UNIVERSIDADE FEDERAL DO CEARÁ
CENTRO DE CIÊNCIAS
DEPARTAMENTO DE COMPUTAÇÃO
PROGRAMA DE PÓS-GRADUAÇÃO EM CIÊNCIA DA COMPUTAÇÃO

DANIEL AUGUSTO RAMOS MACEDO ANTUNES DE SOUZA

CONTRIBUTIONS ON LATENT PROJECTIONS FOR GAUSSIAN PROCESS MODELING

FORTALEZA
2020

DANIEL AUGUSTO RAMOS MACEDO ANTUNES DE SOUZA

CONTRIBUTIONS ON LATENT PROJECTIONS FOR GAUSSIAN PROCESS MODELING

Dissertação apresentada ao Programa de Pós-Graduação em Ciência da Computação do Centro de Ciências da Universidade Federal do Ceará, como requisito parcial à obtenção do título de mestre em Ciência da Computação. Área de Concentração: Aprendizado de Máquina

Orientador: Prof. Dr. João Paulo Pordeus Gomes

Coorientador: Prof. Dr. César Lincoln Cavalcante Mattos

FORTALEZA

2020

Dados Internacionais de Catalogação na Publicação
Universidade Federal do Ceará
Biblioteca Universitária
Gerada automaticamente pelo módulo Catalog, mediante os dados fornecidos pelo(a) autor(a)

S237c Souza, Daniel Augusto Ramos Macedo Antunes de.
Contributions on latent projections for Gaussian process modeling / Daniel Augusto Ramos Macedo Antunes de Souza. – 2020.
72 f. : il. color.

Dissertação (mestrado) – Universidade Federal do Ceará, Centro de Ciências, Programa de Pós-Graduação em Ciência da Computação, Fortaleza, 2020.
Orientação: Prof. Dr. João Paulo Pordeus Gomes.
Coorientação: Prof. Dr. César Lincoln Cavalcante Mattos.

1. Aprendizado de máquina. 2. Processos Gaussianos. 3. Inferência variacional. 4. Deep learning. 5. Aprendizado de variedades. I. Título.

CDD 005

CONTRIBUTIONS ON LATENT PROJECTIONS FOR GAUSSIAN PROCESS MODELING

Dissertação apresentada ao Programa de Pós-Graduação em Ciência da Computação do Centro de Ciências da Universidade Federal do Ceará, como requisito parcial à obtenção do título de mestre em Ciência da Computação. Área de Concentração: Aprendizado de Máquina

Aprovada em:

BANCA EXAMINADORA

Ao Zyg, o cachorro filósofo.

AGRADECIMENTOS

Primeiramente, eu agradeço aos meus orientadores, Prof. João Paulo Pordeus Gomes e Prof. César Lincoln Cavalcante Mattos, pelos conselhos, instruções e a co-autoria de diversos textos durante este mestrado. Eu extendo esse agradecimento para todos aqueles que também me instruíram até aqui, seja no grupo de pesquisa LOGIA, seja nas salas de aulas, ou em particular. Apesar do caminho intrincado, essa dissertação é a culminação do que eu aprendi na Universidade Federal do Ceará.

Sem dúvida, eu não posso dizer o quanto sou agradecido aos meus pais, Jacqueline e Sandoval, por terem me ajudado desde sempre para chegar onde estou e terem me aguentado durante a escrita desta dissertação. Principalmente agora, quando minha mãe passava pelo fim do doutorado.

Eu agradeço a ajuda e o apoio dos meus amigos na escrita deste texto pois, infelizmente, muitas das minhas horas e dos meus pensamentos foram ocupados na escrita dele. Em especial, agradeço à Carolina Herbster por ter me ajudado no fraseamento das várias versões deste texto.

“Voici mon secret. Il est très simple: [...] L’essentiel est invisible pour les yeux.”

(Le Petit Prince by Antoine de Saint Exupéry)

RESUMO

Projetar dados num espaço latente é uma operação rotineira em aprendizado de máquina. Um dos incentivos para realizar tal transformação é a hipótese da variedade (*manifold hypothesis*), que diz que a maioria dos dados amostrados de um processo empírico tendem a estar dentro de um espaço de dimensão menor. Já que essa representação menor não é visível no conjunto de dados, técnicas probabilísticas de aprendizado de máquina conseguem propagar as incertezas nos dados para a representação latente de forma acurada. Em particular, processos Gaussianos (GP) são uma família de métodos de kernel probabilísticos que foram aplicados com sucesso em tarefas de regressão e redução de dimensão. Contudo, no caso da redução de dimensão, inferência variacional determinística e eficiente só existe para um conjunto mínimo de kernels. Portanto, eu proponho o *unscented Bayesian Gaussian process latent variable model* (UGPLVM), um método de inferência alternativo para o *Bayesian Gaussian process latent variable model* que usa a transformação *unscented* a fim de permitir o uso de kernels completamente arbitrários enquanto se mantem eficiente em amostragem. Para regressão com modelos GP, o *deep Gaussian process* (DGP) composicional é um modelo popular que utiliza transformações sucessivas entre espaços latentes para aliviar a dificuldade de escolher de um kernel. Contudo, essa não é a única construção possível para um DGP. Nessa dissertação, eu proponho outra construção para DGP onde cada camada controla a suavidade da próxima, ao invés de compor entradas com saídas diretamente. Esse modelo é chamado de *deep Mahalanobis Gaussian process* (DMGP), e ele é baseado em pesquisas anteriores sobre a integração de hiperparâmetros do kernel Mahalanobis e, então, incorpora a ideia de projeções localmente lineares. Ambas as propostas usam inferência variacional determinística mas ainda mantem os mesmos resultados e escabilidade que métodos não determinísticos em várias tarefas experimentais. Os experimentos para o UGPLVM cobrem tarefas de redução de dimensionalidade e simulação de sistemas dinâmicos com propagação de incerteza, e, para o DMGP, cobrem tarefas de regressão em conjuntos de dados sintéticos e empíricos.

Palavras-chave: Aprendizado de máquina. Processos Gaussianos. Inferência variacional. Deep learning. Aprendizado de variedades.

ABSTRACT

Projecting data to a latent space is a routine procedure in machine learning. One of the incentives to do such transformations is the manifold hypothesis, which states that most data sampled from empirical processes tend to be inside a lower-dimensional space. Since this smaller representation is not visible in the dataset, probabilistic machine learning techniques can accurately propagate uncertainties in the data to the latent representation. In particular, Gaussian processes (GP) are a family of probabilistic kernel methods that researchers have successfully applied to regression and dimensionality reduction tasks. However, for dimensionality reduction, efficient and deterministic variational inference exists only for a minimal set of kernels. As such, I propose the unscented Gaussian process latent variable model (UGPLVM), an alternative inference method for Bayesian Gaussian process latent variable models that uses the unscented transformation to permit the use of arbitrary kernels while remaining sample efficient. For regression with GP models, the compositional deep Gaussian process (DGP) is a popular model that uses successive mappings to latent spaces to alleviate the burden of choosing a kernel function. However, that is not the only DGP construction possible. In this dissertation, I propose another DGP construction in which each layer controls the smoothness of the next layer, instead of directly composing layer outputs into layer inputs. This model is called deep Mahalanobis Gaussian process (DMGP), and it is based on previous literature on the integration of Mahalanobis kernel hyperparameters and, thus, incorporates the idea of locally linear projections. Both proposals use deterministic variational inference while maintaining the same results and scalability as non-deterministic methods in various experimental tasks. The experiments for UGPLVM cover dimensionality reduction and simulation of dynamic systems with uncertainty propagation, and, for DMGP, they cover regression tasks with synthetic and empirical datasets.

Keywords: Machine learning. Gaussian processes. Variational inference. Deep learning. Manifold learning.

LIST OF FIGURES

LIST OF TABLES

LIST OF ABBREVIATIONS AND ACRONYMS

1-NN	1-nearest neighbors algorithm
ARD	automatic relevance determination
DGP	deep Gaussian process
DMGP	deep Mahalanobis Gaussian process
DNN	deep neural network
DS-DGP	doubly stochastic deep G aussian process
DSVI	doubly stochastic variational inference
DVMGP	deep variational Mahalanobis Gaussian process
ELBO	evidence lower bound
GH	Gauss-Hermite
GP	Gaussian process
GP-NARX	Gaussian process nonlinear autoregressive exogenous model
GPLVM	Gaussian process latent variable model
GPR	Gaussian process for regression
L-BFGS-B	limited-memory BFGS algorithm with bound constraints
MAP	maximum a posteriori probability
MC	Monte Carlo
MCMC	Markov chain Monte Carlo
MLE	maximum likelihood estimation
MLP	multilayer perceptron
MRAE	mean relative absolute error
NARX	nonlinear autoregressive exogenous model
NLPD	negative log predictive density
PCA	principal component analysis
RBF	radial basis function
RMSE	root mean squared error
SGP	sparse Gaussian process
SGPR	sparse Gaussian process for regression
SRBF	standard RBF kernel
USPS	United States postal service
UT	unscented transformation

VDMGP	variational inference for Mahalanobis distance metric Gaussian process

LIST OF SYMBOLS

$a, b, \ldots$	Scalar values
$\boldsymbol{a}, \boldsymbol{b}, \ldots$	Column vector values
$\boldsymbol{A}, \boldsymbol{B}, \ldots$	Matrix or tensor values
$p(\cdot)$	Probability density function of a random variable
$p(\cdot \mid \cdot)$	Probability density function of a random variable given the value of another
$q(\cdot)$	Variational probability density function of a random variable
$\langle f(x) \rangle_{p(x)}$	Expected value of a function under a random variable
$\mu, \boldsymbol{\mu}, \boldsymbol{M}$	Mean parameter of a random variable
$\sigma^2, \boldsymbol{\sigma}^2, \boldsymbol{\Sigma}$	Variance parameter of a random variable
$\mathscr{N}\left(\cdot \mid \mu, \sigma^2\right)$	Probability density function of a random variable that follow a Gaussian distribution
$\boldsymbol{a}^\top, \boldsymbol{A}^\top$	Transpose of a vector, matrix or tensor
$\mathbb{R}$	Real numbers
$\boldsymbol{X}$	Training input data
$\boldsymbol{Y}$	Training output data
$\boldsymbol{x}_*$	Test input data
$\boldsymbol{y}_*$	Test output data
N	Number of samples in the training dataset
D_x	Dimension of the input dataset
D_y	Dimension of the output dataset
Q	Dimension of a latent space
k	Kernel function
$\boldsymbol{\xi}, \boldsymbol{\Xi}$	First layer pseudo-inputs
$\mathfrak{z}, \mathfrak{Z}$	Last layer pseudo-inputs
$\check{\boldsymbol{\mu}}, \check{\boldsymbol{M}}$	Variational mean parameter
$\check{\boldsymbol{\Sigma}}$	Variational covariance parameter
$\psi_0, \boldsymbol{\Psi}_1, \boldsymbol{\Psi}_2$	Psi-statistics

$\boldsymbol{s}, \boldsymbol{S}$	Unscented transformation sigma points
H	Gauss-Hermite quadrature grid size
$\ln(\cdot)$	Natural logarithm function
e	Euler’s constant

CONTENTS

1 INTRODUCTION

Projecting data to a latent space is a routine procedure in machine learning, either to higher dimensional or lower dimensional spaces. One of the incentives to do such transformations is the manifold hypothesis. This hypothesis states that most data sampled from empirical processes tend to be inside a lower-dimensional manifold in the sample space (FEFFERMAN *et al.*, 2016). For example, it is unreasonable to assume that all pixels of a digital photograph are independent of each other and, because of their correlations, it is safe to assume that the pixels of an image has too many degrees of freedom compared to the scene that it represents.

Consequently, by projecting the data into a lower dimensional manifold, the degrees of freedom are reduced. This type of reduction is useful because extra degrees of freedom may present non-informative noise or allow learning algorithms to find spurious correlations in data. (CAYTON, 2005)

Because this smaller representation is not visible in the dataset, it must exist in a latent space. Through Bayes' theorem, probabilistic machine learning techniques can directly model characteristics of the unobservable representation and, more importantly, correctly propagate uncertainties in data from and to the latent representation. However, to perform exact inference, in other words, to fit a probabilistic model with observed data can be either too costly or impossible. Nevertheless, expressive probabilistic models with exact inference for regression problems do exist.

In particular, Gaussian processes are a family of probabilistic methods that have been applied to diverse tasks in machine learning (RASMUSSEN; WILLIAMS, 2006). In these models, the prior distribution of the output data is represented as a Gaussian distribution in which a user-chosen kernel computes the covariance between outputs based on the input points. Solutions to regression tasks are obtained by conditioning unobserved output values by the observed output and input values. In this setup, the model inference is exact and any hyperparameters of the kernel are optimized through type-II maximum likelihood estimation (MLE).

Through the assumption that the input data is latent and follows a Gaussian distribution, Lawrence (2004) derived the Gaussian process latent variable model (GPLVM). This model can be used for dimensionality reduction tasks, as previously mentioned, by computing the conditional distribution of the input data given the output data. However, efficient and deterministic approximate inference is only available for a limited set of kernel functions. This drawback limits the possible projections from observed space to latent space, since the kernel function governs the complexity of their relation.

For either task, a lot of the current advances in machine learning have come from moving away from expert-designed feature extractors to learning how to learn the feature space from data without human engineering (LECUN *et al.*, 2015). Because kernels define implicit feature spaces, the same argument applies to Gaussian process (GP) models. Following this trend, there have been proposals to alleviate the decision to choose a kernel. Two major approaches exist, the use of deep neural networks as kernels (WILSON *et al.*, 2016a) or the construction of hierarchies of Gaussian processes with simple kernels (DUNLOP *et al.*, 2018). The first framework is called deep kernel learning, and the second one is called deep Gaussian processes.

Deep kernel learning presents exact inference, but the increased number of parameters optimized by MLE decreases the benefits that Bayesian learning provides when compared to traditional methods. On the other hand, while remaining Bayesian, inference on deep Gaussian process (DGP) models has to be approximate.

The mainstream approach to DGP consists of composing one process' outputs into the input of another, similar to a deep neural network (DAMIANOU; LAWRENCE, 2013). However, without proper care, these models exhibit a pathological behavior where the latent space of each layer becomes excessively compressed compared to the ones that came before (DUVENAUD *et al.*, 2014). Consequently, recent models based on composition, such as the one proposed by Salimbeni and Deisenroth (2017a), sidestep this issue by adding a linear component to the a priori mean.

Nevertheless, as surveyed by Dunlop *et al.* (2018), this is not the only possible approach to build hierarchical GP models. In particular, it is possible to control the lengthscales of a GP's stationary kernel function through another GP. This construction is referred to as composition through kernel function. An advantage of these models is that their layers could have a direct interpretation when compared to the layers of compositional DGPs.

This class of DGP models builds upon the work by Gibbs (1997) and Paciorek (2003) that transforms any stationary kernel into a kernel with input-varying lengthscales. Despite the statement from Dunlop *et al.* (2018) that variational inference methods for this type of construction have not been explored yet, a workshop article by Salimbeni and Deisenroth (2017b) explores a stochastic variational inference method for these models.

However, as noted by Gibbs (1997), the input-varying lengthscales of these models do not inherit the semantics associated with the original stationary kernel. This issue leads to counter-intuitive behavior and hinders the interpretability of such models, thus, reducing its appeal. Not only that, but the lengthscales in conventional stationary kernels also define a projection function of the input data into a lower-dimensional manifold. However, as shown by Paciorek (2003), this is not the case with this class of DGP models.

As a result, there exists a gap for DGP models built from the composition of kernel lengthscales that can maintain at least one of the expected properties of ordinary stationary kernels and can be handled with deterministic variational inference.

1.1 Objectives

This dissertation's objective is to enable greater expressivity and control of the latent dimensions in GP models. Specifically, I plan to explore these deficiencies in deterministic variational inference for flexible GPLVM and DGPs by composition through kernel function. These objectives are listed as follows:

- To offer a different inference method for GPLVM, that is approximate but allows for greater flexibility while achieving comparable results.
- To test this inference method in a dimensionality reduction setting and a dynamic system simulation setting, using classification and regression metrics.
- To present alternative DGP models that compose by kernel function that allows for deterministic variational inference. This alternative should not be affected by the typical pathological behavior of naive DGP while maintaining the interpretation that lengthscales define a projection of the input data and achieving competitive results in experimental tests.
- To evaluate this new model empirically through the analysis of the prior samples from very deep models and the results of applying this model in standard regression datasets.

1.2 Dissertation structure overview

This dissertation is divided into five chapters, including this introduction.

Chapter 2 has a brief overview of the common theoretical background of this text, starting with common notation and basic definitions of Gaussian processes for regression and dimensionality reduction. Chapters 3 and 4 contain the two contributions of this dissertation. Finally, Chapter 5 wraps together all the conclusions of the previous chapters, discusses them in light of the proposed objectives, and analyzes the possibilities of extensions of these proposals and possible works for the future.

More specifically, Chapter 3 presents the unscented Gaussian process latent variable model. This model is an extension of the Bayesian GPLVM that allows for more flexibility in the function that relates the latent variable to the observed data while maintaining scalable inference compared to other methods.

And then, in Chapter 4, I present the deep Mahalanobis distance Gaussian process model. It is based on the methodology proposed by Titsias and Lázaro-Gredilla (2013) but extends it to allow the modeling of functions with varying smoothness. This extension has equivalent expressivity to DGP models but is built upon locally linear projections.

2 COMMON BACKGROUND AND DEFINITIONS

This chapter introduces the common theoretical background and definitions that will be used throughout the dissertation. These refer to well-established results in Gaussian processes and their approximations that are used in both proposed methodologies. However, more specific references and results are contained in their respective chapters as well.

This dissertation is mainly concerned with two traditional machine learning tasks: supervised learning and dimensionality reduction. Supervised learning is the problem of learning an input-output mapping from labeled empirical data. Dimensionality reduction is the problem of eliminating possibly redundant information from data while preserving desirable non-redundant properties from it.

Some mathematical means are needed to express these concepts and the many ways to solve them in a precise manner. In this chapter, I will briefly introduce the mathematical notation used in the dissertation and go through the theoretical background common to all chapters.

2.1 Notation and nomenclature

Every data point of a dataset has to be encoded by vectors of numbers to be handled by a computer. Values that are single numbers are called scalars, rows of numbers are called vectors, and grids of numbers are called matrices.

In this dissertation, all scalars are represented in formulas by lowercase letters, all row vectors by bold lowercase letters, and all matrices are column-major grids of scalars represented by bold uppercase letters. The following formula displays a N by D matrix $\boldsymbol{A}$:

$$\boldsymbol{A} = \begin{bmatrix} \boldsymbol{a}_0 \\ \boldsymbol{a}_1 \\ \vdots \\ \boldsymbol{a}_{(N-1)} \end{bmatrix} = \begin{bmatrix} a_{00} & a_{01} & \cdots & a_{0(D-1)} \\ a_{10} & a_{11} & \cdots & a_{1(D-1)} \\ \vdots & \vdots & \ddots & \vdots \\ a_{(N-1)0} & a_{(N-1)1} & \cdots & a_{(N-1)(D-1)} \end{bmatrix}.$$

When indexing vectors and matrices, the typeface used must match the type of the result, for example, the scalar a_{103} is the fourth element of the vector $\boldsymbol{a}_{10}$, which is the first line of the matrix $\boldsymbol{A}_1$. Matrices can also be organized into larger structures that are called tensors. They are also represented by bold upper case letters. Therefore, the matrix $\boldsymbol{A}_1$ is the second matrix of the tensor $\boldsymbol{A}$. Because tensors are not common entities in this dissertation, this ambiguity may not be a source of errors. Furthermore, throughout this dissertation, footnotes indicate the lower-case or upper-case versions of unfamiliar letters, along with a pronunciation guide.

To indicate that no indexing is done on a specific dimension, colons are used instead of variables or numbers. For example, if $\boldsymbol{A} \in \mathbb{R}^{4\times3\times2}$, then $\boldsymbol{a}_{0:0} \in \mathbb{R}^3$ is the first column of $\boldsymbol{A}_0 \in \mathbb{R}^{3\times2}$ as a row vector.

A labeled dataset $\mathscr{D}$ is composed of a matrix of inputs $\boldsymbol{X} \in \mathbb{R}^{N\times D_x}$ and a matrix of outputs $\boldsymbol{Y} \in \mathbb{R}^{N\times D_y}$. Each row $\boldsymbol{x}_i$ and $\boldsymbol{y}_i$ of $\boldsymbol{X}$ and $\boldsymbol{Y}$ represents a single data point. The dataset $\mathscr{D}$ is said to have N data points and dimension $D_x + D_y$. Then, a supervised learning problem is about learning a mapping $\boldsymbol{f} : \mathbb{R}^{D_x} \to \mathbb{R}^{D_y}$ such that for every i, $\boldsymbol{f}(\boldsymbol{x}_i)$ is as close to $\boldsymbol{y}_i$ as possible. When using the learned function in new unseen data points, only a matrix $\boldsymbol{X}_* \in \mathbb{R}^{n_*\times D_x}$ has to be supplied.

It is usual to assume that the data collected in a dataset is not error-free. This is due to the limited precision of sensor instruments, unexpected interference during measurement, or just the lack of explanatory variables. The correct way to mathematically deal with these circumstances is with the use of probability theory.

The probability density function is a function that assigns plausibility to every possible observation of a random variable. For a random variable $\boldsymbol{x}$, this function is represented as $p(\boldsymbol{x})$. This can be extended for multiple variables by extending the number of inputs of this function, e.g. $p(\boldsymbol{x},\boldsymbol{y},\ldots)$. It is also useful to compute the plausibility of a variable given that another one is known. This plausibility is represented as $p(\boldsymbol{y},\ldots \mid \boldsymbol{x},\ldots)$, where the given variables are listed after the bar.

