: *Symposium on Advances in Approximate Bayesian Inference*. 2020, pp. 1–8.

[24] Alexander G de G Matthews et al. “On sparse variational methods and the Kullback-Leibler divergence between stochastic processes”. In: *Journal of Machine Learning Research* 51 (2016), pp. 231–239.

[25] Pablo Moreno-Muñoz et al. “Heterogeneous multi-output gaussian process prediction”. In: *Advances in neural information processing systems*. 2018, pp. 6711–6720.

[26] Trung V Nguyen et al. “Collaborative Multi-output Gaussian Processes.” In: *UAI*. 2014, pp. 643–652.

[27] Paris Perdikaris et al. “Nonlinear information fusion algorithms for data-efficient multi-fidelity modelling”. In: *Proceedings of the Royal Society A: Mathematical, Physical and Engineering Sciences* 473.2198 (2017), p. 20160751.

[28] Carl Edward Rasmussen. “Gaussian processes in machine learning”. In: *Summer School on Machine Learning*. Springer. 2003, pp. 63–71.

[29] Jesse Read et al. “Classifier chains for multi-label classification”. In: *Machine learning* 85.3 (2011), p. 333.

[30] James Requeima et al. “The gaussian process autoregressive regression model (gpar)”. In: *arXiv preprint arXiv:1802.07182* (2018).

[31] Eric Turner Richard et al. “Two problems with variational expectation maximisation for time-series models”. In: *Bayesian time series models*. Ed. by David Barber et al. Cambridge University Press, 2011. Chap. 5, pp. 109–130.

[32] Hugh Salimbeni et al. “Doubly stochastic variational inference for deep Gaussian processes”. In: *Advances in Neural Information Processing Systems*. 2017, pp. 4588–4599.

[33] Mike Schuster et al. “Bidirectional recurrent neural networks”. In: *IEEE transactions on Signal Processing* 45.11 (1997), pp. 2673–2681.

[34] Grigorios Skolidis et al. “Bayesian multitask classification with Gaussian process priors”. In: *IEEE Transactions on Neural Networks* 22.12 (2011).

[35] Michalis Titsias. “Variational learning of inducing variables in sparse Gaussian processes”. In: *Artificial Intelligence and Statistics*. 2009, pp. 567–574.

[36] Jarno Vanhatalo et al. “Additive multivariate Gaussian processes for joint species distribution modeling with heterogeneous data”. In: *Bayesian analysis* (2018).

[37] Hans Wackernagel. *Multivariate geostatistics: an introduction with applications*. Springer Science & Business Media, 2013.

[38] Mark van der Wilk et al. “A Framework for Interdomain and Multioutput Gaussian Processes”. In: *arXiv preprint arXiv:2003.01115* (2020).

[39] David H Wolpert. “Stacked generalization”. In: *Neural networks* 5.2 (1992), pp. 241–259.

[40] Kai Yu et al. “Learning Gaussian processes from multiple tasks”. In: *Proceedings of the 22nd international conference on Machine learning*. 2005, pp. 1012–1019.

[41] Min-Ling Zhang et al. “A review on multi-label learning algorithms”. In: *IEEE transactions on knowledge and data engineering* 26.8 (2013), pp. 1819–1837.

[42] Han Zhu et al. “Multi-view Deep Gaussian Process with a Pre-training Acceleration Technique”. In: *Pacific-Asia Conference on Knowledge Discovery and Data Mining*. Springer. 2020, pp. 299–311.