Two of the most important values that can be derived from a random variable is its expected value and covariance. The expected value of a random variable is the weighted average of all values that the random variable can take. The covariance quantifies the linear relationship between the dimensions of the variable. Given a random variable x:

$$\langle \boldsymbol{x} \rangle_{p(\boldsymbol{x})} = \int_{\boldsymbol{x}} \boldsymbol{x} p(\boldsymbol{x}), \qquad \text{(Expected value)}$$

$$\mathbf{Cov}(\boldsymbol{x}) = \left\langle (\boldsymbol{x} - \langle \boldsymbol{x} \rangle_{p(\boldsymbol{x})})^\intercal (\boldsymbol{x} - \langle \boldsymbol{x} \rangle_{p(\boldsymbol{x})}) \right\rangle_{p(\boldsymbol{x})}. \qquad \text{(Covariance)}$$

Given an arbitrary function $\boldsymbol{f} \colon \mathbb{R}^a \to \mathbb{R}^b$ and a random variable $\boldsymbol{x}$, then $\boldsymbol{f}(\boldsymbol{x})$ is also a random variable. The expected value of this transformed variable can be computed using the probability density function of $\boldsymbol{x}$:

$$\langle \boldsymbol{f}(\boldsymbol{x}) \rangle_{p(\boldsymbol{x})} = \int_{\boldsymbol{x}} \boldsymbol{f}(\boldsymbol{x}) p(\boldsymbol{x}).$$

One of the most quintessential probability density functions and the most used in the dissertation is the Gaussian distribution. Random variables that follow this distribution are entirely defined by giving their expected value and covariance. For a given random variable $\boldsymbol{x}$ with observations in $\mathbb{R}^n$, it is expressed as:

$$\begin{aligned} p(\boldsymbol{x}) &= \mathscr{N}(\boldsymbol{x} \mid \boldsymbol{\mu}, \boldsymbol{\Sigma}) \\ &= 2^{\frac{n}{2}} |\boldsymbol{\Sigma}|^{\frac{1}{2}} \exp\left(-\frac{1}{2}\left[(\boldsymbol{x}-\boldsymbol{\mu})\boldsymbol{\Sigma}^{-1}(\boldsymbol{x}-\boldsymbol{\mu})^{\mathsf{T}}\right]\right). \end{aligned}$$

Two very important formulas for the Gaussian distribution are for conditioning and marginalization, assume:

$$p(\boldsymbol{x},\boldsymbol{y}) = \mathscr{N}\left(\boldsymbol{x},\boldsymbol{y} \mid [\boldsymbol{\mu}_x,\boldsymbol{\mu}_y], \begin{bmatrix} \boldsymbol{\Sigma}_x & \boldsymbol{\Sigma}_{xy} \\ \boldsymbol{\Sigma}_{yx} & \boldsymbol{\Sigma}_y \end{bmatrix}\right)$$

then

$$p(\boldsymbol{x} \mid \boldsymbol{y}) = \mathscr{N}\left(\boldsymbol{x} \mid \boldsymbol{\mu}_x + (\boldsymbol{y}-\boldsymbol{\mu}_y)\boldsymbol{\Sigma}_y^{-1}\boldsymbol{\Sigma}_{yx}, \boldsymbol{\Sigma}_x + \boldsymbol{\Sigma}_{xy}\boldsymbol{\Sigma}_y^{-1}\boldsymbol{\Sigma}_{yx}\right), \quad \text{(Conditioning)}$$

$$p(\boldsymbol{x}) = \langle p(\boldsymbol{x} \mid \boldsymbol{y})\rangle_{p(\boldsymbol{y})} = \mathscr{N}(\boldsymbol{x} \mid \boldsymbol{\mu}_x, \boldsymbol{\Sigma}_x). \quad \text{(Marginalization)}$$

2.2 Gaussian processes

Let $\boldsymbol{X},\boldsymbol{y}$ be the training dataset with N inputs. We assume that the observations $\boldsymbol{y}$ are a noisy version of a non-observed (latent) vector $\boldsymbol{f} \in \mathbb{R}^N$. If we choose multivariate zero-mean Gaussian priors for $\boldsymbol{f}$, we get (RASMUSSEN; WILLIAMS, 2006):

$$\begin{aligned} p(\boldsymbol{y},\boldsymbol{f} \mid \boldsymbol{X}) &= \mathscr{N}(\boldsymbol{y} \mid \boldsymbol{f}, \sigma^2\boldsymbol{I})\mathscr{N}(\boldsymbol{f} \mid \boldsymbol{0}, \boldsymbol{K}_f), \\ p(\boldsymbol{y} \mid \boldsymbol{X}) &= \langle p(\boldsymbol{y},\boldsymbol{f})\rangle_{p(\boldsymbol{f}\mid\boldsymbol{X})} = \mathscr{N}(\boldsymbol{y} \mid \boldsymbol{0}, \boldsymbol{K}_f + \sigma^2\boldsymbol{I}), \end{aligned} \tag{2.1}$$

where

$[\boldsymbol{K}_f]_{ij} = k(\boldsymbol{x}_i,\boldsymbol{x}_j)$, and k is a kernel function.

Due to the assumptions made, the latent variables $\boldsymbol{f}$ can be analytically integrated out. The kernel function k defines the covariance of the distribution $p(\boldsymbol{f} \mid \boldsymbol{X})$ and has to such that $\boldsymbol{K}_f$ remains a valid covariance matrix for any possible matrix of inputs $\boldsymbol{X}$. Through this prior distribution, smoothing is done by computing the posterior distribution, and prediction is achieved through the predictive distribution.

Which functions can be represented by the posterior of a GP depends on the choice of a kernel function and the value of any of its hyperparameters. This is because the posterior is proportional to $p(\boldsymbol{y} \mid \boldsymbol{f})p(\boldsymbol{f} \mid \boldsymbol{X})$. In other words, functions with small probability density in the prior distribution contribute very little to the posterior distribution. Since the only parameter of the prior distribution is the kernel function, it plays an essential role in determining which functions can be learned.

Because of the assumption that the output data follows a Gaussian distribution and that a kernel function builds its covariance matrix, there is a Gaussian distribution $p(y_* \mid \boldsymbol{x}_*)$ on for every element of the domain of the kernel function. This distribution depends on all the other distributions induced by the elements of the domain, including the training dataset $p(\boldsymbol{y} \mid \boldsymbol{x})$. This potentially infinite distribution is called a Gaussian process.

So, given an arbitrary data point $\boldsymbol{x}_* \in \mathbb{R}^D$, the expected value of the output that would be observed y_* can be estimated using the distribution $p(y_* \mid \boldsymbol{x}_*, \boldsymbol{y}, \boldsymbol{X})$. Using the properties of the multivariate Gaussian distribution and the kernel function this distribution is:

$$\begin{aligned} p(y_* \mid \boldsymbol{x}_*, \boldsymbol{y}, \boldsymbol{X}) &= \langle p(y_* \mid \boldsymbol{x}_* \boldsymbol{f}, \boldsymbol{X}) \rangle_{p(\boldsymbol{f} \mid \boldsymbol{X}, \boldsymbol{y})} \\ &= \mathcal{N}\left(y_* \mid \boldsymbol{y}\left(\boldsymbol{K}_f + \sigma^2 \boldsymbol{I}\right)^{-1} \boldsymbol{k}_{f*},\ k(\boldsymbol{x}_*, \boldsymbol{x}_*) + \boldsymbol{k}_{*f}\left(\boldsymbol{K}_f + \sigma^2 \boldsymbol{I}\right)^{-1} \boldsymbol{k}_{*f}^{\mathsf{T}} + \sigma^2\right), \end{aligned} \tag{2.2}$$

where

$$\left[\boldsymbol{k}_{*f}\right]_i = k(\boldsymbol{x}_*, \boldsymbol{x}_i).$$

Note, as seen in Equation 2.2, computing the mean and variance for the predictive distribution requires the inversion of an $n \times n$ matrix. The cost of this operation increases cubically with the number of training inputs. Many authors proposed solutions to alleviate this problem, but the most relevant to this dissertation was presented by Titsias (2009).

2.3 Variational sparse Gaussian process

The sparse Gaussian process (SGP) is essentially a Gaussian process with a bottleneck. Instead of storing the entire training dataset, this model adds a new stochastic variable $\boldsymbol{u} \in \mathbb{R}^m$ that depends on a new set of inducing inputs $\Xi \in \mathbb{R}^{m \times D_x}$[1] in Equation 2.1:

$$\begin{aligned} p(\boldsymbol{y}, \boldsymbol{f}, \boldsymbol{u} \mid \boldsymbol{X}, \Xi) &= \mathcal{N}\left(\boldsymbol{y} \mid \boldsymbol{f}, \sigma^2 \boldsymbol{I}\right) \mathcal{N}\left(\boldsymbol{f}, \boldsymbol{u} \mid \boldsymbol{0}, \begin{bmatrix} \boldsymbol{K}_f & \boldsymbol{K}_{fu} \\ \boldsymbol{K}_{fu}^{\mathsf{T}} & \boldsymbol{K}_u \end{bmatrix}\right) \\ &= \mathcal{N}\left(\boldsymbol{y} \mid \boldsymbol{f}, \sigma^2 \boldsymbol{I}\right) p(\boldsymbol{f} \mid \boldsymbol{X}, \boldsymbol{u}, \Xi) \mathcal{N}(\boldsymbol{u} \mid \boldsymbol{0}, \boldsymbol{K}_u), \end{aligned}$$

where

$$[\boldsymbol{K}_u]_{ij} = k(\boldsymbol{\xi}_i, \boldsymbol{\xi}_j);$$

[1] Note that ξ is the lowercase version of Ξ. These letter are pronounced “ksee”, rhyming with “sea”.

$$\left[\boldsymbol{K}_{fu}\right]_{ij} = k(\boldsymbol{x}_i, \boldsymbol{\xi}_j).$$

With the bottleneck in place, instead of integrating all the variables and doing exact Bayesian inference, Titsias (2009) choose to approximate the posterior distribution with the following parametric distribution by $q(\boldsymbol{f},\boldsymbol{u} \mid \boldsymbol{X},\Xi) = p(\boldsymbol{f} \mid \boldsymbol{X},\boldsymbol{u},\Xi)\mathcal{N}\left(\boldsymbol{u} \mid \check{\boldsymbol{\mu}},\check{\boldsymbol{\Sigma}}\right)$ where $\check{\boldsymbol{\mu}}$ and $\check{\boldsymbol{\Sigma}}$ are parameters[2].

These new parameters Ξ, $\check{\boldsymbol{\mu}}$ and $\check{\boldsymbol{\Sigma}}$ are optimized with respect to the model's evidence $p(\boldsymbol{y} \mid \boldsymbol{X},\Xi)$. However, computing the evidence still requires the inversion of $\boldsymbol{K}_f$ because $p(\boldsymbol{u} \mid \Xi)$ can be integrated away. Nevertheless, the evidence can be algebraically manipulated to include terms with $q(\boldsymbol{u})$:

$$\begin{aligned}
p(\boldsymbol{y} \mid \boldsymbol{X},\Xi) &= \langle p(\boldsymbol{y} \mid \boldsymbol{f})\rangle_{p(\boldsymbol{u},\boldsymbol{f}|\boldsymbol{X},\Xi)} \\
&= \left\langle p(\boldsymbol{y} \mid \boldsymbol{f})\frac{q(\boldsymbol{f},\boldsymbol{u} \mid \boldsymbol{X},\Xi)}{q(\boldsymbol{f},\boldsymbol{u} \mid \boldsymbol{X},\Xi)}\right\rangle_{p(\boldsymbol{u},\boldsymbol{f}|\boldsymbol{X},\Xi)} \\
&= \left\langle \frac{p(\boldsymbol{y} \mid \boldsymbol{f})p(\boldsymbol{f} \mid \boldsymbol{X},\boldsymbol{u},\Xi)p(\boldsymbol{u} \mid \Xi)}{q(\boldsymbol{f},\boldsymbol{u} \mid \boldsymbol{X},\Xi)}\right\rangle_{q(\boldsymbol{f},\boldsymbol{u}|\boldsymbol{X},\Xi)} \\
&= \left\langle \frac{p(\boldsymbol{y} \mid \boldsymbol{f})p(\boldsymbol{f} \mid \boldsymbol{X},\boldsymbol{u},\Xi)p(\boldsymbol{u} \mid \Xi)}{p(\boldsymbol{f} \mid \boldsymbol{X},\boldsymbol{u},\Xi)q(\boldsymbol{u})}\right\rangle_{q(\boldsymbol{f},\boldsymbol{u}|\boldsymbol{X},\Xi)} \\
&= \left\langle p(\boldsymbol{y} \mid \boldsymbol{f})\frac{p(\boldsymbol{u} \mid \Xi)}{q(\boldsymbol{u})}\right\rangle_{q(\boldsymbol{f},\boldsymbol{u}|\boldsymbol{X},\Xi)}.
\end{aligned}$$

The derivation might seem to be stuck at this point, but one last trick remains. By applying log on both sides of the equation, Jensen's inequality can be applied. t

$$\begin{aligned}
\log p(\boldsymbol{y} \mid \boldsymbol{X},\Xi) &= \log\left\langle p(\boldsymbol{y} \mid \boldsymbol{f})\frac{p(\boldsymbol{u} \mid \Xi)}{q(\boldsymbol{u})}\right\rangle_{q(\boldsymbol{f},\boldsymbol{u}|\boldsymbol{X},\Xi)} \\
&\geq \left\langle \log\left[p(\boldsymbol{y} \mid \boldsymbol{f})\frac{p(\boldsymbol{u} \mid \Xi)}{q(\boldsymbol{u})}\right]\right\rangle_{q(\boldsymbol{f},\boldsymbol{u}|\boldsymbol{X},\Xi)} && \text{(By Jensen's inequality)} \\
&\geq \langle \log p(\boldsymbol{y} \mid \boldsymbol{f})\rangle_{q(\boldsymbol{f},\boldsymbol{u}|\boldsymbol{X},\Xi)} + \left\langle \frac{p(\boldsymbol{u} \mid \Xi)}{q(\boldsymbol{u})}\right\rangle_{q(\boldsymbol{u}|\Xi)} \\
&\geq \langle \log p(\boldsymbol{y} \mid \boldsymbol{f})\rangle_{q(\boldsymbol{f},\boldsymbol{u}|\boldsymbol{X},\Xi)} - \mathrm{KL}(q(\boldsymbol{u}) \parallel p(\boldsymbol{u} \mid \Xi)). && (2.3)
\end{aligned}$$

Because Equation 2.3 is a lower bound to the evidence, it is called the evidence lower bound (ELBO). By optimizing this ELBO, the evidence is optimized accordingly. Better yet, as the distribution $q(\boldsymbol{u})$ becomes closer to $p(\boldsymbol{u} \mid \boldsymbol{y})$, this inequality tends to equality. Furthermore, by analyzing the ELBO, the optimal values for $\check{\boldsymbol{\mu}}$ and $\check{\boldsymbol{\Sigma}}$ can be obtained without the need for

[2] The uppercase version of μ if $\boldsymbol{M}$, this letter is pronounced "mew". The lowercase version of $\boldsymbol{\Sigma}$ is σ. These letters are pronounced "sigma", rhyming with "enigma".

gradient-based optimization methods. The closed-form expressions of these parameters depend only on $\boldsymbol{X}$, $\boldsymbol{y}$, and $\boldsymbol{\Xi}$. Throughout this dissertation, I will refer to this distribution as the "optimal distribution for $q(\boldsymbol{u})$".

Now, by inspecting the approximate predictive distribution:

$$\begin{aligned}
q(y_* \mid \boldsymbol{x}_*, \boldsymbol{X}, \boldsymbol{y}) &= \left\langle \mathcal{N}\left(y_* \mid f^*, \sigma^2\right) p(f^* \mid \boldsymbol{f}) q(\boldsymbol{f}, \boldsymbol{u}) \right\rangle_{q(\boldsymbol{f},\boldsymbol{u})} \\
&= \left\langle \mathcal{N}\left(y_* \mid f^*, \sigma^2\right) p(f^* \mid \boldsymbol{u}) q(\boldsymbol{u}) \right\rangle_{q(\boldsymbol{u})} \\
&= \mathcal{N}\left(y_* \mid \check{\boldsymbol{\mu}} \boldsymbol{K}_u^{-1} \boldsymbol{k}_{u*},\, k(\boldsymbol{x}_*, \boldsymbol{x}_*) - \boldsymbol{k}_{*u} \boldsymbol{K}_u^{-1} \left(\boldsymbol{K}_u - \check{\boldsymbol{\Sigma}}\right) \boldsymbol{K}_u^{-1} \boldsymbol{k}_{*u}^{\intercal} + \sigma^2\right).
\end{aligned}$$

The only matrix inversion in this formula is on $\boldsymbol{K}_v$, which is $m \times m$; consequently, these operations increase cubically with m. Indeed, because m is set by the user and is independent of the dataset's size, this model can be trained on datasets where the standard GP could not.

2.4 Bayesian Gaussian Process Latent Variable Model

The GPLVM, which was proposed by Lawrence (2004), extends the GP framework for scenarios where we do not have the inputs $\boldsymbol{X}$, which generated the observations $\boldsymbol{Y}$ via the modeled function. In that approach, a Gaussian prior $p(\boldsymbol{X}) = \prod_{i=1}^{N} \mathcal{N}(\boldsymbol{x}_i | \boldsymbol{0}, \boldsymbol{I})$ is chosen for the unobserved latent variables $\boldsymbol{X}$, and the maximum a posteriori probability (MAP) solution $\boldsymbol{X}_{\text{MAP}} = \arg\max_{\boldsymbol{X}} \log p(\boldsymbol{Y}|\boldsymbol{X}) p(\boldsymbol{X})$ is obtained. Kernel hyperparameters can also be optimized using gradient-based methods. However, such an approach has some drawbacks. First, since it directly optimizes the latent variables, it is susceptible to overfitting. Second, larger D_x values always result in a better fit to the training data, which thwarts the optimization of the latent space dimensionality infeasible.

The GPLVM was initially proposed for nonlinear dimensionality reduction, which can be done by choosing $D_x < D_y$. However, the approach has proved to be flexible enough to be used in several other scenarios. For instance, in supervised tasks, the matrix $\boldsymbol{X}$ can be seen as a set of observed but uncertain inputs (DAMIANOU *et al.*, 2016).

The Bayesian GPLVM, proposed by Titsias and Lawrence (2010), tackles the above issues with a variational approach (JORDAN *et al.*, 1999; BLEI *et al.*, 2017) to approximately integrate the latent variables $\boldsymbol{X}$. Inspired by the variational sparse GP framework from Titsias (2009), the Bayesian GPLVM avoids overfitting by considering the latent space's uncertainty and enables the determination of D_x by using a kernel function with automatic relevance determination (ARD) hyperparameters.

Following Titsias and Lawrence (2010), we start by including m inducing points $\boldsymbol{u}_{:d} \in \mathbb{R}^M$ associated to each output dimension and evaluated in the pseudo-inputs $\Xi \in \mathbb{R}^{m \times D_x}$, where $p(\boldsymbol{u}_{:d}) = \mathcal{N}(\boldsymbol{u}_{:d}|\boldsymbol{0}, \boldsymbol{K}_z)$. The joint distribution of all the variables in the GPLVM is now given by:

$$p(\boldsymbol{Y}, \boldsymbol{X}, \boldsymbol{F}, \boldsymbol{U} \mid \Xi) = \left(\prod_{d=1}^{D_y} p(\boldsymbol{y}_{:d}|\boldsymbol{f}_{:d}) p(\boldsymbol{f}_{:d}|\boldsymbol{X}, \boldsymbol{u}_{:d}, \Xi) p(\boldsymbol{z}_{:d} \mid \Xi) \right) p(\boldsymbol{X}).$$

Following the same steps of the previous section to the above expression gives a closed-form ELBO with the variational distribution $p(\boldsymbol{F}|\boldsymbol{X}, \boldsymbol{U}, \Xi) q(\boldsymbol{X}) \prod_d^{D_y} q(\boldsymbol{u}_{:d})$. Like SGP, the distribution $q(\boldsymbol{u}_{:d})$ has an optimal closed-form for every d. However, $q(\boldsymbol{X})$ does not. Therefore, $q(\boldsymbol{X})$ has to be defined to be as $q(\boldsymbol{X}) = \prod_i^n \mathcal{N}\left(\boldsymbol{x}_i \mid \check{\boldsymbol{\mu}}_{xi}, \check{\boldsymbol{\Sigma}}_{xi}\right)$ where $\check{\boldsymbol{M}}_x$ and $\check{\boldsymbol{\Sigma}}_x$ are the variational parameters. The bound is defined as follows by Titsias and Lawrence (2010):

$$\begin{aligned} p(\boldsymbol{Y} \mid \Xi) \geq & -\frac{nD_y}{2} \log(\sigma^2) - \frac{nD_y}{2} \log(2\pi) \\ & + \frac{D_y}{2} |\boldsymbol{K}_u| - \frac{D}{2} |\sigma^2 \boldsymbol{K}_u + \boldsymbol{\Psi}_2| \\ & + \frac{1}{2\sigma^2} \sum_d^{D_y} \boldsymbol{y}_{:d} \boldsymbol{\Psi}_1 \left(\sigma^2 \boldsymbol{K}_u + \boldsymbol{\Psi}_2\right)^{-1} \boldsymbol{\Psi}_1^{\mathsf{T}} \boldsymbol{y}_{:d}^{\mathsf{T}} - \boldsymbol{y}_{:d} \boldsymbol{y}_{:d}^{\mathsf{T}} \\ & - \frac{D_y \psi_0}{2\sigma^2} + -\frac{D_y}{2\sigma^2} \operatorname{Tr}\left(\boldsymbol{K}_u^{-1} \boldsymbol{\Psi}_2\right) \\ & - \operatorname{KL}(q(\boldsymbol{X}) \parallel p(\boldsymbol{X})), \end{aligned}$$

where:

$$\begin{aligned} \psi_0 &= \sum_i^n \langle k(\boldsymbol{x}_i, \boldsymbol{x}_i) \rangle_{q(\boldsymbol{x}_i)}; \\ [\boldsymbol{\Psi}_1]_{ij} &= \left\langle k(\boldsymbol{x}_i, \boldsymbol{\xi}_j) \right\rangle_{q(\boldsymbol{x}_i)}; \\ [\boldsymbol{\Psi}_2]_{jm} &= \sum_i^n \left\langle k(\boldsymbol{x}_i, \boldsymbol{\xi}_j) k(\boldsymbol{x}_i, \boldsymbol{\xi}_m) \right\rangle_{q(\boldsymbol{x}_i)}. \end{aligned}$$

The ELBO being closed-form or not depends on the three terms, named Ψ-statistics[3]. Those terms represent convolutions of the kernel function with the variational distribution $q(\boldsymbol{X})$ and are tractable only for a few kernel functions, such as the radial basis function (RBF) kernel and the linear kernels. The closed-form Ψ-statistics for the RBF kernel were derived by Titsias and Lawrence (2010).

[3] The lowercase version of Ψ is ψ. These letters are pronounced “sigh”.

3 UNSCENTED GAUSSIAN PROCESS LATENT VARIABLE MODEL

The variational approach for the Bayesian GPLVM presents calculations that are tractable for few choices for the kernel function, like the RBF kernel (TITSIAS; LAWRENCE, 2010). However, this kernel is known to have limited extrapolation capability (MACKAY, 1998); thus, various complex kernels have been proposed: a compositional approach to build more expressive kernels from simpler ones (DUVENAUD *et al.*, 2013; LLOYD *et al.*, 2014), a complex spectral mixture kernel family capable of automatically discovering patterns and extrapolating beyond the training data (WILSON; ADAMS, 2013) and the use of deep neural networks to learn kernel functions directly from the available data (WILSON *et al.*, 2016a; WILSON *et al.*, 2016b; AL-SHEDIVAT *et al.*, 2017).

Although those proposals achieve flexible kernels, the Bayesian GPLVM's evidence lower bound becomes intractable with these kernels. The "reparametrization trick" (KINGMA; WELLING, 2014; REZENDE *et al.*, 2014) introduced in the doubly stochastic variational inference framework by Titsias and Lázaro-Gredilla (2014) is used to handle non-RBF kernels with uncertain inputs (ELEFTHERIADIS *et al.*, 2017; SALIMBENI; DEISENROTH, 2017a). Such an approach results in a flexible inference methodology, but, since it resorts to stochastic sampling, it strays from the deterministic advantage of standard variational methods, such as using popular second-order optimization algorithms like L-BFGS-B (BYRD *et al.*, 1995).

In this chapter, I describe a methodology aimed to handle the issue of propagating the uncertainty in the GPLVM while maintaining the deterministic framework presented by Titsias and Lawrence (2010). The intractabilities of uncertain inputs and non-RBF kernels are tackled by exploiting the unscented transformation (UT), a deterministic technique to approximate nonlinear mappings of a probability distribution (JULIER; UHLMANN, 2004; MENEGAZ *et al.*, 2015). The UT projects a finite number of *sigma points* through a nonlinear function and uses their computed statistics to estimate the transformed mean and covariance, resulting in a method more scalable than the competing deterministic Gauss-Hermite quadrature.

More specifically, I propose approximating the integrals that arise from the convolutions of the kernel function with a Gaussian density in the variational framework by Titsias and Lawrence (2010). It enables the use of any kernel, including those obtained via auxiliary parametric models in a kernel learning setup. Then, I evaluated this methodology in GPLVM's primary task of dimensionality reduction and the task of uncertainty propagation during a free simulation (multistep-ahead prediction) of dynamical models. The results indicate the feasibility of this proposal, both in terms of quantitative metrics and computational effort.

The main contributions contained in this chapter are: (*i*) an extension to the Bayesian GPLVM using the UT to handle intractable integrals deterministically and enable the use of any kernel; (*ii*) a set of experiments comparing the proposed approach and alternative approximations using Gauss-Hermite quadrature and Monte Carlo sampling in tasks involving dimensionality reduction and dynamical free simulation.

The remainder of this chapter is organized as follows: Section 3.1 contains a brief overview of the use of the UT in GP models, Section 3.2 contains the theoretical background by summarizing the UT approximation, Section 3.3 details the proposal to apply the UT within the Bayesian GPLVM setting and Section 3.4 presents and discuss the obtained empirical results. Finally, I conclude the chapter in Section 3.5 with remarks about the results.

3.1 Literature review

A few authors have already considered the UT in the context of GP models. The GP-UKF model (KO *et al.*, 2007; KO; FOX, 2009) extends the Unscented Kalman Filter (UKF) with GP-based transition and observation functions. Other authors (ANGER *et al.*, 2012; WANG *et al.*, 2014; SAFARINEJADIAN; KOWSARI, 2014) have successfully applied this model.

The GPBF-LEARN framework (KO; FOX, 2010) extends the previous works by considering the original GPLVM (LAWRENCE, 2004), where the latent variables are optimized instead of integrated.

The Unscented Gaussian Process (STEINBERG; BONILLA, 2014) tackle other kinds of intractabilities in regular GP models with non-Gaussian likelihood through the UT in a variational framework. The authors evaluated this model in synthetic inversion problems and binary classification.

The above works do not consider the Bayesian GPLVM, where the latent variables representing uncertain inputs are approximately integrated.

3.2 Theoretical background

The unscented transformation (UT) is a method for estimating the first two moments of a random variable transformed under an arbitrary function. First proposed by Uhlmann (1995) for non-linear Kalman filters, the transformation itself is decoupled from the proposed

Unscented Kalman Filter.

In the UT, the transformed random variable's mean and covariance are approximated with a weighted average of transformed *sigma points* $\boldsymbol{S}$, derived from the first two moments of the original input.

Let $p(\boldsymbol{x}) = \mathcal{N}(\boldsymbol{\mu}, \boldsymbol{\Sigma})$, where $\boldsymbol{x} \in \mathbb{R}^D$, be the input of an arbitrary transformation $\boldsymbol{f}\colon \mathbb{R}^D \to \mathbb{R}^Q$. Given uniform weights for the sigma points, the output moments are computed by:

$$\begin{aligned} \langle \boldsymbol{f}(\boldsymbol{x}) \rangle_{p(\boldsymbol{x})} &\approx \frac{\sum_i^{2D} \boldsymbol{f}(\boldsymbol{s}_i)}{2D} = \tilde{\boldsymbol{\mu}}_{\mathrm{UT}}, \\ \mathbf{Cov}(\boldsymbol{f}(\boldsymbol{x})) &\approx \frac{\sum_i^{2D} (\boldsymbol{f}(\boldsymbol{s}_i) - \tilde{\boldsymbol{\mu}}_{\mathrm{UT}})(\boldsymbol{f}(\boldsymbol{s}_i) - \tilde{\boldsymbol{\mu}}_{\mathrm{UT}})^\intercal}{2D}. \end{aligned} \tag{3.1}$$

There are several strategies to select sigma points, e.g., (MENEGAZ *et al.*, 2015). However, I choose to follow the original scheme by Uhlmann (1995), with uniform weights and sigma points chosen from the columns of the squared root of $D\boldsymbol{\Sigma}$, an efficient way to generate a symmetric distribution of sigma points.

This scheme is defined as follows. Let $\mathbf{Chol}(\boldsymbol{\Sigma})$ be the Cholesky decomposition of the matrix $\boldsymbol{\Sigma}$. Then, the sigma points $\boldsymbol{S}$ are defined as:

$$\begin{aligned} \boldsymbol{s}_i &= \boldsymbol{\mu} + [\mathbf{Chol}(D\boldsymbol{\Sigma})]_{:i} \\ \boldsymbol{s}_{i+D} &= \boldsymbol{\mu} - [\mathbf{Chol}(D\boldsymbol{\Sigma})]_{:i}, \quad \forall i \in [0, D), \end{aligned}$$

Julier and Uhlmann (2004) demonstrated that this choice for sigma points and any choice of points that correctly encode the mean and covariance of the input could compute the projected mean and covariance correctly up to the second-order Taylor approximation. As argued in the next section, since I consider Gaussian inputs, the premise mentioned above is satisfied. Further formal justification for the symmetric sigma points choice is presented by Menegaz *et al.* (2015).

Since only a relatively small number of sigma points is used ($2D$) and their computation is entirely deterministic, the UT presents a viable alternative to other quadrature methods, such as Gauss-Hermite and Monte Carlo sampling.

3.3 Proposed methodology

As mentioned in Section 2.4, the computation of the Ψ-statistics in Equations. (3.2)-(3.4) is the only part that prevents the application of the Bayesian GPLVM with arbitrary kernels.

Since the Ψ-statistics are Gaussian expectations of nonlinear functions, I propose to approximate their computation in intractable cases using the mean approximated by the UT (see Equation. (3.1)), as follows:

$$\psi_0 = \sum_i^N \langle k(\boldsymbol{x}_i, \boldsymbol{x}_i) \rangle_{q(\boldsymbol{x}_i)} \approx \frac{1}{2D} \sum_k^{2D} k(\boldsymbol{s}_k, \boldsymbol{s}_k), \tag{3.2}$$

$$[\boldsymbol{\Psi}_1]_{ij} = \left\langle k(\boldsymbol{x}_i, \mathfrak{z}_j) \right\rangle_{q(\boldsymbol{x}_i)} \approx \frac{1}{2D} \sum_k^{2D} k(\boldsymbol{s}_k, \mathfrak{z}_j)), \tag{3.3}$$

$$[\boldsymbol{\Psi}_2]_{jm} = \sum_i^N \left\langle k(\boldsymbol{x}_i, \mathfrak{z}_j) k(\boldsymbol{x}_i, \mathfrak{z}_m) \right\rangle_{q(\boldsymbol{x}_i)} \approx \frac{1}{2D} \sum_i^N \sum_k^{2D} k(\boldsymbol{s}_k^{(i)}, \mathfrak{z}_j)) k(\boldsymbol{s}_k^{(i)}, \mathfrak{z}_m)), \tag{3.4}$$

where $\boldsymbol{\Psi}_1 \in \mathbb{R}^{n \times m}, \boldsymbol{\Psi}_2 \in \mathbb{R}^{m \times m}$, and $\boldsymbol{s}_k^{(i)}$ indicates the k-th sigma point related to $q(\boldsymbol{x}_i)$.

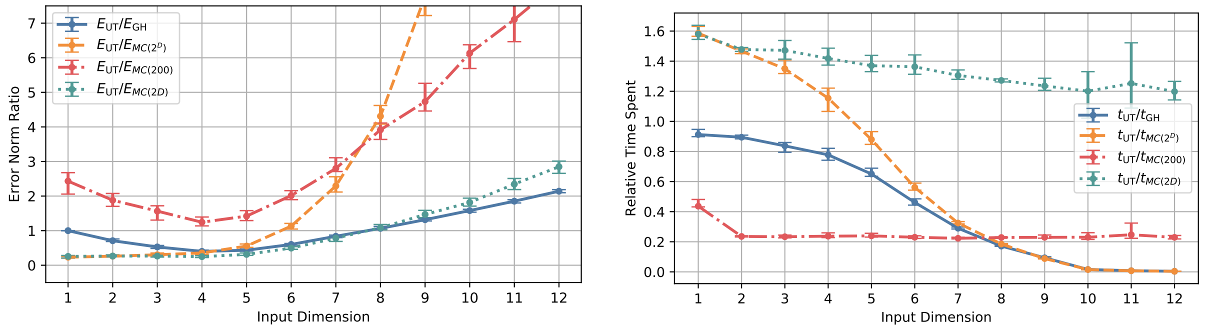
It is important to note that sing UT to solve the Gaussian expectations of nonlinear functions in the above Ψ-statistics is convenient since we can often limit the integrand dimension when learning latent spaces. Furthermore, the UT is most suited for Gaussian integrals with lower dimensionality (HONKELA, 2004), which is usually the case with the Bayesian GPLVM. A similar result was previously obtained by Zhang *et al.* (2009) when comparing UT with other sampling strategies.

Besides enabling the use of non-analytical kernels in the Bayesian GPLVM, using UT-based approximations in place of, for instance, the Gauss-Hermite (GH) quadrature, brings great computational benefits due to the number of points that are evaluated to compute the Gaussian integral. Given a D-dimensional random variable, the UT requires just a linear number of $2D$ sampled points for evaluation. In contrast, the GH quadrature requires H^D points, where H is a user-chosen order parameter. Even for $H = 2$ and moderate dimensionality values, e.g. $D = 20$, the GH approach would require at least 2^{20} evaluations per approximation, which is infeasible. Moreover, the UT and the GH quadrature have similar forms in the single dimension case.

In the Bayesian GPLVM, the amount of sampled points is relevant since the approximations are computed at each step of the variational lower bound optimization. Thus, the number of times the Ψ-statistics are evaluated gives a raw estimate of the chosen approximation computational budget.

To verify how accuracy evolves with dimension when using UT in the context of the Bayesian GPLVM, I computed $\boldsymbol{\Psi}_1$ (see Equation. (3.3)) considering an RBF kernel on random data ($n = 40, m = 20$) of varying dimension and compared the UT result with the GH quadrature and Monte Carlo (MC) sampling. First, D-dimensional input dataset X and matrix of inducing

Figure 1 – Comparison between the UT and other methods for computing Ψ-statistics.



(a) Error ratios between the UT and other approximations.

(b) Relative computational effort between the methods.

points Ξ were generated using samples from a standard Gaussian distribution. Then, on a loop starting from 1 up to D, Equation. (3.3) is computed on a d-dimensional slice of X and Ξ using both approximations and analytical solutions. I measured the approximations' accuracy by comparing the error matrix Frobenius norm between each approximation and the analytical solution, which is feasible for this choice of kernel.

Figure 1a compares the error norm ratios between the UT and four competing quadratures: GH and MC with $2D$, 2^D, and 200 samples. The picture shows a tendency for smaller errors in the GH quadrature as the input dimension increases. However, the UT has better accuracy than the GH on low dimensions (≤ 8). Notably, the UT requires exponentially fewer sample points than GH. The GH variants with more samples outperform the other approximations, especially when the dimensionality increases. Figure 1b illustrates the UT's relative effort compared with each other method in the same scenarios of Figure 1a. There is an increase in time because of the need for more samples, as expected. The difference in time favors UT in some regimes due to its low sampling count.

3.4 EXPERIMENTS

For the experimental validation of the method, I considered two standard tasks for the GPLVM: dimensionality reduction and free simulation of dynamical models with uncertainty propagation. I compared the proposed UT approach with the GH quadrature and the reparametrization trick based MC sampling for computing the Ψ-statistics of the Bayesian GPLVM. In the tractable cases, I also considered analytical expressions. All experiments were implemented in Python using the GPflow framework (MATTHEWS *et al.*, 2017). The code can be found at <https://github.com/spectraldani/UnscentedGPLVM>.

To maintain a reasonable computational cost for the GH experiments, I used 2^D points, where D is the input dimension. For the MC approximations, I used three different numbers of samples: the same number used by UT, the same number used by GH, and a fixed quantity of 200 samples. Each MC experiment was run ten times, with averages and standard deviations reported. The MC approximation is similar to the one in the doubly stochastic variational framework (TITSIAS; LÁZARO-GREDILLA, 2014) but without mini-batch updates.

The kernel hyperparameters, likelihood noise, and variational parameters are all jointly optimized using the second-order optimization method L-BFGS-B (BYRD *et al.*, 1995). However, it is not feasible to use L-BFGS-B for the models that use MC sampling, so these models were optimized using Adam (KINGMA; WELLING, 2014) with a learning rate of 0.01.

3.4.1 DIMENSIONALITY REDUCTION

The dimensionality reduction task is especially suitable for the UT-based approach since the dimension of the integrand in the Ψ-statistics are usually small for data visualization purposes.

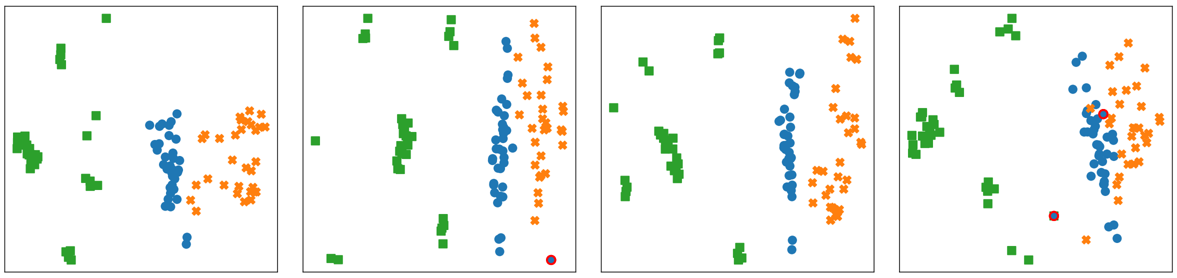
I used two datasets, previously used in Lawrence (2004) and Titsias and Lawrence (2010), the Oil flow dataset and the USPS digit dataset. In both cases, I compared the analytic Bayesian GPLVM model with the RBF kernel against a kernel with non-analytic Ψ-statistics. The following kernels were considered: Matérn 3/2, the periodic kernel[1] and an multilayer perceptron (MLP) composed on an RBF kernel, similar to the manifold learning approach by Calandra *et al.* (2016).

The means of the variational distribution were initialized based on standard principal component analysis (PCA), and the latent variances were initialized to 0.1. Also, 20 points from the initial latent space were selected as inducing pseudo-inputs and were appropriately optimized during training.

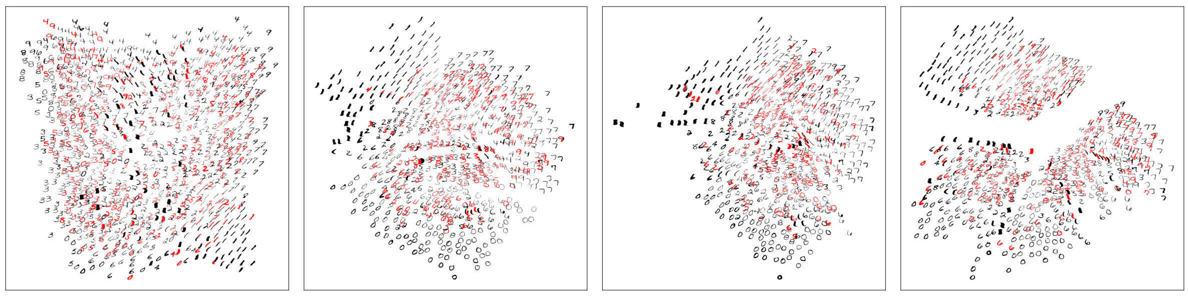
Each scenario was evaluated following two approaches: a qualitative analysis of the learned two-dimensional latent space, a quantitative metric in which I took the known labels from each dataset and computed the predictive accuracy of the predicted classes of points in the latent space. In the latter, I used a five-fold cross-validated 1-nearest neighbors algorithm (1-NN). For the quantitative results, I also show the accuracy of the PCA projection for reference.

[1] As defined by MacKay (1998) at Equation. (47).

Figure 2 – Projections of the Oil flow (top) and USPS digits (bottom) datasets for GPLVM with different kernels and approximations. The projections shown are the best ones obtained in the cross-validation steps. 1-NN mislabels marked in red.



(a) Analytic RBF. (b) Matérn 3/2 (GH). (c) Matérn 3/2 (UT). (d) Matérn 3/2 (MC(32)).



(e) Analytic RBF. (f) MLP kernel (GH). (g) MLP kernel (UT). (h) MLP kernel (MC(200)).

3.4.1.1 *Oil flow dataset*

The multiphase Oil flow dataset consists of 1000 observations with 12 attributes, where each one belongs to one of three classes (BISHOP; JAMES, 1993). I applied GPLVM with five latent dimensions and selected the two dimensions with the greatest inverse lengthscales.

For the approximations with the GH quadrature, I used $2^5 = 32$ samples. This contrasts with the UT, which only uses $2 \cdot 5 = 10$ samples. Note that I have attempted to follow Titsias and Lawrence (2010) and use ten latent dimensions, but that would require the GH to evaluate $2^{10} = 1024$ samples at each optimization step, making the method too slow on the used hardware.

Figure 2 shows that independent of the chosen method to solve the Ψ-statistics, either the analytic expressions or any approximation yields similar overall qualitative results. Tables 1 and 2 contain the 1-NN predicted accuracy results for all kernels and approximation methods. As expected, all the nonlinear approaches performed better than regular PCA. The RBF results for the deterministic approaches are similar, while the Matérn 3/2 kernel with the UT approximation obtained slightly better results. However, when using MC sampling with the same amount of points that UT and GH used, the results for all kernels were worse than both

Table 1 – 1-NN accuracies results for the Oil flow dataset. Note that the UT managed to achieve better results while using $\frac{1}{3}$ of the evaluations as GH.

Method	**Samples used**	**Kernel**	**Accuracy**
PCA	-	-	79.0 ± 6.5
Analytic	-	RBF	98.0 ± 2.7
Gauss-Hermite	32	Matérn 3/2	95.0 ± 6.1
		RBF	98.0 ± 2.7
Unscented	10	Matérn 3/2	100.0 ± 0.0
		RBF	98.0 ± 2.7
Monte Carlo	10	Matérn 3/2	85.6 ± 8.7
		RBF	98.2 ± 2.4
	32	Matérn 3/2	87.9 ± 5.4
		RBF	98.0 ± 2.5
	200	Matérn 3/2	95.4 ± 3.0
		RBF	97.0 ± 4.0

Table 2 – 1-NN accuracies results for the USPS dataset. The use of a more complex kernel brought benefits to all methods. Despite its simplicity, the UT has better or similar results on all kernels.

Method	**Samples used**	**Kernel**	**Accuracy**
PCA	-		35.5 ± 1.4
Analytic	-	RBF	36.0 ± 1.0
Gauss-Hermite	4	MLP	68.8 ± 0.9
	32	RBF	36.7 ± 0.6
Unscented	4	MLP	69.0 ± 1.9
	10	RBF	39.0 ± 1.2
Monte Carlo	4	MLP	47.9 ± 1.8
	10	RBF	26.6 ± 1.4
	32	RBF	27.0 ± 1.4
	200	MLP	54.3 ± 1.7
		RBF	29.5 ± 1.5

UT and GH.

3.4.1.2 USPS digit dataset

The United States postal service (USPS) digit dataset contains 7000 16×16 gray-scale images of handwritten numerals from 0 to 9. To soften the required computational effort, I used just 500 samples of each class. I used a GPLVM with five latent dimensions on all kernels except the MLP kernel, where two latent dimensions were used. The same evaluation methodology previously described was followed.

I expected the MLP kernel to fare better than the RBF kernel due to neural networks' well-known capabilities to find lower-dimensional representations of higher dimensional structured data (WILSON *et al.*, 2016a). From Table 2, this was the case since all methods had an increase of 30% accuracy compared to their results with RBF. Note that even MC approximations with more evaluations than UT and GH do not achieve the same results.

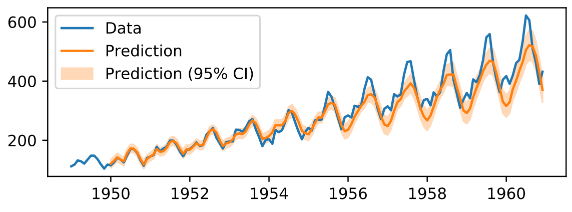
Figure 2 compares the analytic solution with RBF versus the approximate solutions using the MLP kernel with a single hidden layer and [2, 30, 60] neurons (input, hidden, and output, respectively). Visually, the difference between the kernels is as stark, as noted in the quantitative results. These plots also show that the MC approximation finds a very different projection than the other methods that are arguably more difficult to interpret due to the appearance of a gap in the latent data.

3.4.2 DYNAMICAL FREE SIMULATION

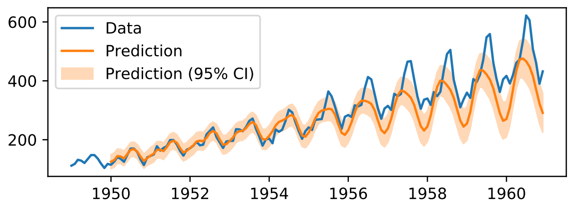
Free simulation, or multistep-ahead prediction, is a task that consists of forecasting the values of a dynamical system arbitrarily far into the future based on past predicted values. In most simple models, such as the Gaussian process nonlinear autoregressive exogenous model (GP-NARX) (KOCIJAN *et al.*, 2005), each prediction does not depend on past predictions' uncertainty, but only past mean predicted values. The lack of dependency between the current predictions and past predictions' uncertainty can be a significant problem because the user cannot be confident about the prediction quality if it does not consider the compounded errors from past estimates.

To propagate the uncertainty of each prediction to the next implies to perform predictions with uncertain inputs. This task has been tackled before, for instance, by Girard *et al.* (2003), but for GP models using the RBF kernel.

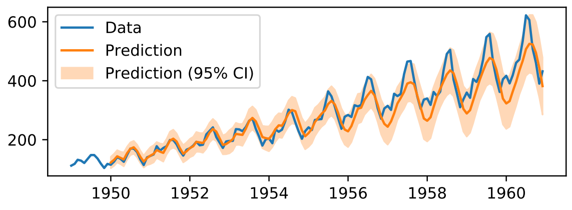
Figure 3 – Results obtained in the dynamical free simulation experiments. The best-obtained runs are shown. Visibly, the MC approximation with 24 points has a much lower quality in its mean compared to the UT approximation by failing to model the peaks of the curve properly.



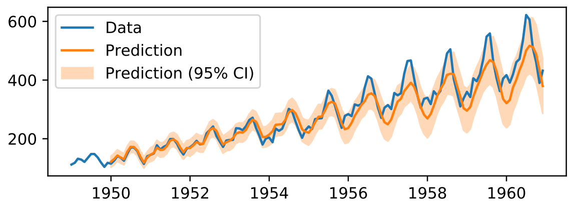
(a) GP-NARX, Periodic + RBF + Linear.



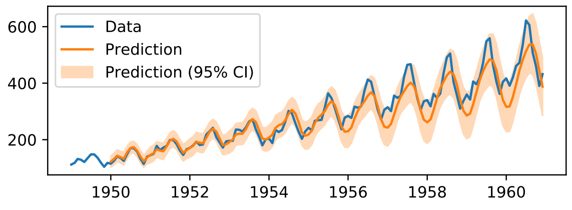
(b) GPLVM, RBF + Linear.



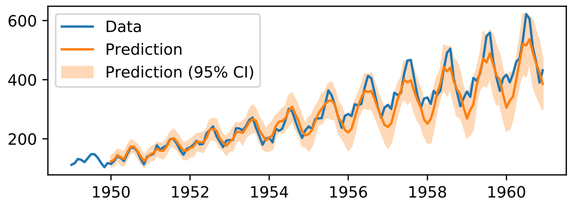
(c) GPLVM, Periodic + RBF + Linear (GH).



(d) GPLVM, Periodic + RBF + Linear (UT).



(e) GPLVM, Periodic + RBF + Linear (MC(4096)).



(f) GPLVM, Periodic + RBF + Linear (MC(24)).

In this section, I first trained a GP-NARX without considering uncertain inputs, following the regular nonlinear autoregressive exogenous model (NARX) approach (KOCIJAN *et al.*, 2005). Then, I applied the same optimized kernel hyperparameters in a GPLVM, selecting all the training inputs as pseudo-inputs. Finally, the GPLVM is used to perform a free simulation with uncertain inputs formed by the past predictive distributions. Since I applied approximations for computing the Ψ-statistics in the predictions, any valid kernel function can be chosen.

3.4.2.1 Airline passenger dataset

The Airline passenger numbers dataset records monthly passenger numbers from 1949 to 1961 (HYNDMAN; YANG, 2018). I used the first four years for training and left the rest for testing, and chose an autoregressive lag of 12 past observations as input. After the GP-NARX kernel hyperparameters are optimized, as previously mentioned, I choose the variance of the variational distribution in the GPLVM to be equal to the optimized noise variance. The free simulation starts from the beginning of the training set until the end of the test set, using past predicted variances as variational variances of the uncertain inputs, enabling approximate uncertainty propagation during the simulation.

I used the following kernels: a mixture of an RBF kernel with a linear kernel, a mixture of periodic, RBF, and linear kernels. The latter combination of kernels was chosen because of my prior knowledge that airplane ticket sales follow a periodic trend and have an overall upward tendency because of the popularity increase and decrease in ticket prices. I emphasize that the choice of such a flexible combination of kernels would not be possible without using approximate methods when considering the uncertain inputs scenario and the GPLVM framework.

Quantitative evaluation is done by computing the root mean squared error (RMSE), given by $\sqrt{\frac{1}{n^*}\sum_i^{n^*}(y_i - \mu_i^*)^2}$, where n^* is the number of test samples, y_i is the actual output, and μ_i^* is the predicted mean output. The average negative log predictive density (NLPD) is also used as an evaluation metric, where the NLPD score is given by $\frac{1}{2}\log 2\pi + \frac{1}{2n^*}\sum_i^{n^*}\left[\log \sigma_i^{*2} + \frac{(y_i-\mu_i^*)^2}{\sigma_i^{*2}}\right]$, and σ_i^{*2} is the i-th predicted variance. Both metrics are “the lower, the better” and are computed only for the test set.

Table 3 presents the obtained results. Although with similar RMSE, all GPLVM variants presented better NLPD values than their standard GP-NARX counterparts. That is expected since the uncertainty of each prediction is being approximately propagated to the next predictions.

Considering this experiment deals with 12-dimensional inputs and following the discussion in Section 3.3, the GH approximation might have better accuracy than the UT approximation. However, its results were better than the equivalent MC sample sizes but had a much better cost-benefit over the other methods given that they use 8 to 170 times more samples for a 0.07 to 0.06 decrease in NLPD. As shown in Figure 3, the visual difference between the two methods is subtle.

3.5 Conclusion

This chapter considered learning GP models from unavailable or uncertain inputs within the Bayesian GPLVM framework. I tackled the intractabilities that arise in the original variational methodology by Titsias and Lawrence (2010) when non-RBF and nonlinear kernels are used by proposing the use of the unscented transformation.

Table 3 – Summary of the free simulation results for the Air passengers dataset. All methods have comparable RMSE, but when comparing with GH, a 170 fold increase in evaluations resulted in just a 0.06 decrease in NLPD.

Method	**Samples used**	**Kernel**	**NLPD**	**RMSE**
GP-NARX	-	RBF+Linear	11.37	69.40
	-	Per.+RBF+Lin.	7.46	44.98
GPLVM - Analytic	-	RBF+Linear	7.08	68.93
GPLVM - GH	4096	RBF+Linear	7.07	68.88
		Per.+RBF+Lin.	5.20	45.00
GPLVM - UT	24	RBF+Linear	7.10	69.11
		Per.+RBF+Lin.	5.26	45.27
GPLVM - MC	24	RBF+Linear	7.52 ± 0.41	71.16 ± 3.15
		Per.+RBF+Lin.	5.41 ± 0.17	46.99 ± 3.04
	200	RBF+Linear	7.09 ± 0.20	68.82 ± 2.09
		Per.+RBF+Lin.	5.19 ± 0.06	45.19 ± 1.32
	4096	RBF+Linear	7.07 ± 0.03	68.81 ± 0.37
		Per.+RBF+Lin.	5.19 ± 0.01	45.29 ± 0.28

I performed experiments on two tasks: dimensionality reduction and free simulation of dynamical models with uncertainty propagation. In both cases, the UT-based approach scaled much better than the compared Gauss-Hermite quadrature while obtaining a similar overall approximation. The UT results were also more stable and consistent than those obtained by Monte Carlo sampling, which may require a more significant number of samples and can not be used with the popular quasi-Newton BFGS optimization algorithm. Notably, the method is simple to implement and does not impose any stochasticity, maintaining the deterministic inference feature of the standard Bayesian GPLVM variational framework.

4 DEEP MAHALANOBIS GAUSSIAN PROCESS

As previously presented in the last chapter, Gaussian processes are a non-parametric alternative to more traditional learning methods that many have applied in various tasks in machine learning. The flexibility of GP models, or in other words, which functions its prior distribution represents, is determined by the kernel function used.

Almost all kernels have hyperparameters that need to be chosen or learned from data. Despite being promoted as a Bayesian model, most users of GP models use type-II maximum likelihood optimization to obtain point estimates of these hyperparameters. This procedure can undermine Bayesian learning's beneficial properties, one example being the resilience to overfitting, but this is not a problem when using kernels with a small number of hyperparameters. However, when using more expressive kernels, a higher amount of training data is needed to acquire reasonable point estimates of its hyperparameters. To regain the guarantees of proper Bayesian learning, some authors have explored non-deterministic Markov chain Monte Carlo estimation methods to approximate the posterior of these full Bayesian models for any kernel (MURRAY; ADAMS, 2010; HENSMAN *et al.*, 2015), and others explored a deterministic variational approach for RBF and Mahanalobis kernels (TITSIAS; LáZARO-GREDILLA, 2013).

Nonetheless, even when the kernel has the best hyperparameters possible, the model might not be expressive enough to learn the desired function. Some of the widely used kernels, such as RBF and the Matérn family, are stationary, which means that the classes of functions they induce can only have trends that depend solely on the inputs' relative location, but not the positions relative to the origin. This limitation rules out the recovery of functions with varying smoothness or behavior that depends on the input coordinates.

This issue leads researchers to develop kernels that can overcome them. Some of the proposals include more expressive kernels (WILSON; ADAMS, 2013), automatic search of kernels through the composition of simpler ones (DUVENAUD *et al.*, 2013; LLOYD *et al.*, 2014), and the use of GPs to drive the hyperparameters of existing kernels (PACIOREK; SCHERVISH, 2004).

Many of the current advances in machine learning have come from moving away from expert-designed feature extractors to learning how to learn the feature space from data without any human engineering of the features (LECUN *et al.*, 2015), primarily through the composition of simpler functions. This strategy maps the input data into latent spaces that are more suitable for processes that use simple kernels. This mapping can be done either through deep neural networks (WILSON *et al.*, 2016a; WILSON *et al.*, 2016b) or through the composition of GPs (LAWRENCE; MOORE, 2007; DAMIANOU; LAWRENCE, 2013).

In this chapter, I explore a novel model that achieves modeling flexibility while keeping variational Bayesian learning of the kernel hyperparameters. This model, dubbed deep Mahalanobis Gaussian process (DMGP), builds upon the Mahalanobis kernel hyperparameters' variational inference and parallels the development of compositional DGPs that builds upon GPLVM. Then, I evaluate this model in the regression tasks with synthetic and empirical datasets.

This chapter is divided as follows: Section 4.1 briefly reviews the use of GP models in tackling functions with variable smoothness. Section 4.2 contains the background of the proposed model by addressing the theory behind the variational integration of Mahalanobis kernel hyperparameters and deep Gaussian processes. Section 4.3 presents the model and a proposal for inference. Finally, Section 4.4 evaluates a specific instance of this model in synthetic and empirical experiments.

4.1 Literature review

In many domains of interest, domain-specific knowledge rules out the assumption that the underlying function that one wants to learn has uniform smoothness. Many authors suggested means to ease the learning of these functions using Gaussian processes. These proposals can be summarized as follow: map the input data to a latent space where the target function can be smoother, drive the hyperparameters of stationary kernels by functions, or use models that are composed of mixtures of models over subsets of the data.

Two significant approaches to mapping input data into latent representations exist, which have a smoother relation to the outputs: deep kernel learning (WILSON *et al.*, 2016a) and DGP (LAWRENCE; MOORE, 2007; DAMIANOU; LAWRENCE, 2013). In deep kernel learning, a parametric deep neural network (DNN) is used as a deterministic mapping function between the input and a latent space. The output of this DNN goes to a GP with a simple kernel, and these two models are jointly optimized. However, the parameters of this DNN are not trained through a Bayesian procedure but using maximum likelihood estimation. Thus, this approach is only suitable for large datasets.

Concerning DGPs, Dunlop *et al.* (2018) presents four possible constructions of hierarchical GPs. Two of those constructions are closely related to this dissertation: by the composition of processes and by making a GP the other's kernel function. The first instance is more common and is ordinarily referred to as the deep Gaussian process. By composing a GP's output into another's input, the same effect seen with deep kernel learning is achieved. However, this method is more susceptible to full Bayesian learning.

The second construction is less common but has been worked on by other authors. As shown by Paciorek (2003), for any stationary kernel k, it can be expressed as the following form:

$$k(\boldsymbol{a},\boldsymbol{b}) = \phi\left((\boldsymbol{a}-\boldsymbol{b})\boldsymbol{\Delta}^{-1}(\boldsymbol{a}-\boldsymbol{b})^{\intercal}\right), \tag{4.1}$$

where $\boldsymbol{\Delta}$ is the kernel lengthscale matrix[1] and ϕ is a scalar function[2]. Note that for kernels with uniform lengthscales $\boldsymbol{\Delta} = \ell^2\boldsymbol{I}$, and for some scalar ℓ, for kernels with ARD lengthscales $\boldsymbol{\Delta}$ is a diagonal matrix with diagonal $\boldsymbol{\ell}^2$.

The author then construct a new kernel k_{NS} based on ϕ such that:

$$k_{\text{NS}}(\boldsymbol{a},\boldsymbol{b}) = \sqrt{2}\,\frac{|\boldsymbol{\Delta}(\boldsymbol{a})|^{\frac{1}{4}}|\boldsymbol{\Delta}(\boldsymbol{b})|^{\frac{1}{4}}}{|\boldsymbol{\Delta}(\boldsymbol{a})+\boldsymbol{\Delta}(\boldsymbol{b})|^{\frac{1}{2}}}\,\phi\left((\boldsymbol{a}-\boldsymbol{b})\frac{(\boldsymbol{\Delta}(\boldsymbol{a})+\boldsymbol{\Delta}(\boldsymbol{b}))^{-1}}{2}(\boldsymbol{a}-\boldsymbol{b})^{\intercal}\right),$$

where $\boldsymbol{\Delta}(\boldsymbol{x})$ is a function that returns positive semi-definite matrices.

Placing a warped GP prior on $\boldsymbol{\Delta}$ produces their proposed two-layer model, but the authors opt to use Markov chain Monte Carlo (MCMC) inference due to the model's complexity. Salimbeni and Deisenroth (2017b) generalized this model by allowing multiple layers while allowing variational inference using the doubly stochastic variational inference (DSVI) framework. This same kernel transformation procedure was applied to the Spectral Mixture kernel by Remes *et al.* (2017), but the authors decided to use MAP inference on the hidden layer processes.

However, by replacing $\boldsymbol{\Delta}$ with a function $\boldsymbol{\Delta}(\boldsymbol{x})$, Gibbs (1997) shows that the input-varying lengthscales of these kernels do not inherit the semantics associated with the original stationary kernel. Also, it is well known that the term $(\boldsymbol{a}-\boldsymbol{b})\boldsymbol{\Delta}^{-1}(\boldsymbol{a}-\boldsymbol{b})^{\intercal}$ defines an inner product space, as if by the projection of $\boldsymbol{a}$ and $\boldsymbol{b}$ through the matrix $\mathbf{Chol}(\boldsymbol{\Delta})$. Nevertheless, the quadratic form in this k_{NS} is $(\boldsymbol{a}-\boldsymbol{b})\frac{(\boldsymbol{\Delta}(\boldsymbol{a})+\boldsymbol{\Delta}(\boldsymbol{b}))^{-1}}{2}(\boldsymbol{a}-\boldsymbol{b})^{\intercal}$, note that the projection matrix of $\boldsymbol{a}$ depends on $\boldsymbol{b}$. As Paciorek (2003) discusses, this change means that this new kernel can not induce metrics because the triangle inequality is violated. Therefore, two properties that contribute to the lengthscales' interpretability are lost in this family of kernels.

An alternative to learning functions with variable smoothness and input dependent noise is to mix multiple stationary GPs through a weighted sum depending on the input data. Because each GP has its distinct kernel and noise variance, this model can capture input-dependent

[1] The lowercase version of $\boldsymbol{\Delta}$ is δ. This letter is pronounced "delta", like the landform at the mouth of a river.

[2] The uppercase version of ϕ is $\boldsymbol{\Phi}$. This letter is pronounced "fie".

phenomena. A type of mixture model was developed by Rasmussen and Ghahramani (2002) and by Meeds and Osindero (2006) in which various GPs are trained on subsets of the data to obtain a scalable and non-stationary model by mixing them. However, these methods had to rely on sampled inference due to their complex nature. Another type of mixture was introduced by Wilson *et al.* (2012) and improved by Nguyen and Bonilla (2013). This mixture model is defined by the equation $\boldsymbol{y}(\boldsymbol{x}) = \boldsymbol{W}(\boldsymbol{x})(\boldsymbol{f}(\boldsymbol{x}) + \boldsymbol{\varepsilon})$ where $\boldsymbol{f}$ is a vector of independent GPs, $\boldsymbol{\varepsilon}$ is an independent Gaussian noise vector, and $\boldsymbol{W}$ is a matrix of independent GPs. The main advantage of this approach is that deterministic variational inference is possible.

4.2 Theoretical background

In this section, I briefly describe two models used as inspiration and building blocks for the proposed method. First, an overview of the variational inference for Mahalanobis distance metric Gaussian process (VDMGP) model proposed by Titsias and Lázaro-Gredilla (2013), where a variational inference method was presented for a Gaussian process for regression (GPR) model that performs dimensionality reduction and Bayesian learning of the kernel hyperparameters at the same time. Lastly, I discuss how DGP models are built and how DSVI is implemented in that model through the work of Salimbeni and Deisenroth (2017a).

4.2.1 Variational Inference for Mahalanobis kernel hyperparameters

The variational inference for Mahalanobis distance metric Gaussian process (VDMGP) model is an alternative to GPR models that use the squared exponential family of kernels. This kernel family includes the ordinary RBF kernel, the ARD-RBF kernel, and the Mahalanobis distance metric kernel. Before starting with the model's description, I will go through some of the minutiae of this family of kernels.

Any kernel k of the squared exponential family has the following form:

$$k(\boldsymbol{a},\boldsymbol{b}) = \sigma_f^2 \exp\left[-\frac{1}{2}\|\boldsymbol{a}\boldsymbol{W}^\intercal - \boldsymbol{b}\boldsymbol{W}^\intercal\|^2\right],$$

where values $\boldsymbol{W}$ and σ_f are the hyperparameters of the kernel, notice that this equation follows the form presented in Eq. 4.1 with $\boldsymbol{\Delta}^{-1} = \boldsymbol{W}\boldsymbol{W}^\intercal$.

The form of the matrix $\boldsymbol{W}$ determines the type of kernel from this family. If $\boldsymbol{W} = \ell^{-1}\boldsymbol{I}$ for some scalar ℓ, then we say that k is an RBF kernel. If $\boldsymbol{W}$ is a diagonal matrix, then we say that k is an ARD-RBF kernel, with the advantage that the value of diagonal controls the relevance of the associated input dimension. Finally, if $\boldsymbol{W}$ is an arbitrary matrix $\mathbb{R}^{Q\times D}$, then k is said to be a Mahalanobis distance kernel, which has an important property of performing a linear projection of the input space into a smaller one if $Q < D$. In addition to these kernels, when $\boldsymbol{W} = \boldsymbol{I}$ and $\sigma_f = 1$, k has no hyperparameters and is called an standard RBF kernel (SRBF).

Kernel hyperparameters are difficult to be learned through Bayesian methods because they appear non-linearly in the equations of a GP model, like the inverse of $\boldsymbol{K}_f$ in the equation for the predictive distribution (Eq. 2.2). However, if MLE optimization is done on kernels with many hyperparameters, like the Mahalanobis distance kernel, overfitting may occur.

The main contribution of Titsias and Lázaro-Gredilla (2013) is a way to avoid non-linear terms which depend on $\boldsymbol{W}$ to appear on the equations of the model. Remember that in Equation 2.3, the only inverse kernel matrix that appear in the equation is $\boldsymbol{K}_u^{-1}$ and not $\boldsymbol{K}_f$. By forcing the latent process $\boldsymbol{u}$ to use a kernel like SRBF, it might be possible to avoid inverses of matrices that depend on the hyperparameters. Indeed, the VDMGP is defined as follows:

$$
\begin{aligned}
p(\boldsymbol{W}) &= \prod_{q,d}^{Q,D} \mathcal{N}\left(w_{qd} \mid 0, r_d^2\right);\\
p(\boldsymbol{s} \mid \boldsymbol{Z}) &= \mathcal{N}(\boldsymbol{s} \mid \boldsymbol{0}, \boldsymbol{K}_s);\\
\boldsymbol{f} &= \sigma_f \boldsymbol{s};\\
\boldsymbol{y} &= \mathcal{N}\left(\boldsymbol{y} \mid \boldsymbol{f}, \sigma^2 \boldsymbol{I}\right),
\end{aligned}
$$

where:

$$
\begin{aligned}
\boldsymbol{z}_i &= \boldsymbol{W}\boldsymbol{x}_i^\top;\\
[\boldsymbol{K}_s]_{ij} &= \mathrm{SRBF}(\boldsymbol{z}_i, \boldsymbol{z}_j).
\end{aligned}
$$

By this definition, $p(\boldsymbol{f} \mid \boldsymbol{W}, \boldsymbol{X}) = \mathcal{N}\left(\boldsymbol{0}, \boldsymbol{K}_f\right)$ where $\boldsymbol{K}_f$ is the matrix derived from a Mahalanobis distance kernel with hyperparameters $\boldsymbol{W}$ and σ_f. Then, by placing inducing variables $\boldsymbol{u}$ in $\boldsymbol{s}$ instead of $\boldsymbol{f}$ and placing an appropriate variational distribution on $\boldsymbol{u}$ and $\boldsymbol{W}$, the following ELBO is obtained:

$$
\begin{aligned}
p(\boldsymbol{y} \mid \boldsymbol{X}, \mathfrak{Z}) \geq & -\frac{1}{2}\left\langle \sigma^{-2}(\boldsymbol{y} - \sigma_f \boldsymbol{s})^\top (\boldsymbol{y} - \sigma_f \boldsymbol{s})\right\rangle_{q(\boldsymbol{s},\boldsymbol{u},\boldsymbol{W} \mid \boldsymbol{X}, \mathfrak{Z})} + n\ln(2\pi) + \ln\left|\sigma^2 \boldsymbol{I}\right|\\
& - \mathrm{KL}(q(\boldsymbol{u}) \parallel p(\boldsymbol{u} \mid \mathfrak{Z}))\\
& - \sum_{q,d}^{Q,D} \mathrm{KL}(q(w_{qd}) \parallel p(w_{qd})),
\end{aligned}
$$

where:

$$\boldsymbol{\mathfrak{Z}} \in \mathbb{R}^{Q\times m};$$

$q(\boldsymbol{s})$ is the optimal distribution;

$$q(\boldsymbol{W}) = \prod_{q,d}^{Q,D} \mathcal{N}\left(w_{qd} \mid \check{\mu}_{Wqd}, \check{\sigma}_{Wq}\right).$$

No term in this ELBO depends on the inverse of $\boldsymbol{K}_f$, and all expected values have closed-form solutions. With $m + QD + Q + 2$ total parameters, this model does not add many new parameters while still enjoying the advantages of Bayesian learning compared to just using MLE for the kernel hyperparameters.

4.2.2 Deep Gaussian processes

First proposed by Lawrence and Moore (2007) and improved by Damianou and Lawrence (2013), compositional deep Gaussian process (DGP) is a class of models that build upon GPLVM. In Equation 2.4 that in a Bayesian GPLVM model, the input data is assumed to be a Gaussian random variable instead of plain deterministic data, therefore, the input data can be considered a GPLVM of another set of inputs. By composing the input of a process into the output of another, a very flexible model can be built. Let's start with the description of a single node of this model, it is defined as:

$$p(\boldsymbol{X}) = \prod_{d}^{D} p(\boldsymbol{x}_d);$$

$$p(\boldsymbol{F} \mid \boldsymbol{X}) = \prod_{q}^{Q} \mathcal{N}\left(\boldsymbol{f_q} \mid \boldsymbol{\mu}_f, \boldsymbol{K}_f\right),$$

where:

$$\left[\boldsymbol{\mu}_f\right]_i = \mu(\boldsymbol{x}_i);$$

$$\left[\boldsymbol{K}_f\right]_{ij} = k(\boldsymbol{x}_i, \boldsymbol{x}_j).$$

In this equation, $p(\boldsymbol{x}_d)$ can be either a Gaussian distribution, a GP distribution, or deterministic. As it can be compared with Equation 2.4, this node is almost the same as Bayesian GPLVM except that now we allow the distribution on $\boldsymbol{X}$ to be a GP, and the mean vector is based on a function on $\boldsymbol{X}$ instead of being always zero. The change from a zero-mean process to a linear mean process is due to pathologies that arise when composing zero mean GPs (DUVENAUD *et al.*, 2014; SALIMBENI; DEISENROTH, 2017a).

The simplest DGP model is simply a sequence of these nodes one after the other, and, for simplicity, I will only describe the model with a single hidden layer. I refer to this model as DGP(2) despite its relation with Bayesian warped GP (LáZARO-GREDILLA, 2012).

$$p(\boldsymbol{Z} \mid \boldsymbol{X}) = \prod_{q}^{Q} \mathcal{N}\left(z_{:q} \mid \boldsymbol{\mu}_{hq}, \boldsymbol{K}_h\right);$$
$$p(\boldsymbol{f} \mid \boldsymbol{Z}) = \mathcal{N}\left(\boldsymbol{h} \mid \boldsymbol{\mu}_f, \boldsymbol{K}_f\right);$$
$$p(\boldsymbol{y} \mid \boldsymbol{f}) = \mathcal{N}\left(\boldsymbol{y} \mid \boldsymbol{f}, \sigma^2 \boldsymbol{I}\right),$$

where:

$$\boldsymbol{M}_{h:} = \boldsymbol{X}\boldsymbol{P}_x^{\top};$$
$$[\boldsymbol{K}_h]_{ij} = \text{ARD-RBF}_h(\boldsymbol{x}_i, \boldsymbol{x}_j);$$
$$\boldsymbol{\mu}_f = \boldsymbol{Z}\boldsymbol{P}_z^{\top};$$
$$\left[\boldsymbol{K}_f\right]_{ij} = \text{ARD-RBF}_f(\boldsymbol{z}_i, \boldsymbol{z}_j).$$

And each layer has its own ARD-RBF kernel, and $\boldsymbol{P}_x$ and $\boldsymbol{P}_z$ are PCA projection matrices of $\boldsymbol{X}$ and $\boldsymbol{Z}$.

Variational inference, just like GPLVM, is applicable for this model, as done by Damianou and Lawrence (2013). However, a different variational method was proposed in this context by Salimbeni and Deisenroth (2017a). This method allows for better retention of the correlation between layers and permits mini-batching at the cost of non-deterministic inference; however, it is also empirically better. The ELBO for the two-layer model is:

$$p(\boldsymbol{y} \mid \boldsymbol{X}, \mathfrak{Z}, \Xi) \geq \sum_{i}^{N} \langle \log p(y_i \mid f_i) \rangle_{q(f_i)} - \text{KL}(q(\boldsymbol{u}) \parallel p(\boldsymbol{u} \mid \mathfrak{Z}))$$
$$- \sum_{q}^{Q} \text{KL}(q(\boldsymbol{v}_{:q}) \parallel p(\boldsymbol{v}_{:q} \mid \Xi)),$$

where:

$$q(\boldsymbol{V}) = \prod_{q}^{Q} \mathcal{N}\left(\boldsymbol{v}_{:q} \mid \check{\boldsymbol{\mu}}_v, \check{\boldsymbol{\Sigma}}_v\right);$$
$$q(\boldsymbol{u}) = \mathcal{N}\left(\boldsymbol{u} \mid \check{\boldsymbol{\mu}}_u, \check{\boldsymbol{\Sigma}}_u\right);$$
$$q(f_i) = \langle q(f_i \mid \boldsymbol{x}_i, \boldsymbol{U}, \Xi, \boldsymbol{V}, \mathfrak{Z}) \rangle_{q(\boldsymbol{V}, \boldsymbol{U})}.$$

And $\mathfrak{Z}$ is the pseudo inputs of the layer closest to the outputs[3].

Note that the equation of $q(f_i)$ does not have a closed-form expression. This is why the integral $\langle \log p(y_i \mid f_i \rangle_{q(f_i)}$ has to be solved using MC integration. Hence, the ELBO has to be

[3] The lowercase version of $\mathfrak{Z}$ is $\mathfrak{z}$. These letters are pronounced zeta in this dissertation.

estimated through non-deterministic means. In this chapter, the DGP with this inference method is referred to as doubly stochastic deep G aussian process (DS-DGP).

4.3 Proposed methodology

In an analogous manner to the development of DGP by creating networks of the Bayesian GPLVM nodes, in this section, I will present how to extend VDMGP to build a deep model that incorporates variable smoothness. Since VDMGP allows kernel hyperparameters to be drawn from a Gaussian distribution, replacing this input-independent distribution with an input-dependent GP should be viable. Figure 5 shows the parallel between the proposed model and DGP concerning VDMGP and Bayesian GPLVM.

By generalizing the Gaussian prior in $\boldsymbol{W}$ to a GP prior, a network of these nodes can be built, but instead of composing inputs into outputs, it is a composition of layer outputs into layer kernel hyperparameters. Therefore, every node of the network still directly depends on the input data $\boldsymbol{x}$, in contrast with the compositional DGP models where only the first hidden nodes directly depend on the input data.

The description of a node in a Deep VDMGP network can be characterized by:

$$p(\boldsymbol{W} \mid \boldsymbol{X}) = \prod_{q,d}^{Q,D} p(\boldsymbol{w}_{:qd} \mid \boldsymbol{X});$$

$$p(\boldsymbol{s} \mid \boldsymbol{Z}) = \mathcal{N}(\boldsymbol{s} \mid \boldsymbol{0}, \boldsymbol{K}_s);$$

$$\boldsymbol{f} = \sigma_f \boldsymbol{s},$$

where:

$$\boldsymbol{z}_i = \boldsymbol{W}_i \boldsymbol{x}_i^{\mathsf{T}};$$

$$[\boldsymbol{K}_s]_{ij} = \mathrm{SRBF}(\boldsymbol{z}_i, \boldsymbol{z}_j);$$

$p(\boldsymbol{w}_{:qd} \mid \boldsymbol{X})$ is either a Gaussian distribution, a GP distribution, or deterministic.

By setting a GP prior distribution in $p(\boldsymbol{w}_{:qd} \mid \boldsymbol{X})$, a hierarchical model is defined where one node controls the other's smoothness. If a Gaussian prior with no dependency on $\boldsymbol{X}$ is set, a VDMGP model is recovered, and if $\boldsymbol{W}$ is deterministic, an ordinary GP model is recovered. The $\boldsymbol{W}$ for hidden nodes can only have GP distributions while the $\boldsymbol{W}$ for input nodes can be either Gaussian or deterministic.

For layers that have $p(\boldsymbol{W} \mid \boldsymbol{X})$ as a GP distribution, the prior distribution for the output of that layer given $\boldsymbol{W}$ is:

$$p(\boldsymbol{f} \mid \boldsymbol{W}, \boldsymbol{X}) = \mathcal{N}\left(\boldsymbol{0}, \boldsymbol{K}_f\right),$$

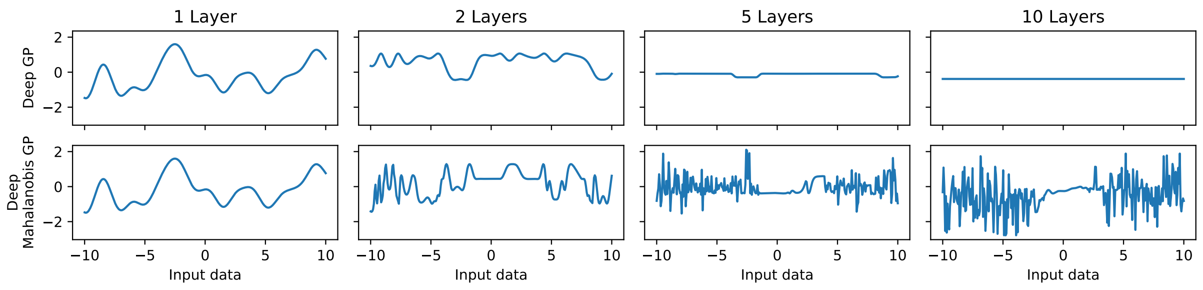
where

$$\begin{aligned}\left[\boldsymbol{K}_f\right]_{ij} &= \sigma_f \exp\left[-\frac{1}{2}\left\|\boldsymbol{z}_i - \boldsymbol{z}_j\right\|^2\right] \\ &= \sigma_f \exp\left[-\frac{1}{2}\left\|\boldsymbol{x}_i \boldsymbol{W}_i^\intercal - \boldsymbol{x}_j \boldsymbol{W}_j^\intercal\right\|^2\right].\end{aligned}$$

Note that the covariance between the outputs has a form identical to the Mahalanobis kernel, except that each input's projection matrix is different. Unlike the DGP constructions seen before, the exponential quadratic term still induces inner product. Indeed, each point's projection only depends on that point and not on both input points, thus preserving the triangle inequality. This is one of the significant differences between the model proposed in this dissertation and other models based on Gibbs (1997) and Paciorek (2003), as these models do not directly induce proper manifolds.

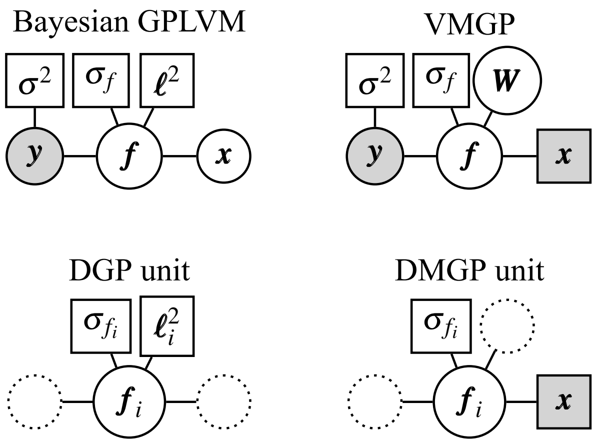
Also, because this model does not compose a layer's output directly into the other's input, there is no need to replace the zero mean GP with another mean function like the DGP model. Here, I reproduce an empirical experiment from Duvenaud *et al.* (2014) where the range of functions sampled from a DGP prior with zero mean reduces to smaller intervals as the number of layers increases.

Figure 4 – Samples from a prior DGP with zero mean and a prior DMGP with zero mean. Each column represent the number of layers of each model with one layer being a regular GP.



As seen in Figure 4, samples from a zero-mean DGP tend to reduce their range as the number of layers increases. Still, such behavior is not present for DMGP; rather, there exists a consistent decrease in smoothness with each attached layer. Nonetheless, the neighborhood around zero is always smooth a prior, but this does not hinder the model's expressivity in empirical tasks, as the experimental results in Section 4.4 show.

Figure 5 – Graphical models for Bayesian GPLVM, VDMGP and the simplest units of DGP and DMGP. σ_f and ℓ^2 represent RBF hyperparameters. Shaded nodes represent observed data, and dashed nodes represent placeholders where deterministic and random variables can be placed.



4.3.1 *Variational inference for deep Mahalanobis Gaussian process*

In this section, I explore a simple DMGP model with support for variational inference. This model only has two layers: the first layer is $Q \cdot D$ GPs connected to a single DMGP node with a Gaussian likelihood in the second layer. Each row of the first layer shares the same ARD-RBF kernel. Because I want to allow for variational inference, I must introduce auxiliary variables and pseudo-inputs on each layer. All processes of the first layer share the same set of pseudo-inputs but have an auxiliary variable for each, and the last layer has a single set of pseudo-inputs and a single auxiliary variable. Formally, given a training dataset $\boldsymbol{X} \in \mathbb{R}^{n \times D}$ and $\boldsymbol{y} \in \mathbb{R}^n$, pseudo-inputs $\boldsymbol{\Xi} \in \mathbb{R}^{m_v \times D}$, and $\boldsymbol{\mathfrak{Z}} \in \mathbb{R}^{m_u \times Q}$, the joint distribution of this model is:

$$\begin{aligned} p(\boldsymbol{y}, \boldsymbol{f}, \boldsymbol{W}, \boldsymbol{u}, \boldsymbol{V} \mid \boldsymbol{X}, \boldsymbol{\Xi}, \boldsymbol{\mathfrak{Z}}) = {} & \mathcal{N}\big(\boldsymbol{y} \mid \boldsymbol{f}, \sigma^2 \boldsymbol{I}\big) p(\boldsymbol{f} \mid \boldsymbol{u}, \boldsymbol{W}, \boldsymbol{X}) \mathcal{N}(\boldsymbol{u} \mid \boldsymbol{0}, \boldsymbol{K}_u) \\ & \cdot \prod_{q,d}^{Q,D} p(\boldsymbol{w}_{qd} \mid \boldsymbol{v}_{qd}, \boldsymbol{X}) \mathcal{N}\left(\boldsymbol{v}_{qd} \,\middle|\, \boldsymbol{0}, \boldsymbol{K}_v^{(q)}\right) \end{aligned}$$

where:

$$p(\boldsymbol{w}_{qd} \mid \boldsymbol{v}_{qd}, \boldsymbol{X}) = \mathcal{N}\left(\boldsymbol{w}_{qd} \,\middle|\, \boldsymbol{v}_{qd} {\boldsymbol{K}_v^{(q)}}^{-1} \boldsymbol{K}_{vw}^{(q)}, \boldsymbol{K}_w^{(q)} - \boldsymbol{K}_{wv}^{(q)} {\boldsymbol{K}_v^{(q)}}^{-1} \boldsymbol{K}_{vw}^{(q)}\right);$$

$$p(\boldsymbol{f} \mid \boldsymbol{u}, \boldsymbol{W}, \boldsymbol{X}) = \mathcal{N}\left(\boldsymbol{f} \mid \boldsymbol{u} {\boldsymbol{K}_u}^{-1} \boldsymbol{K}_{uf}, \boldsymbol{K}_f - \boldsymbol{K}_{fu} {\boldsymbol{K}_u}^{-1} \boldsymbol{K}_{uf}\right);$$

$$\left[\boldsymbol{K}_f\right]_{ij} = \sigma_f^2 \cdot \mathrm{SRBF}\left(\boldsymbol{W}_i \boldsymbol{x}_i^\mathsf{T}, \boldsymbol{W}_j \boldsymbol{x}_j^\mathsf{T}\right);$$

$$\left[\boldsymbol{K}_{fu}\right]_{ij} = \sigma_f \cdot \mathrm{SRBF}\left(\boldsymbol{W}_i \boldsymbol{x}_i^\mathsf{T}, \boldsymbol{\mathfrak{z}}_j\right);$$

$$\left[\boldsymbol{K}_w^{(q)}\right]_{ij} = \mathrm{RBF}_q(\boldsymbol{x}_i, \boldsymbol{x}_j);$$

$$\left[\boldsymbol{K}_v^{(q)}\right]_{ij} = \mathrm{RBF}_q\left(\boldsymbol{\xi}_i, \boldsymbol{\xi}_j\right);$$
$$\left[\boldsymbol{K}_{wv}^{(q)}\right]_{ij} = \mathrm{RBF}_q\left(\boldsymbol{x}_i, \boldsymbol{\xi}_j\right).$$

By analogy with VDMGP, I force each row $\boldsymbol{W}_i$ to share the same variance of the same kernel. This way, the dimensionality of the dataset can be determined by the signal variance of each kernel. I will further illustrate this procedure in Section 4.4.

Two main approaches to variational inference in sparse GP models exist: the one by Titsias (2009) with fewer variational parameters but does not support mini-batching and another one by Hensman *et al.* (2013) that trades more variational parameters for the support of mini-batching. This chapter will only explore the approach that does not support mini-batching but has fewer parameters. The step-by-step derivation of the variational bounds is in Appendix A.

To avoid integrating terms like $p(\boldsymbol{f} \mid \boldsymbol{u}, \boldsymbol{W}, \boldsymbol{X})$, getting solutions with terms that contain $\boldsymbol{K}_f^{-1}$ or $\left(\boldsymbol{K}_w^{(q)}\right)^{-1}$, and having variational parameters for $q(\boldsymbol{u})$, the variational distribution must have the following form, where dependencies on $\boldsymbol{X}$, Ξ, and $\mathfrak{Z}$ were hidden:

$$q(\boldsymbol{f}, \boldsymbol{u}, \boldsymbol{V}) = p(\boldsymbol{f} \mid \boldsymbol{u}, \boldsymbol{W}) p(\boldsymbol{W} \mid \boldsymbol{V}) q(\boldsymbol{u}) q(\boldsymbol{V})$$

where:

$$\begin{aligned}
q(\boldsymbol{V}) &= \prod_{q,d,i}^{Q,D,m_v} \mathcal{N}\left(v_{qdi} \mid \check{\mu}_{vqdi}, \check{\sigma}_{vqi}\right); \\
q(\boldsymbol{u}) &= \mathcal{N}\left(\boldsymbol{u} \mid \boldsymbol{y}\boldsymbol{\Psi}_1\left(\sigma^2\boldsymbol{K}_u + \boldsymbol{\Psi}_2\right)^{-1}\boldsymbol{K}_u, \boldsymbol{K}_u\left(\sigma^2\boldsymbol{K}_u + \boldsymbol{\Psi}_2\right)^{-1}\boldsymbol{K}_u\right); \\
q(\boldsymbol{W}) &= \prod_{q,d}^{Q,D} \mathcal{N}\left(\boldsymbol{w}_{qd} \,\middle|\, \check{\boldsymbol{\mu}}_{vqd}\boldsymbol{K}_v^{(q)^{-1}}\boldsymbol{K}_{vw}^{(q)}, \boldsymbol{K}_w^{(q)} - \boldsymbol{K}_{wv}^{(q)}\boldsymbol{K}_v^{(q)^{-1}}\left(\boldsymbol{K}_v^{(q)} + \check{\boldsymbol{\Sigma}}_{vq}\right)\boldsymbol{K}_v^{(q)^{-1}}\boldsymbol{K}_{vw}^{(q)}\right); \\
&= \prod_{q,d}^{Q,D} \mathcal{N}\left(\boldsymbol{w}_{qd} \mid \tilde{\boldsymbol{\mu}}_{wqd}, \tilde{\boldsymbol{\Sigma}}_{wq}\right); \\
[\boldsymbol{\Psi}_1]_{ij} &= \sigma_f \prod_{q}^{Q} \left[\tilde{\sigma}_{wqii}\boldsymbol{x}_i\boldsymbol{x}_i^\mathsf{T} + 1\right]^{-\frac{1}{2}} \exp\left[-\frac{\left(\boldsymbol{x}_i\tilde{\boldsymbol{\mu}}_{wq:i}^\mathsf{T} - z_{jq}\right)^2}{2\tilde{\sigma}_{wqii}\boldsymbol{x}_i\boldsymbol{x}_i^\mathsf{T} + 2}\right] \\
[\boldsymbol{\Psi}_2]_{jk} &= \sigma_f^2 \sum_{i}^{n} \prod_{q}^{Q} \left[2\tilde{\sigma}_{wqii}\boldsymbol{x}_i\boldsymbol{x}_i^\mathsf{T} + 1\right]^{-\frac{1}{2}} \exp\left[-\frac{\left(\boldsymbol{x}_i\tilde{\boldsymbol{\mu}}_{wq:i}^\mathsf{T} - \bar{\mathfrak{z}}_{jkq}\right)^2}{2\tilde{\sigma}_{wqii}\boldsymbol{x}_i\boldsymbol{x}_i^\mathsf{T} + 1} - \frac{\left(\mathfrak{z}_{jq} - \mathfrak{z}_{kq}\right)^2}{4}\right]; \\
\bar{\boldsymbol{\mathfrak{z}}}_{jk} &= \frac{\boldsymbol{\mathfrak{z}}_j + \boldsymbol{\mathfrak{z}}_k}{2}.
\end{aligned}$$

This variational distribution has $(m_uD + m_vQ + 2m_vQD)$ parameters in total: $m_u \cdot D$ parameters for $\mathfrak{Z}$, $m_v \cdot Q$ parameters for Ξ, $m_v \cdot Q \cdot D$ for $\check{\boldsymbol{M}}_v$, and $m_v \cdot Q \cdot D$ for $\check{\boldsymbol{\Sigma}}$. This brings the total parameter and hyperparameter count to a total of $(Q + QD + m_u D + m_v Q + 2m_v QD)$ when including the variances and lengthscales of each of the Q RBF kernels.

With the variational distribution defined, the standard derivation of the evidence lower bound can be carried out. By applying Jensen's inequality to $p(\boldsymbol{y} \mid \boldsymbol{X}, \Xi, \mathfrak{Z})$, I derived the following ELBO:

$$\begin{aligned} p(\boldsymbol{y} \mid \boldsymbol{X}, \Xi, \mathfrak{Z}) \geq & -\frac{n}{2}\log(2\pi) - \frac{n-m_u}{2}\log\left(\sigma^2\right) - \frac{1}{2\sigma^2}\psi_0 \\ & + \frac{1}{2}\log\left(|\sigma^2\boldsymbol{K}_u + \boldsymbol{\Psi}_2|\right) + \log(|\boldsymbol{K}_u|) + \frac{1}{2\sigma^2}\mathrm{Tr}\left(\boldsymbol{K}_u^{-1}\boldsymbol{\Psi}_2\right) \\ & - \frac{1}{2\sigma^2}\boldsymbol{y}\boldsymbol{y}^{\mathsf{T}} + \frac{1}{2\sigma^2}\boldsymbol{y}\boldsymbol{\Psi}_1\left(\sigma^2\boldsymbol{K}_u + \boldsymbol{\Psi}_2\right)^{-1}\boldsymbol{\Psi}_1^{\mathsf{T}}\boldsymbol{y}^{\mathsf{T}} \\ & - \sum_{q,d}^{Q,D}\mathrm{KL}(q(\boldsymbol{v}_{qd}) \parallel p(\boldsymbol{v}_{qd})), \end{aligned}$$

where

$$\psi_0 = n\sigma_f^2.$$

Using this closed-form ELBO, the joint optimization of kernel hyperparameters and variational parameters can be performed. At last, I describe the predictive distribution $q(y_* \mid \boldsymbol{x}_*, \boldsymbol{y}, \boldsymbol{X}, \Xi, \mathfrak{Z})$. This distribution is not Gaussian; nevertheless, as Girard *et al.* (2003) did, the predictive mean and variance can be analytically derived. The predictive distribution has the following form:

$$q(y_* \mid \boldsymbol{x}_*, \boldsymbol{y}, \boldsymbol{X}, \Xi, \mathfrak{Z}) \approx \mathcal{N}\left(y_* \mid \mu_*, \sigma_*^2\right),$$

where:

$$\begin{aligned} \boldsymbol{C} &= \left(\sigma^2\boldsymbol{K}_u + \boldsymbol{\Psi}_2\right)^{-1}; \\ \mu_* &= \boldsymbol{y}\boldsymbol{\Psi}_1\boldsymbol{C}\boldsymbol{\psi}_{1*}^{\mathsf{T}}; \\ \sigma_*^2 &= \sigma_f^2 + \sigma^2 - \mu_*^2 + \mathrm{Tr}\left(\left[\sigma^2\boldsymbol{C} + \boldsymbol{C}\boldsymbol{\Psi}_1\boldsymbol{y}\boldsymbol{y}^{\mathsf{T}}\boldsymbol{\Psi}_1^{\mathsf{T}}\boldsymbol{C} - \boldsymbol{K}_u^{-1}\right]\boldsymbol{\Psi}_{2*}\right) + \sigma^2. \end{aligned}$$

and $\boldsymbol{\psi}_{1*}$ and $\boldsymbol{\Psi}_{2*}$ are computed like $\boldsymbol{\Psi}_1$ and $\boldsymbol{\Psi}_2$, but replacing $\boldsymbol{X}$ with $\boldsymbol{x}_*$.

4.4 Experiments

To validate that deep variational Mahalanobis Gaussian process (DVMGP) accurately captures non-smooth behavior and has competitive results with other GP models, I performed a regression experiment in a synthetic dataset with input-dependent linear projections and another experiment in varied empirical datasets. In this section, I examine the two-layer

DVMGP against a two-layer DS-DGP and the shallow sparse Gaussian process for regression (SGPR). These models are evaluated on average negative log predictive density (NLPD), mean relative absolute error (MRAE), and root mean squared error (RMSE). All experiments in this section were implemented in Python using the GPflow framework (MATTHEWS *et al.*, 2017). The code is available at <https://github.com/spectraldani/DeepMahaGP>.

The average NLPD is given by $\frac{1}{2}\log 2\pi + \frac{1}{2n^*}\sum_i^{n^*}\left[\log \sigma_i^{*2} + \frac{(y_i - {\mu^*}_i)^2}{\sigma_i^{*2}}\right]$, where n^* is the number of test samples, y_i is the true output, ${\mu^*}_i$ is the predicted mean output, and σ_i^{*2} is the i-th predicted variance. The MRAE is given by $\frac{1}{n^*}\sum_i^{n^*}\frac{\|y_i - {\mu^*}_i\|}{\|y_i\|}$. And finally, the RMSE, given by $\sqrt{\frac{1}{n^*}\sum_i^{n^*}(y_i - {\mu^*}_i)^2}$. All metrics are “the lower, the better”.

In both experiments, training and test data have been Z-normalized based on the mean and variance of the training dataset, and Adam was used to maximize the ELBO (or evidence) of each model. For 2000 iterations, all parameters except noise variance are trained, and for the next 5000 iterations, all parameters are trained.

The noise variance was initialized at 0.1, and the other parameters were initialized as follows:

- **SGPR**, Uses 50 inducing points, $\mathfrak{Z}$ was obtained from the K-means of the training set, and ARD-RBF kernel’s variance and lengthscales were all set to 1;
- **DVMGP**, $Q = D$, 50 inducing points for $\boldsymbol{f}$ and 25 inducing points for $\boldsymbol{W}$, $\check{\boldsymbol{M}}_{v::i}$ is equal to the PCA of the training data and $\check{\sigma}_{vqi} = \frac{1.001}{D}$ for every i and q, $\mathfrak{Z}$ was obtained from the K-means of projection of the training data through $\check{\boldsymbol{M}}_v$ and $\boldsymbol{\Xi}$ was obtained from the K-means of $\boldsymbol{X}$;
- **DGP**, Follows the initialization detailed in (SALIMBENI; DEISENROTH, 2017a).

4.4.1 Synthetic experiment

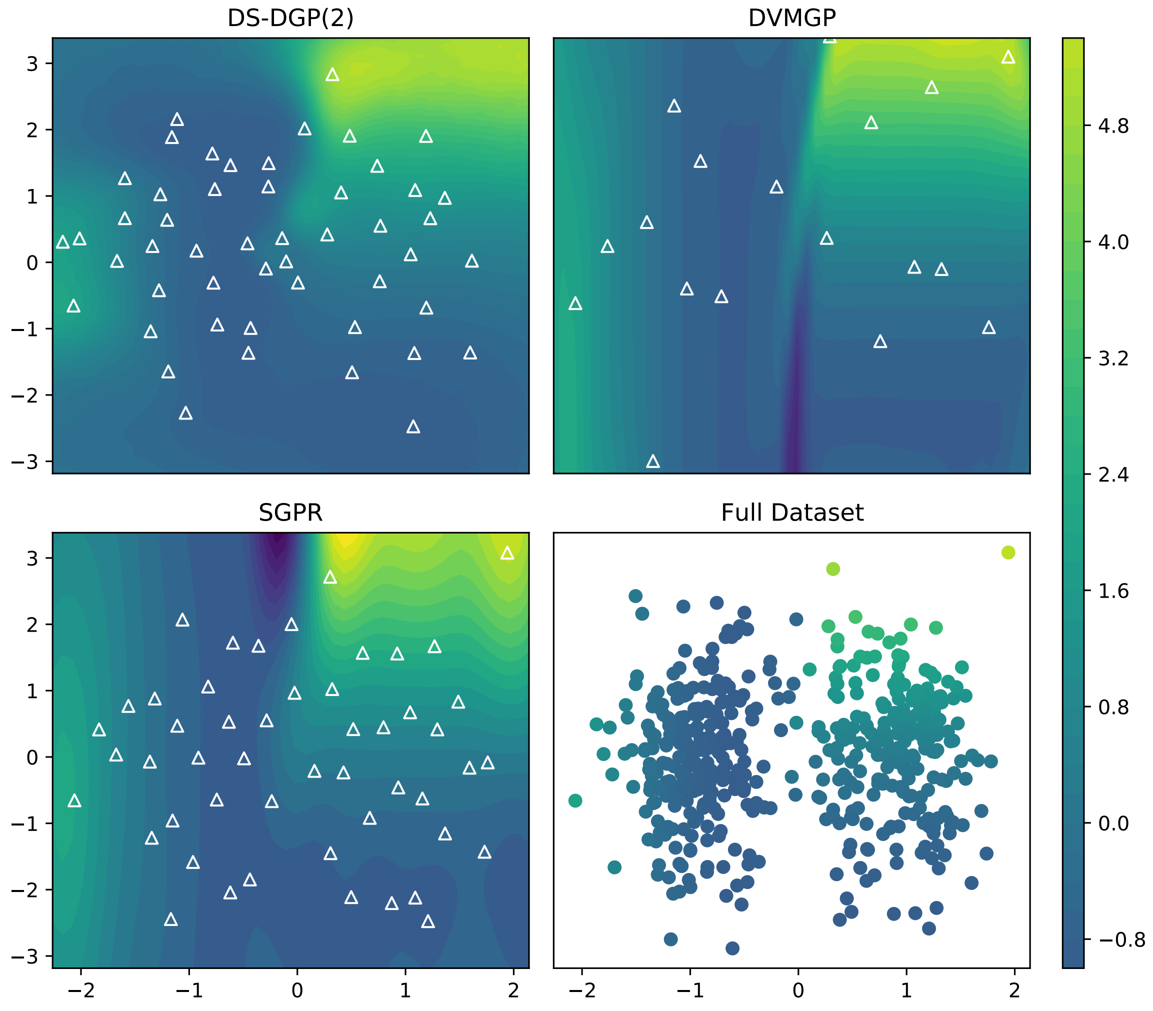
The dataset for this experimented was generated by considering two 2D Gaussian distributed clusters $\boldsymbol{C}_0$ and $\boldsymbol{C}_1$, each with 250 elements. The first cluster has outputs $\boldsymbol{o}_0 = \boldsymbol{c}_{0:0}^2 + \boldsymbol{c}_{0:0}$, and the second cluster has outputs $\boldsymbol{o}_1 = \boldsymbol{c}_{1:0}^2 + \boldsymbol{c}_{1:0}$. This dataset clearly has an input-varying dimensional reduction. Then, the dataset $\{\boldsymbol{C}, \boldsymbol{O}\}$ is shuffled and split into a training dataset $\{\boldsymbol{X}, \boldsymbol{y}\}$ with 250 entries and a test dataset $\{\boldsymbol{x}_*, \boldsymbol{y}_*\}$ with 250 entries.

Table 4 displays the results of each model evaluated on the test data. While Figure 6 displays the mean of the function that each model learned through a filled contour plot, each color represents a value of y.

Table 4 – Evaluation of the models on the synthetic dataset.

	NLPD	MRAE	RMSE
SGPR	0.041	0.188	0.215
DS-DGP(2)	-0.181	0.318	0.297
DVMGP	-1.883	0.040	0.129

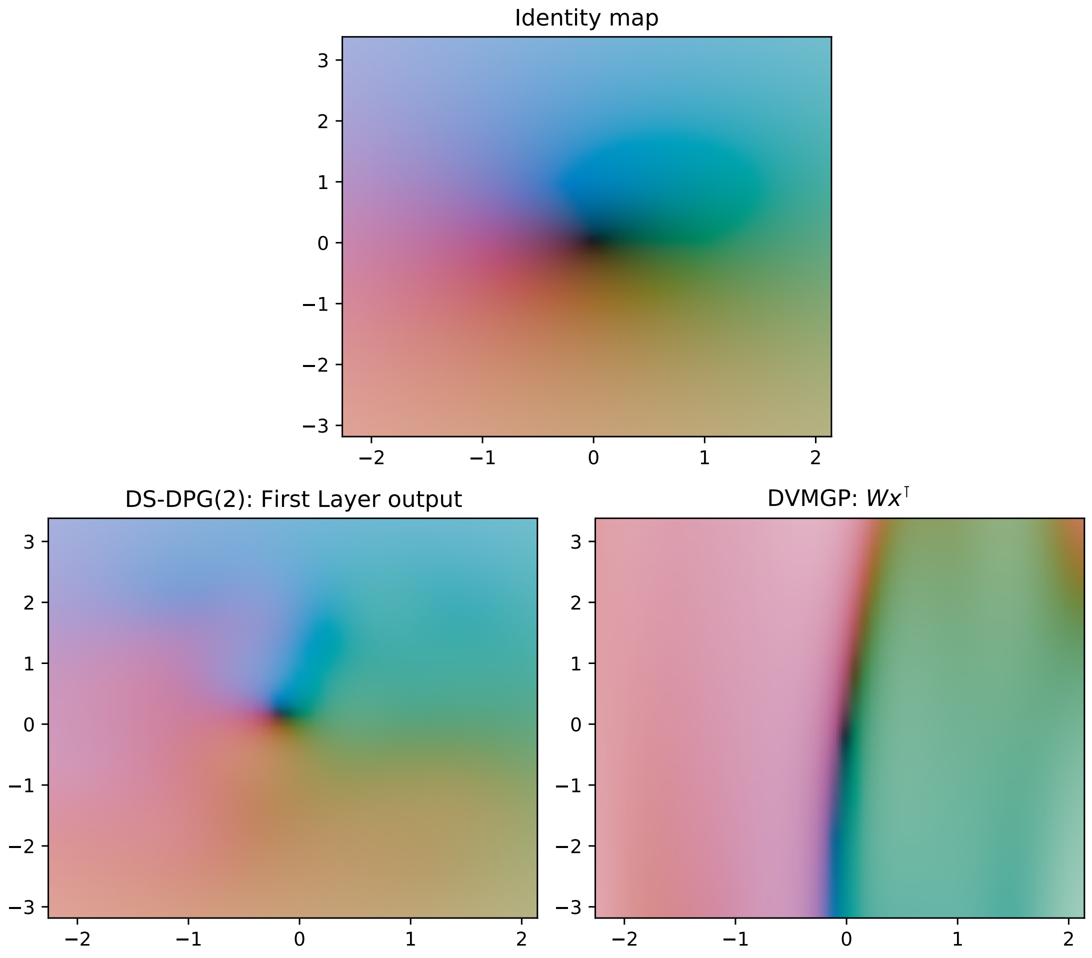
Figure 6 – The mean of each model's predicted value in the domain represented as color alongside a scatter plot of the entire dataset. Empty triangles with white outlines are the pseudo-inputs of the first layer of each model.



To further explore the differences between DS-DGP and DVMGP, I present a quick analysis of the behavior of the first layer of each model. First, I plotted the input space against the input of each model's output layer. Because the input dimension is two and the hidden layer's output dimension is also two, the mapping done by the hidden layer was represented through a domain colored plot in Figure 7.

The output in a domain colored plot is represented by color over the input point. The output point's angle with the origin determines the hue of the color, and the distance from the

Figure 7 – Domain colored plots of the second layer's input with respect to the first layer's input. The plot of the identity function is shown as a reference.



origin determines its brightness. The uppermost plot in Figure 7 is just the identity map. This plot serves as a reference to the original position of each color. By comparing the bottom plots with the top plot, the mapping of the input space through the hidden layer can be perceived.

The distinction between DS-DGP and DVMGP is strikingly clear. DVMGP obtained a sharp separation between the clusters. However, in DS-DGP's plot, most hues are still present in their starting place, and the brightness progression is still from the center-out. In other words, the first layer has not significantly changed the input dimension. The dominance of two tones in DVMGP's plot indicates that the output layer's inputs live in a thin band that passes through the origin.

This distinction can be further investigated through the values related to each model's hidden layers' relevant dimensions. For DS-DGP, the variable to look for is the inverse of the output layer's kernel lengthscales, as it shows how sensitive the output layer is to the output of the hidden layer. In DVMGP, the corresponding variable is the kernel variance of each hidden layer's row, since the larger this value is, the more distant from zero the values of that row are.

Figure 8 – A plot of the values that show the relevance of each latent dimension of the synthetic dataset for DS-DGP and DVMGP. On each graph, the higher the bars are, the more relevant that dimension is.

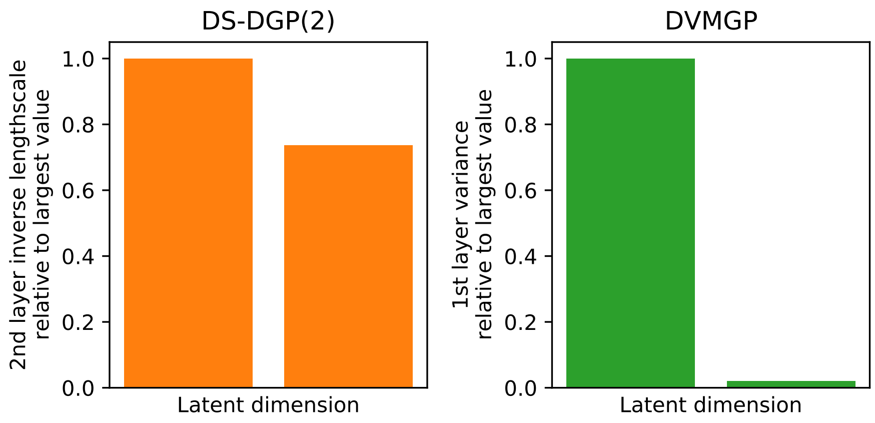


Figure 9 – Test negative log-likelihood of each model on each of the datasets. Each dot represents the result of a fold, and the cross is the mean of all folds.

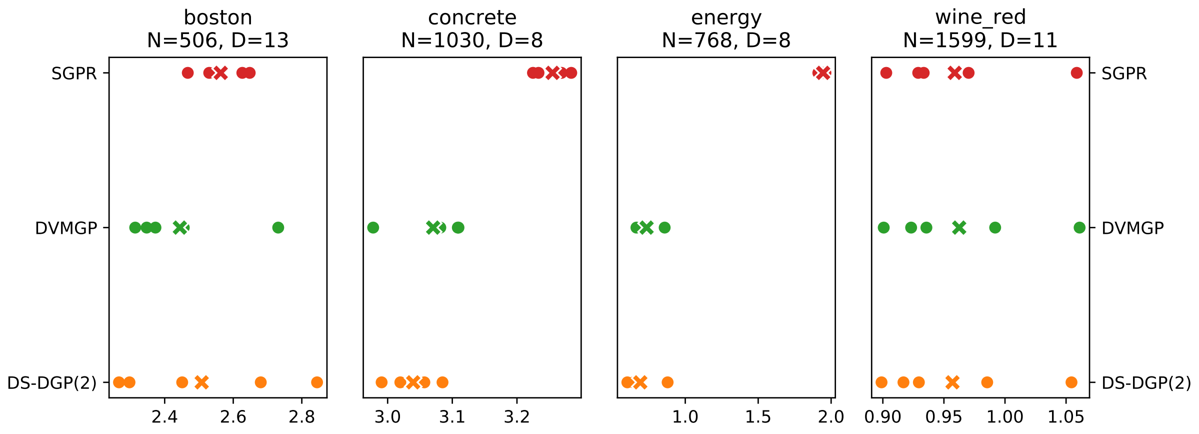


Figure 8 shows a plot of these values for each model. As suspected, it shows that DVMGP gives greater importance to just one of the two latent dimensions.

4.4.2 Empirical datasets

In this section, I compare the DVMGP model with the other models on well-known regression datasets. To assess each model fairly, I have chosen to adopt a five-fold separation of the datasets into training and test. Briefly, each dataset is shuffled and split into five divisions where each part is chosen to be the test dataset while all the others are combined into a training dataset. This process results in five sets of test metrics. Figure 9 shows a brief overview of the datasets and the test NLPD of each model.

As seen in Figure 9, all models better or equal to the single-layer model. As reinforced by the other metrics in Table 5, DVMGP holds comparable to DS-DGP. However, as

Figure 10 – A plot of the mean and 1σ interval of the values that correspond to the relevance of each latent dimension.

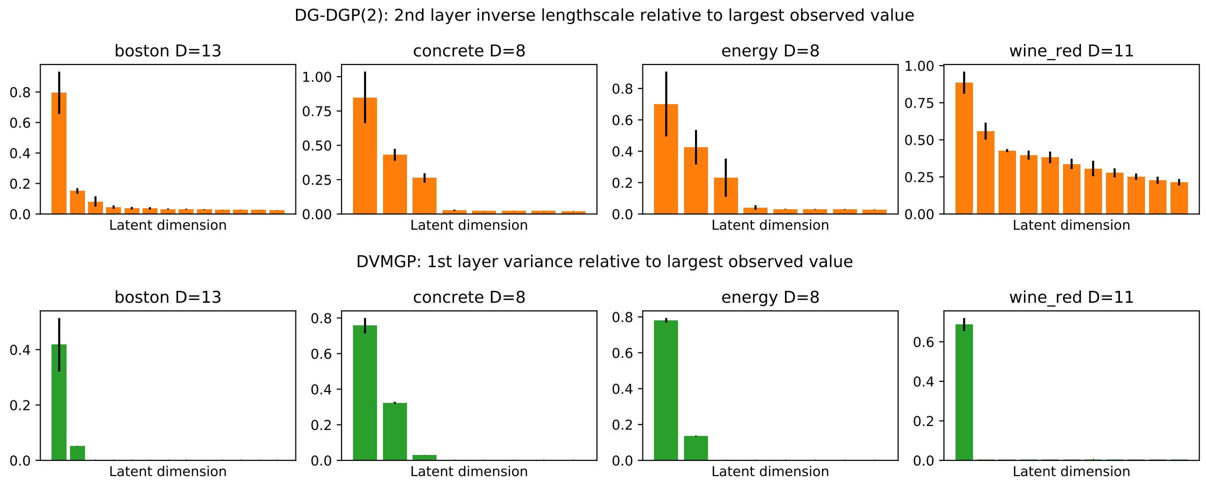
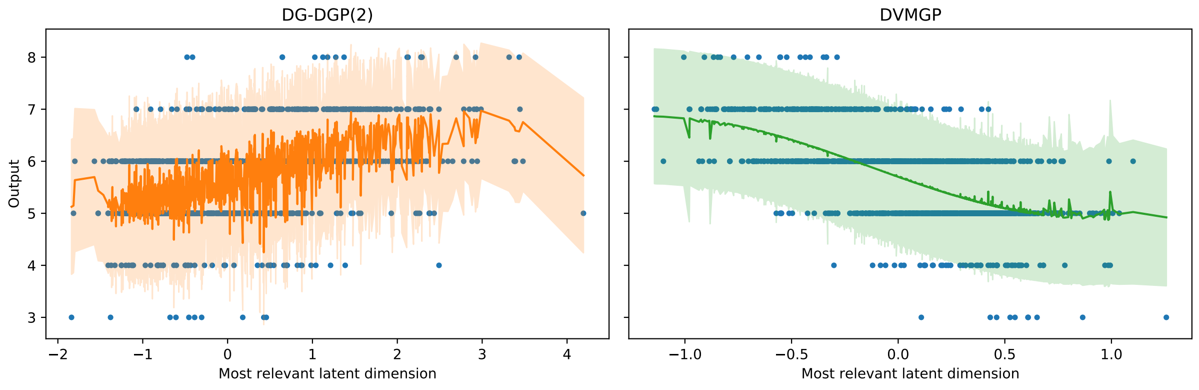


Figure 11 – A plot of the predicted output and observed output versus the most relevant dimension of input of each model's last layer on the wine_red dataset. The shaded area represents the 2σ interval centered on the predicted mean.



seen in the previous section, ease to project the input space into smaller dimensions is one of the main features of modeling behind DVMGP. Figure 10 displays the same set of variables that correlate with the latent dimension's relevance. Despite the similar performance, a sharper division between dimensions can be seen in DVMGP's results.

The most significant discrepancy within the sensitivity of the latent dimensions presented in Figure 10 occurs within the wine_red dataset. In that dataset, DVMGP assigned most of the relevance to one latent dimension, but DS-DGP assigned roughly the same order of magnitude of significance to all of them. To better visualize the difference, I plotted each model's most significant hidden dimension against their predicted outputs in Figure 11. Since both models' last layer learns smooth functions, one expects that the functions in the figure would be as smooth as possible if they depend only on their most relevant dimension.

As expected, DS-DGP's function is more jagged than DVMGP's. However, the

Table 5 – Mean and standard deviation of all tested metrics on the UCI datasets.

Dataset	Model	NLPD	MRAE	RMSE
boston	DS-DGP(2)	2.508 ± 0.249	0.108 ± 0.018	2.508 ± 0.249
	DVMGP	2.444 ± 0.169	0.109 ± 0.017	2.444 ± 0.169
	SGPR	2.563 ± 0.074	0.112 ± 0.014	2.563 ± 0.074
concrete	DS-DGP(2)	3.040 ± 0.036	0.124 ± 0.010	3.040 ± 0.036
	DVMGP	3.070 ± 0.054	0.127 ± 0.007	3.070 ± 0.054
	SGPR	3.255 ± 0.025	0.167 ± 0.013	3.255 ± 0.025
energy	DS-DGP(2)	0.692 ± 0.108	0.016 ± 0.001	0.692 ± 0.108
	DVMGP	0.735 ± 0.073	0.017 ± 0.002	0.735 ± 0.073
	SGPR	1.945 ± 0.019	0.067 ± 0.005	1.945 ± 0.019
wine_red	DS-DGP(2)	0.957 ± 0.063	0.090 ± 0.008	0.957 ± 0.063
	DVMGP	0.963 ± 0.065	0.090 ± 0.008	0.963 ± 0.065
	SGPR	0.959 ± 0.061	0.090 ± 0.008	0.959 ± 0.061

function learned by DVMGP is not perfectly smooth when considering just the most relevant dimension, which indicates that some extra-dimensional data still correlates with the output but not as strongly as in DS-DGP.

4.5 Conclusion

In this chapter, I presented an alternative deep Gaussian process based on composing a GP's output into the lengthscales of the other's kernel. This alternative builds upon a previous variational model by Titsias and Lázaro-Gredilla (2013), thus, distinguishing itself from other models based on composition through kernel lengthscales, which uses Monte Carlo methods for their inference. Using the variational inference procedure with inducing variables first developed by Titsias (2009), many of the modern applications of Gaussian processes to big data or scalable inference can be developed for this new model.

By evaluating the model in synthetic and empirical datasets, I showed that this new model is either comparable or better to ordinary Deep Gaussian Processes, especially in tasks that require learning projections of the input data. This model is not susceptible to the pathological behavior described by Duvenaud *et al.* (2014). It also has the advantage that the quadratic form inside the kernel still defines an inner product space, contrasting with models based on Paciorek (2003).

However, the presented model may suffer from some drawbacks related to optimization and expressivity. When using the Mahalanobis kernel for dimensionality reduction, each layer hidden layer has to have $Q \times D$ Gaussian processes, which means that the number of variational parameters may become very large, slowing down optimization. Despite being a deep Gaussian process by the composition of kernel functions, this model's theoretical limits, as explored by Dunlop *et al.* (2018) for other constructions, are not well understood.

5 CONCLUDING REMARKS

This dissertation focused on two issues on the use of Gaussian process models with latent projections. I presented in Chapter 1 the Gaussian process latent variable model (GPLVM) and the deep Gaussian process (DGP) with some of the current gaps present in their structure and inference. Briefly, this dissertation's objectives were to explore deterministic variational inference on Bayesian GPLVM models with arbitrary kernels and DGP from the composition by kernel functions.

In Chapter 3, I presented the unscented Gaussian process latent variable model. This extension to Bayesian GPLVM uses the unscented transformation (UT) to enable deterministic variational inference when using arbitrary kernel functions and their combinations. This new methodology was tested in dimensionality reduction and dynamical free simulation tasks against other approximations that used Gauss-Hermite (GH) quadrature and Monte Carlo (MC) integration. On both tasks, the unscented GPLVM had competitive results while requiring small amounts of samples.

In Chapter 4, I presented the deep Mahalanobis Gaussian process. This deep Gaussian process builds upon the model by Titsias and Lázaro-Gredilla (2013) and is inspired by the DGP by kernel function construction defined by Dunlop *et al.* (2018). In this deep model, each layer defines an input-dependent projection matrix for the next layer. This construction is equivalent to a layer driving the following layer's lengthscales but enables a natural interpretation through space embedding. This model was then compared to a compositional DGP by Salimbeni and Deisenroth (2017a) in synthetic and empirical regression datasets. In these datasets, the DMGP model had equivalent or better results than DS-DGP while discovering better latent space projections of data.

In conclusion, the objectives set in Chapter 1 were met; both propositions shown promising results despite some of their shortcomings. However, investigations on the theoretical guarantees of these models still need to be done. Specifically, the relation of the error guarantees of UT approximation and the inference in GPLVM models still needs to be explored, and also, how the limitations of DGP models presented by Dunlop *et al.* (2018) apply to the DMGP model is still not clear.

For future work related to unscented GPLVM, I aim to evaluate how other methods of obtaining sigma points might increase or decrease the quality of the approximations taken. Also, I intend to assess the UT in more scenarios where inference with GP models falls into intractable integrals. For instance, we wish to tackle intractable expressions that arise with DGP that have intractable inference due to low-dimensional intractable integrals like the doubly stochastic Gaussian process by Salimbeni and Deisenroth (2017a) and recurrent Gaussian processes by Mattos *et al.* (2016). I also suspect that further extensions to DMGP models that use other kernels, like the Matérn family of kernels, may cause some $\boldsymbol{\Psi}$-statistics to become intractable. Still, it may be possible to reduce them to low-dimensional integrals.

Building upon DMGP, I wish to expand the current variational inference method to enable mini-batching during training as done by Hensman *et al.* (2013). I also aim to test this model in manifold learning settings against well-known methods like UMAP (MCINNES *et al.*, 2018). Alongside the aforementioned theoretical exploration, more empirical validation of DMGP models with extra layers is also needed.

BIBLIOGRAPHY

AL-SHEDIVAT, M.; WILSON, A. G.; SAATCHI, Y.; HU, Z.; XING, E. P. Learning scalable deep kernels with recurrent structure. **Journal of Machine Learning Research**, v. 18, no. 82, pp. 1–37, 2017. ISSN 1532-4435.

ANGER, C.; SCHRADER, R.; KLINGAUF, U. Unscented Kalman filter with Gaussian process degradation model for bearing fault prognosis. In: BREGON, A.; SAXENA, A. (Ed.). **Proceedings of First European Conference of the Prognostics and Health Management Society**. Dresden, Germany: PHM society, 2012. v. 3, no. 1, pp. 202–213.

BISHOP, C. M. **Pattern Recognition and Machine Learning**. New York, NY, USA: Springer-Verlag New York Inc., 2006. ISBN 9781493938438.

BISHOP, C. M.; JAMES, G. D. Analysis of multiphase flows using dual-energy gamma densitometry and neural networks. **Nuclear Instruments and Methods in Physics Research Section A: Accelerators, Spectrometers, Detectors and Associated Equipment**, Elsevier BV, v. 327, no. 2-3, pp. 580–593, apr. 1993.

BLEI, D. M.; KUCUKELBIR, A.; MCAULIFFE, J. D. Variational inference: a review for statisticians. **Journal of the American Statistical Association**, Taylor & Francis, v. 112, no. 518, pp. 859–877, 2017.

BYRD, R. H.; LU, P.; NOCEDAL, J.; ZHU, C. A limited memory algorithm for bound constrained optimization. **SIAM Journal on Scientific Computing**, Society for Industrial & Applied Mathematics (SIAM), v. 16, no. 5, pp. 1190–1208, sep. 1995.

CALANDRA, R.; PETERS, J.; RASMUSSEN, C. E.; DEISENROTH, M. P. Manifold Gaussian processes for regression. In: IEEE. **2016 International Joint Conference on Neural Networks**. Vancouver, BC, Canada: IEEE, 2016. pp. 3338–3345.

CAYTON, L. **Algorithms for manifold learning**. San Diego, CA, USA, 2005. Available in: http://www.lcayton.com/resexam.pdf. Accessed on: 10 mar. 2020.

DAMIANOU, A.; LAWRENCE, N. Deep Gaussian processes. In: CARVALHO, C. M.; RAVIKUMAR, P. (Ed.). **Proceedings of the 16th International Conference on Artificial Intelligence and Statistics**. Scottsdale, AZ, USA: PMLR, 2013. (Proceedings of Machine Learning Research, v. 31), pp. 207–215.

DAMIANOU, A. C.; TITSIAS, M. K.; LAWRENCE, N. D. Variational inference for latent variables and uncertain inputs in Gaussian processes. **Journal of Machine Learning Research**, v. 17, no. 42, pp. 1–62, 2016.

DUNLOP, M. M.; GIROLAMI, M. A.; STUART, A. M.; TECKENTRUP, A. L. How deep are deep Gaussian processes? **Journal of Machine Learning Research**, v. 19, no. 54, pp. 1–46, 2018.

DUVENAUD, D.; LLOYD, J.; GROSSE, R.; TENENBAUM, J.; ZOUBIN, G. Structure discovery in nonparametric regression through compositional kernel search. In: DASGUPTA, S.; MCALLESTER, D. (Ed.). **Proceedings of the 30th International Conference on Machine Learning**. Atlanta, GA, USA: PMLR, 2013. (Proceedings of Machine Learning Research, 3), pp. 1166–1174.

DUVENAUD, D.; RIPPEL, O.; ADAMS, R.; GHAHRAMANI, Z. Avoiding pathologies in very deep networks. In: KASKI, S.; CORANDER, J. (Ed.). **Proceedings of the 17th International Conference on Artificial Intelligence and Statistics**. Reykjavik, Iceland: PMLR, 2014. (Proceedings of Machine Learning Research, v. 33), pp. 202–210.

ELEFTHERIADIS, S.; NICHOLSON, T.; DEISENROTH, M.; HENSMAN, J. Identification of Gaussian process state space models. In: GUYON, I.; LUXBURG, U. V.; BENGIO, S.; WALLACH, H.; FERGUS, R.; VISHWANATHAN, S.; GARNETT, R. (Ed.). **Advances in Neural Information Processing Systems**. [*S.l.*]: Curran Associates, Inc., 2017. v. 30, pp. 5309–5319.

FEFFERMAN, C.; MITTER, S.; NARAYANAN, H. Testing the manifold hypothesis. **Journal of the American Mathematical Society**, American Mathematical Society (AMS), v. 29, no. 4, pp. 983–1049, feb. 2016.

GIBBS, M. N. **Bayesian Gaussian processes for regression and classification**. Thesis (Ph.D.) — University of Cambridge, 1997.

GIRARD, A.; RASMUSSEN, C.; CANDELA, J. Q. n.; MURRAY-SMITH, R. Gaussian process priors with uncertain inputs application to multiple-step ahead time series forecasting. In: BECKER, S.; THRUN, S.; OBERMAYER, K. (Ed.). **Advances in Neural Information Processing Systems**. [*S.l.*]: MIT Press, 2003. v. 15, pp. 545–552.

HENSMAN, J.; FUSI, N.; LAWRENCE, N. D. Gaussian processes for big data. In: **Proceedings of the 29h Conference on Uncertainty in Artificial Intelligence**. Arlington, VA, USA: AUAI Press, 2013. (UAI'13), pp. 282–290.

HENSMAN, J.; MATTHEWS, A. G.; FILIPPONE, M.; GHAHRAMANI, Z. MCMC for variationally sparse Gaussian processes. In: CORTES, C.; LAWRENCE, N.; LEE, D.; SUGIYAMA, M.; GARNETT, R. (Ed.). **Advances in Neural Information Processing Systems**. [*S.l.*]: Curran Associates, Inc., 2015. v. 28, pp. 1648–1656.

HONKELA, A. Approximating nonlinear transformations of probability distributions for nonlinear independent component analysis. In: IEEE. **2004 IEEE International Joint Conference on Neural Networks**. Budapest, Hungary: IEEE, 2004. v. 3, pp. 2169–2174.

HYNDMAN, R.; YANG, Y. **Time Series Data Library**. 2018. Available in: https://pkg.yangzhuoranyang.com/tsdl/. Accessed on: 10 mar. 2020.

JORDAN, M. I.; GHAHRAMANI, Z.; JAAKKOLA, T. S.; SAUL, L. K. An introduction to variational methods for graphical models. **Machine Learning**, Springer, v. 37, no. 2, pp. 183–233, 1999.

JULIER, S. J.; UHLMANN, J. K. Unscented filtering and nonlinear estimation. **Proceedings of the IEEE**, IEEE, v. 92, no. 3, pp. 401–422, mar. 2004.

KINGMA, D. P.; WELLING, M. Auto-encoding variational Bayes. In: BENGIO, Y.; LECUN, Y. (Ed.). **2nd International Conference on Learning Representations**. Banff, AB, Canada: [*s.n.*], 2014.

KO, J.; FOX, D. GP-BayesFilters: Bayesian filtering using Gaussian process prediction and observation models. **Autonomous Robots**, Springer, v. 27, no. 1, pp. 75–90, may. 2009.

KO, J.; FOX, D. Learning GP-BayesFilters via Gaussian process latent variable models. **Autonomous Robots**, Springer, v. 30, no. 1, pp. 3–23, oct. 2010.

KO, J.; KLEIN, D. J.; FOX, D.; HAEHNEL, D. GP-UKF: Unscented Kalman filters with Gaussian process prediction and observation models. In: IEEE. **Proceedings of the 2007 IEEE/RSJ International Conference on Intelligent Robots and Systems**. San Diego, CA, USA: IEEE, 2007. v. 1, pp. 1901–1907.

KOCIJAN, J.; GIRARD, A.; BANKO, B.; MURRAY-SMITH, R. Dynamic systems identification with Gaussian processes. **Mathematical and Computer Modelling of Dynamical Systems**, Informa UK Limited, v. 11, no. 4, pp. 411–424, dec. 2005.

LAWRENCE, N. Gaussian process latent variable models for visualisation of high dimensional data. In: THRUN, S.; SAUL, L.; SCHöLKOPF, B. (Ed.). **Advances in Neural Information Processing Systems**. [*S.l.*]: MIT Press, 2004. v. 16, pp. 329–336.

LAWRENCE, N. D.; MOORE, A. J. Hierarchical Gaussian process latent variable models. In: **Proceedings of the 24th international conference on Machine learning - ICML '07**. New York, NY, USA: ACM Press, 2007. (ICML ’07), pp. 481–488. ISBN 9781595937933.

LáZARO-GREDILLA, M. Bayesian warped Gaussian processes. In: PEREIRA, F.; BURGES, C. J. C.; BOTTOU, L.; WEINBERGER, K. Q. (Ed.). **Advances in Neural Information Processing Systems**. [*S.l.*]: Curran Associates, Inc., 2012. v. 25, pp. 1619–1627.

LECUN, Y.; BENGIO, Y.; HINTON, G. Deep learning. **Nature**, v. 521, no. 7553, pp. 436–444, 2015. ISSN 1476-4687.

LLOYD, J. R.; DUVENAUD, D.; GROSSE, R.; TENENBAUM, J. B.; GHAHRAMANI, Z. Automatic construction and natural-language description of nonparametric regression models. In: **Proceedings of the 28th AAAI Conference on Artificial Intelligence**. Québec City, Québec, Canada: AAAI Press, 2014. (AAAI’14), pp. 1242–1250.

MACKAY, D. J. C. Introduction to Gaussian processes. In: BISHOP, C. M. (Ed.). **Neural Networks and Machine Learning**. Berlin, Germany: Springer, 1998, (NATO ASI Series, v. 168). pp. 133–166.

MATTHEWS, A. G. de G.; WILK, M. van der; NICKSON, T.; FUJII, K.; BOUKOUVALAS, A.; LEÓN-VILLAGRÁ, P.; GHAHRAMANI, Z.; HENSMAN, J. GPflow: A gaussian process library using TensorFlow. **Journal of Machine Learning Research**, v. 18, no. 40, pp. 1–6, 2017.

MATTOS, C. L. C.; DAI, Z.; DAMIANOU, A. C.; FORTH, J.; BARRETO, G. A.; LAWRENCE, N. D. Recurrent Gaussian processes. In: BENGIO, Y.; LECUN, Y. (Ed.). **4th International Conference on Learning Representations**. Caribe Hilton, San Juan, Puerto Rico: [*s.n.*], 2016.

MCINNES, L.; HEALY, J.; MELVILLE, J. UMAP: Uniform manifold approximation and projection for dimension reduction. feb. 2018.

MEEDS, E.; OSINDERO, S. An alternative infinite mixture of Gaussian process experts. In: WEISS, Y.; SCHöLKOPF, B.; PLATT, J. (Ed.). **Advances in Neural Information Processing Systems**. [*S.l.*]: MIT Press, 2006. v. 18, pp. 883–890.

MENEGAZ, H. M. T.; ISHIHARA, J. Y.; BORGES, G. A.; VARGAS, A. N. A systematization of the unscented Kalman filter theory. **IEEE Transactions on Automatic Control**, IEEE, v. 60, no. 10, pp. 2583–2598, oct. 2015.

MURRAY, I.; ADAMS, R. P. Slice sampling covariance hyperparameters of latent Gaussian models. In: LAFFERTY, J.; WILLIAMS, C.; SHAWE-TAYLOR, J.; ZEMEL, R.; CULOTTA, A. (Ed.). **Advances in Neural Information Processing Systems**. [*S.l.*]: Curran Associates, Inc., 2010. v. 23, pp. 1732–1740.

NGUYEN, T.; BONILLA, E. Efficient variational inference for Gaussian process regression networks. In: CARVALHO, C. M.; RAVIKUMAR, P. (Ed.). **Proceedings of the 16th International Conference on Artificial Intelligence and Statistics**. Scottsdale, Arizona, USA: PMLR, 2013. (Proceedings of Machine Learning Research, v. 31), pp. 472–480.

PACIOREK, C.; SCHERVISH, M. Nonstationary covariance functions for Gaussian process regression. In: THRUN, S.; SAUL, L.; SCHöLKOPF, B. (Ed.). **Advances in Neural Information Processing Systems**. [*S.l.*]: MIT Press, 2004. v. 16, pp. 273–280.

PACIOREK, C. J. **Nonstationary Gaussian Processes for Regression and Spatial Modelling**. Thesis (Ph.D.) — Carnegie Mellon University, may. 2003.

RASMUSSEN, C.; GHAHRAMANI, Z. Infinite mixtures of gaussian process experts. In: DIETTERICH, T.; BECKER, S.; GHAHRAMANI, Z. (Ed.). **Advances in Neural Information Processing Systems**. [*S.l.*]: MIT Press, 2002. v. 14, pp. 881–888.

RASMUSSEN, C.; WILLIAMS, C. **Gaussian Processes for Machine Learning**. 1. ed. Cambridge, MA, USA: MIT Press, 2006. ISBN 9780262182539.

REMES, S.; HEINONEN, M.; KASKI, S. Non-stationary spectral kernels. In: GUYON, I.; LUXBURG, U. V.; BENGIO, S.; WALLACH, H.; FERGUS, R.; VISHWANATHAN, S.; GARNETT, R. (Ed.). **Advances in Neural Information Processing Systems**. [*S.l.*]: Curran Associates, Inc., 2017. v. 30, pp. 4642–4651.

REZENDE, D. J.; MOHAMED, S.; WIERSTRA, D. Stochastic backpropagation and approximate inference in deep generative models. In: XING, E. P.; JEBARA, T. (Ed.). **Proceedings of the 31st International Conference on Machine Learning**. Bejing, China: PMLR, 2014. (Proceedings of Machine Learning Research, 2), pp. 1278–1286.

SAFARINEJADIAN, B.; KOWSARI, E. Fault detection in non-linear systems based on GP-EKF and GP-UKF algorithms. **Systems Science & Control Engineering**, Informa UK Limited, v. 2, no. 1, pp. 610–620, oct. 2014.

SALIMBENI, H.; DEISENROTH, M. Doubly stochastic variational inference for deep Gaussian processes. In: GUYON, I.; LUXBURG, U. V.; BENGIO, S.; WALLACH, H.; FERGUS, R.; VISHWANATHAN, S.; GARNETT, R. (Ed.). **Advances in Neural Information Processing Systems**. [*S.l.*]: Curran Associates, Inc., 2017. v. 30, pp. 4588–4599.

SALIMBENI, H.; DEISENROTH, M. P. Deeply non-stationary Gaussian processes. In: **Second workshop on Bayesian Deep Learning (NeurIPS 2017)**. [*S.l.*: *s.n.*], 2017.

STEINBERG, D. M.; BONILLA, E. V. Extended and unscented Gaussian processes. In: GHAHRAMANI, Z.; WELLING, M.; CORTES, C.; LAWRENCE, N.; WEINBERGER, K. Q.

(Ed.). **Advances in Neural Information Processing Systems**. [*S.l.*]: Curran Associates, Inc., 2014. v. 27, pp. 1251–1259.

TITSIAS, M. K. Variational learning of inducing variables in sparse Gaussian processes. In: DYK, D. van; WELLING, M. (Ed.). **Proceedings of the 12th International Conference on Artificial Intelligence and Statistics**. Hilton Clearwater Beach Resort, Clearwater Beach, Florida USA: PMLR, 2009. (Proceedings of Machine Learning Research, v. 5), pp. 567–574.

TITSIAS, M. K.; LAWRENCE, N. D. Bayesian Gaussian process latent variable model. In: TEH, Y. W.; TITTERINGTON, M. (Ed.). **Proceedings of the 13th International Conference on Artificial Intelligence and Statistics**. Chia Laguna Resort, Sardinia, Italy: PMLR, 2010. (Proceedings of Machine Learning Research, v. 9), pp. 844–851.

TITSIAS, M. K.; LáZARO-GREDILLA, M. Variational inference for Mahalanobis distance metrics in Gaussian process regression. In: BURGES, C. J. C.; BOTTOU, L.; WELLING, M.; GHAHRAMANI, Z.; WEINBERGER, K. Q. (Ed.). **Advances in Neural Information Processing Systems**. [*S.l.*]: Curran Associates, Inc., 2013. v. 26, pp. 279–287.

TITSIAS, M. K.; LÁZARO-GREDILLA, M. Doubly stochastic variational Bayes for non-conjugate inference. In: **Proceedings of the 31st International Conference on International Conference on Machine Learning**. Beijing, China: JMLR.org, 2014. (ICML'14, v. 32), pp. II–1971–II–1980.

UHLMANN, J. K. **Dynamic map building and localization: New theoretical foundations**. Thesis (Ph.D.) — University of Oxford, 1995.

WANG, Z.; KINUGAWA, J.; WANG, H.; KAZAHIRO, K. A human motion estimation method based on GP-UKF. In: IEEE. **2014 IEEE International Conference on Information and Automation**. Hailar, China: IEEE, 2014. pp. 1228–1232.

WILSON, A.; ADAMS, R. Gaussian process kernels for pattern discovery and extrapolation. In: DASGUPTA, S.; MCALLESTER, D. (Ed.). **Proceedings of the 30th International Conference on Machine Learning**. Atlanta, GA, USA: PMLR, 2013. (Proceedings of Machine Learning Research, 3), pp. 1067–1075.

WILSON, A.; KNOWLES, D.; GHAHRAMANI, Z. Gaussian process regression networks. In: LANGFORD, J.; PINEAU, J. (Ed.). **Proceedings of the 29th International Conference on Machine Learning (ICML-12)**. New York, NY, USA: Omnipress, 2012. (ICML '12), pp. 599–606. ISBN 978-1-4503-1285-1.

WILSON, A. G.; HU, Z.; SALAKHUTDINOV, R.; XING, E. P. Deep kernel learning. In: GRETTON, A.; ROBERT, C. C. (Ed.). **Proceedings of the 19th International Conference on Artificial Intelligence and Statistics**. Cadiz, Spain: PMLR, 2016. (Proceedings of Machine Learning Research, v. 51), pp. 370–378.

WILSON, A. G.; HU, Z.; SALAKHUTDINOV, R. R.; XING, E. P. Stochastic variational deep kernel learning. In: LEE, D.; SUGIYAMA, M.; LUXBURG, U.; GUYON, I.; GARNETT, R. (Ed.). **Advances in Neural Information Processing Systems**. [*S.l.*]: Curran Associates, Inc., 2016. v. 29, pp. 2586–2594.

ZHANG, W.; LIU, M.; ZHAO, Z. gui. Accuracy analysis of unscented transformation of several sampling strategies. In: IEEE. **2009 10th ACIS International Conference on**

Software Engineering, Artificial Intelligences, Networking and Parallel/Distributed Computing. Daegu, Korea: IEEE, 2009. pp. 377–380.

APPENDIX A – MATHEMATICAL DETAILS OF DEEP VARIATIONAL MAHALANOBIS GAUSSIAN PROCESS

This appendix is an extension of Section 4.3.1. In this appendix, I will show how to derive the evidence lower bound (ELBO) of deep Mahalanobis Gaussian process (DMGP) model for an arbitrary number of layers following the standard variational inference method (TITSIAS, 2009).

First, let us begin with the definition of the model. Given a training dataset $\boldsymbol{X} \in \mathbb{R}^{n\times D}$ and $\boldsymbol{y} \in \mathbb{R}^n$:

$$
\begin{aligned}
p(\boldsymbol{W} \mid \boldsymbol{X}) &= \prod_{q,d}^{Q,D} \mathcal{N}\left(\boldsymbol{W} \mid \boldsymbol{0}, \boldsymbol{K}_w^{(q)}\right);\\
p(\boldsymbol{s} \mid \boldsymbol{W}, \boldsymbol{X}) &= \mathcal{N}(\boldsymbol{s} \mid \boldsymbol{0}, \boldsymbol{K}_s);\\
\boldsymbol{f} &= \sigma_f \boldsymbol{s};\\
p(\boldsymbol{y} \mid \boldsymbol{f}) &= \mathcal{N}\left(\boldsymbol{y} \mid \boldsymbol{f}, \sigma^2 \boldsymbol{I}\right),
\end{aligned}
$$

where:

$$
\begin{aligned}
\left[\boldsymbol{K}_w^{(q)}\right]_{ij} &= \text{ARD-RBF}_q(\boldsymbol{x}_i, \boldsymbol{x}_j);\\
[\boldsymbol{K}_s]_{ij} &= \text{SRBF}(\boldsymbol{x}_i \boldsymbol{W}_i^\intercal, \boldsymbol{x}_j \boldsymbol{W}_j^\intercal).
\end{aligned}
$$

And Q is the dimensionality of the latent space. Now, I augment the model with pseudo-inputs $\boldsymbol{\Xi} \in \mathbb{R}^{m_v\times D}$, and $\mathfrak{Z} \in \mathbb{R}^{m_u\times Q}$ and inducing variables $\boldsymbol{V}$ and $\boldsymbol{u}$. The joint distribution of this model is:

$$
\begin{aligned}
p(\boldsymbol{y}, \boldsymbol{f}, \boldsymbol{W}, \boldsymbol{u}, \boldsymbol{V} \mid \boldsymbol{X}, \boldsymbol{\Xi}, \mathfrak{Z}) = \mathcal{N}\left(\boldsymbol{y} \mid \boldsymbol{f}, \sigma^2 \boldsymbol{I}\right) p(\boldsymbol{f} \mid \boldsymbol{u}, \boldsymbol{W}, \boldsymbol{X}) \mathcal{N}(\boldsymbol{u} \mid \boldsymbol{0}, \boldsymbol{K}_u)\\
\cdot \prod_{q,d}^{Q,D} p(\boldsymbol{w}_{qd} \mid \boldsymbol{v}_{qd}, \boldsymbol{X}) \mathcal{N}\left(\boldsymbol{v}_{qd} \,\middle|\, \boldsymbol{0}, \boldsymbol{K}_v^{(q)}\right),
\end{aligned}
$$

where:

$$
\begin{aligned}
p(\boldsymbol{w}_{qd} \mid \boldsymbol{v}_{qd}, \boldsymbol{X}) &= \mathcal{N}\left(\boldsymbol{w}_{qd} \,\middle|\, \boldsymbol{v}_{qd} \boldsymbol{K}_v^{(q)^{-1}} \boldsymbol{K}_{vw}^{(q)}, \boldsymbol{K}_w^{(q)} - \boldsymbol{K}_{wv}^{(q)} \boldsymbol{K}_v^{(q)^{-1}} \boldsymbol{K}_{vw}^{(q)}\right);\\
p(\boldsymbol{f} \mid \boldsymbol{u}, \boldsymbol{W}, \boldsymbol{X}) &= \mathcal{N}\left(\boldsymbol{f} \mid \boldsymbol{u} \boldsymbol{K}_u{}^{-1} \boldsymbol{K}_{uf}, \boldsymbol{K}_f - \boldsymbol{K}_{fu} \boldsymbol{K}_u{}^{-1} \boldsymbol{K}_{uf}\right);\\
\left[\boldsymbol{K}_f\right]_{ij} &= \sigma_f^2 \cdot \text{SRBF}\left(\boldsymbol{x}_i \boldsymbol{W}_i^\intercal, \boldsymbol{x}_j \boldsymbol{W}_j^\intercal\right);\\
\left[\boldsymbol{K}_{fu}\right]_{ij} &= \sigma_f \cdot \text{SRBF}\left(\boldsymbol{x}_i \boldsymbol{W}_i^\intercal, \mathfrak{z}_j\right);\\
\left[\boldsymbol{K}_w^{(q)}\right]_{ij} &= \text{RBF}_q(\boldsymbol{x}_i, \boldsymbol{x}_j);
\end{aligned}
$$

$$\left[\boldsymbol{K}_v^{(q)}\right]_{ij} = \mathrm{RBF}_q\left(\boldsymbol{\xi}_i, \boldsymbol{\xi}_j\right);$$
$$\left[\boldsymbol{K}_{wv}^{(q)}\right]_{ij} = \mathrm{RBF}_q\left(\boldsymbol{x}_i, \boldsymbol{\xi}_j\right).$$

Now, to continue further, I must define the variational distribution of $\boldsymbol{f}$,$\boldsymbol{u}$,$\boldsymbol{W}$, and $\boldsymbol{V}$.

$$q(\boldsymbol{f}, \boldsymbol{u}, \boldsymbol{V} \mid \boldsymbol{X}, \Xi, \mathfrak{Z}) = p(\boldsymbol{f} \mid \boldsymbol{u}, \boldsymbol{W}, \boldsymbol{X}) p(\boldsymbol{W} \mid \boldsymbol{V}, \boldsymbol{X}) q(\boldsymbol{u}) q(\boldsymbol{V}),$$

where

$$q(\boldsymbol{V}) = \prod_{q,d,i}^{Q,D,m_v} \mathcal{N}\left(v_{qdi} \mid \check{\mu}_{vqdi}, \check{\sigma}_{vqi}\right).$$

For now, I will not define $q(\boldsymbol{u})$. The reason behind this is that I want to find an optimal variational distribution for this variable. The reason that $q(\boldsymbol{V})$ is fully factored is purely computational. If I let it be free-form and find an optimal distribution by it, the ELBO would contain an inverse of an $n \times n$ matrix. I also chose to factor the variance between samples fully because it would reduce the number of variational parameters.

A.1 Evidence lower bound

From now on, I will hide all the dependencies on inputs and pseudo-inputs and, for ease of notation, I will let $\check{\boldsymbol{\Sigma}}_{vq}$ be the matrix with diagonal $\check{\boldsymbol{\sigma}}_{vq}$. The ELBO is then:

$$\begin{aligned}
\ln p(\boldsymbol{y} \mid \boldsymbol{X}, \Xi, \mathfrak{Z}) &= \ln \left\langle p(\boldsymbol{y} \mid \boldsymbol{f}) \frac{q(\boldsymbol{f}, \boldsymbol{W}, \boldsymbol{u}, \boldsymbol{V})}{q(\boldsymbol{f}, \boldsymbol{u}, \boldsymbol{V})} \right\rangle_{p(\boldsymbol{f}, \boldsymbol{W}, \boldsymbol{u}, \boldsymbol{V})} \\
&= \ln \left\langle \frac{p(\boldsymbol{y}, \boldsymbol{f}, \boldsymbol{W}, \boldsymbol{u}, \boldsymbol{V})}{q(\boldsymbol{f}, \boldsymbol{u}, \boldsymbol{V})} \right\rangle_{q(\boldsymbol{f}, \boldsymbol{u}, \boldsymbol{V})} \\
&= \ln \left\langle \frac{p(\boldsymbol{y} \mid, \boldsymbol{f}), p(\boldsymbol{u}) p(\boldsymbol{V})}{q(\boldsymbol{u}) q(\boldsymbol{V})} \right\rangle_{q(\boldsymbol{f}, \boldsymbol{u}, \boldsymbol{V})} \\
&\geq \langle \ln p(\boldsymbol{y} \mid \boldsymbol{f}) \rangle_{q(\boldsymbol{f}, \boldsymbol{u}, \boldsymbol{W}, \boldsymbol{V})} + \left\langle \ln \frac{p(\boldsymbol{u})}{q(\boldsymbol{u})} \right\rangle_{q(\boldsymbol{u})} + \left\langle \ln \frac{p(\boldsymbol{V})}{q(\boldsymbol{V})} \right\rangle_{q(\boldsymbol{V})} \\
&\geq \langle \ln p(\boldsymbol{y} \mid \boldsymbol{f}) \rangle_{q(\boldsymbol{f}, \boldsymbol{u}, \boldsymbol{W}, \boldsymbol{V})} - \mathrm{KL}(q(\boldsymbol{u}) \parallel p(\boldsymbol{u})) - \mathrm{KL}(q(\boldsymbol{V}) \parallel p(\boldsymbol{V})).
\end{aligned}$$

By expanding the KL of $\boldsymbol{V}$:

$$\begin{aligned}
-\mathrm{KL}(q(\boldsymbol{V}) \parallel p(\boldsymbol{V})) = -\frac{1}{2} \sum_{q,d}^{Q,D} \left[\mathrm{Tr}\left[\boldsymbol{K}_v^{(q)^{-1}} \check{\boldsymbol{\Sigma}}_{vq} \right] + \check{\boldsymbol{\mu}}_{qd}^{\mathsf{T}} \boldsymbol{K}_v^{(q)^{-1}} \check{\boldsymbol{\mu}}_{qd} + \ln\left|\boldsymbol{K}_v^{(q)}\right| - \ln\left|\check{\boldsymbol{\Sigma}}_{vq}\right| \right] \\
+ \frac{m_v Q D}{2}.
\end{aligned} \tag{A.1}$$

A.1.1 Expanding $p(\mathbf{y})$

Now, the terms with $\boldsymbol{y}$ and $\boldsymbol{u}$ remain. Let us start with the term with $\boldsymbol{y}$:

$$\begin{aligned}
\langle \ln p(\boldsymbol{y} \mid \boldsymbol{f})\rangle_{q(\boldsymbol{f},\boldsymbol{u},\boldsymbol{W},\boldsymbol{V})} &= \left\langle \ln \mathcal{N}\left(\boldsymbol{y} \mid \boldsymbol{f}, \sigma^2 \boldsymbol{I}\right)\right\rangle_{q(\boldsymbol{f},\boldsymbol{u},\boldsymbol{W},\boldsymbol{V})} \\
&= -\frac{1}{2}\left\langle (\boldsymbol{y}-\boldsymbol{f})(\sigma^2\boldsymbol{I})^{-1}(\boldsymbol{y}-\boldsymbol{f})^{\mathsf{T}} + n\ln(2\pi) + \ln\left|\sigma^2\boldsymbol{I}\right|\right\rangle_{q(\boldsymbol{f},\boldsymbol{u},\boldsymbol{W},\boldsymbol{V})} \\
&= -\frac{1}{2}\left\langle \frac{1}{\sigma^2}(\boldsymbol{y}-\boldsymbol{f})(\boldsymbol{y}-\boldsymbol{f})^{\mathsf{T}} + n\ln(2\pi\sigma^2)\right\rangle_{q(\boldsymbol{f},\boldsymbol{u},\boldsymbol{W},\boldsymbol{V})} \\
&= -\frac{1}{2\sigma^2}\langle (\boldsymbol{y}-\boldsymbol{f})(\boldsymbol{y}-\boldsymbol{f})^{\mathsf{T}}\rangle_{q(\boldsymbol{f},\boldsymbol{u},\boldsymbol{W},\boldsymbol{V})} - \frac{n}{2}\ln(2\pi\sigma^2) \\
&= -\frac{1}{2\sigma^2}\langle \boldsymbol{y}\boldsymbol{y}^{\mathsf{T}} - 2\boldsymbol{y}\boldsymbol{f}^{\mathsf{T}} + \boldsymbol{f}\boldsymbol{f}^{\mathsf{T}}\rangle_{q(\boldsymbol{f},\boldsymbol{u},\boldsymbol{W},\boldsymbol{V})} - \frac{n}{2}\ln(2\pi\sigma^2) \\
&= -\frac{1}{2\sigma^2}\left[\boldsymbol{y}\boldsymbol{y}^{\mathsf{T}} - 2\boldsymbol{y}\langle \boldsymbol{f}\rangle^{\mathsf{T}}_{q(\boldsymbol{f},\boldsymbol{u},\boldsymbol{W},\boldsymbol{V})} + \langle \boldsymbol{f}\boldsymbol{f}^{\mathsf{T}}\rangle_{q(\boldsymbol{f},\boldsymbol{u},\boldsymbol{W},\boldsymbol{V})}\right] - \frac{n}{2}\ln(2\pi\sigma^2).
\end{aligned}$$

By expanding the expected values:

$$\begin{aligned}
\langle \boldsymbol{f}\rangle^{\mathsf{T}}_{q(\boldsymbol{f},\boldsymbol{u},\boldsymbol{W},\boldsymbol{V})} &= \left\langle \boldsymbol{K}_{fu}\boldsymbol{K}_u^{-1}\boldsymbol{u}^{\mathsf{T}}\right\rangle_{q(\boldsymbol{f},\boldsymbol{u},\boldsymbol{W},\boldsymbol{V})} \\
&= \left\langle \boldsymbol{K}_{fu}\right\rangle_{q(\boldsymbol{W},\boldsymbol{V})}\boldsymbol{K}_u^{-1}\langle \boldsymbol{u}\rangle^{\mathsf{T}}_{q(\boldsymbol{u})} \\
&= \boldsymbol{\Psi}_1\boldsymbol{K}_u^{-1}\langle \boldsymbol{u}\rangle^{\mathsf{T}}_{q(\boldsymbol{u})};
\end{aligned}$$

$$\begin{aligned}
\langle \boldsymbol{f}\boldsymbol{f}^{\mathsf{T}}\rangle_{q(\boldsymbol{f},\boldsymbol{u},\boldsymbol{W},\boldsymbol{V})} &= \left\langle \boldsymbol{u}\boldsymbol{K}_u^{-1}\boldsymbol{K}_{uf}\boldsymbol{K}_{fu}\boldsymbol{K}_u^{-1}\boldsymbol{u}^{\mathsf{T}} + \mathrm{Tr}\left[\boldsymbol{K}_f - \boldsymbol{K}_{fu}\boldsymbol{K}_u^{-1}\boldsymbol{K}_{uf}\right]\right\rangle_{q(\boldsymbol{u},\boldsymbol{W},\boldsymbol{V})} \\
&= \left\langle \boldsymbol{u}\boldsymbol{K}_u^{-1}\left\langle \boldsymbol{K}_{uf}\boldsymbol{K}_{fu}\right\rangle_{q(\boldsymbol{W},\boldsymbol{V})}\boldsymbol{K}_u^{-1}\boldsymbol{u}^{\mathsf{T}} + \left\langle \mathrm{Tr}\left[\boldsymbol{K}_f\right] - \mathrm{Tr}\left[\boldsymbol{K}_{fu}\boldsymbol{K}_u^{-1}\boldsymbol{K}_{uf}\right]\right\rangle_{q(\boldsymbol{W},\boldsymbol{V})}\right\rangle_{q(\boldsymbol{u})} \\
&= \left\langle \boldsymbol{u}\boldsymbol{K}_u^{-1}\boldsymbol{\Psi}_2\boldsymbol{K}_u^{-1}\boldsymbol{u}^{\mathsf{T}} + \Psi_0 - \mathrm{Tr}\left[\left\langle \boldsymbol{K}_{uf}\boldsymbol{K}_{fu}\right\rangle_{q(\boldsymbol{W},\boldsymbol{V})}\boldsymbol{K}_u^{-1}\right]\right\rangle_{q(\boldsymbol{u})} \\
&= \left\langle \boldsymbol{u}\boldsymbol{K}_u^{-1}\boldsymbol{\Psi}_2\boldsymbol{K}_u^{-1}\boldsymbol{u}^{\mathsf{T}}\right\rangle_{q(\boldsymbol{u})} + \Psi_0 - \mathrm{Tr}\left[\boldsymbol{\Psi}_2\boldsymbol{K}_u^{-1}\right] \\
&= \left\langle \boldsymbol{u}\boldsymbol{K}_u^{-1}\boldsymbol{\Psi}_2\boldsymbol{K}_u^{-1}\boldsymbol{u}^{\mathsf{T}}\right\rangle_{q(\boldsymbol{u})} + \Psi_0 - \mathrm{Tr}\left[\boldsymbol{\Psi}_2\boldsymbol{K}_u^{-1}\right].
\end{aligned}$$

Replacing them back into $\langle \ln p(\boldsymbol{y} \mid \boldsymbol{f})\rangle_{q(\boldsymbol{f},\boldsymbol{u},\boldsymbol{W},\boldsymbol{V})}$:

$$\begin{aligned}
&-\frac{1}{2\sigma^2}\left[\boldsymbol{y}\boldsymbol{y}^{\mathsf{T}} - 2\boldsymbol{y}\langle \boldsymbol{f}\rangle^{\mathsf{T}}_{q(\boldsymbol{f},\boldsymbol{u},\boldsymbol{W},\boldsymbol{V})} + \langle \boldsymbol{f}\boldsymbol{f}^{\mathsf{T}}\rangle_{q(\boldsymbol{f},\boldsymbol{u},\boldsymbol{W},\boldsymbol{V})}\right] - \frac{n}{2}\ln(2\pi\sigma^2) \\
&-\frac{1}{2\sigma^2}\left[\boldsymbol{y}\boldsymbol{y}^{\mathsf{T}} - 2\boldsymbol{y}\boldsymbol{\Psi}_1\boldsymbol{K}_u^{-1}\langle \boldsymbol{u}\rangle^{\mathsf{T}}_{q(\boldsymbol{u})} + \left\langle \boldsymbol{u}\boldsymbol{K}_u^{-1}\boldsymbol{\Psi}_2\boldsymbol{K}_u^{-1}\boldsymbol{u}^{\mathsf{T}}\right\rangle_{q(\boldsymbol{u})} + \Psi_0 - \mathrm{Tr}\left[\boldsymbol{\Psi}_2\boldsymbol{K}_u^{-1}\right]\right] - \frac{n}{2}\ln(2\pi\sigma^2) \\
&-\frac{1}{2\sigma^2}\left[\boldsymbol{y}\boldsymbol{y}^{\mathsf{T}} + \Psi_0 - \mathrm{Tr}\left[\boldsymbol{\Psi}_2\boldsymbol{K}_u^{-1}\right] + \left\langle \boldsymbol{u}\boldsymbol{K}_u^{-1}\boldsymbol{\Psi}_2\boldsymbol{K}_u^{-1}\boldsymbol{u}^{\mathsf{T}} - 2\boldsymbol{y}\boldsymbol{\Psi}_1\boldsymbol{K}_u^{-1}\boldsymbol{u}^{\mathsf{T}}\right\rangle_{q(\boldsymbol{u})}\right] - \frac{n}{2}\ln(2\pi\sigma^2) \\
&-\frac{1}{2\sigma^2}\left[\boldsymbol{y}\boldsymbol{y}^{\mathsf{T}} + \Psi_0 - \mathrm{Tr}\left[\boldsymbol{\Psi}_2\boldsymbol{K}_u^{-1}\right] + \langle U\rangle_{q(\boldsymbol{u})}\right] - \frac{n}{2}\ln(2\pi\sigma^2) \\
&-\frac{1}{2\sigma^2}\left[\boldsymbol{y}\boldsymbol{y}^{\mathsf{T}} + \Psi_0 - \mathrm{Tr}\left[\boldsymbol{\Psi}_2\boldsymbol{K}_u^{-1}\right]\right] - \frac{n}{2}\ln(2\pi\sigma^2) - \frac{1}{2\sigma^2}\langle U\rangle_{q(\boldsymbol{u})} \qquad \text{(A.2)}
\end{aligned}$$

A.1.2 Expanding terms with u

Let us focus in the expected values in $q(\boldsymbol{u})$:

$$\begin{aligned}
-\frac{1}{2\sigma^2}\langle \mathrm{U}\rangle_{q(\boldsymbol{u})} - \mathrm{KL}(q(\boldsymbol{u}) \parallel p(\boldsymbol{u})) &= \left\langle -\frac{1}{2\sigma^2}\mathrm{U} + \ln\frac{p(\boldsymbol{u})}{q(\boldsymbol{u})} \right\rangle_{q(\boldsymbol{u})} \\
&= \left\langle \ln\exp\left(-\frac{1}{2\sigma^2}\mathrm{U}\right) + \ln\frac{p(\boldsymbol{u})}{q(\boldsymbol{u})} \right\rangle_{q(\boldsymbol{u})} \\
&= \left\langle \ln\frac{p(\boldsymbol{u})\exp\left(-\frac{1}{2\sigma^2}\mathrm{U}\right)}{q(\boldsymbol{u})} \right\rangle_{q(\boldsymbol{u})} \\
&= -\left\langle \ln\frac{q(\boldsymbol{u})}{p(\boldsymbol{u})\exp\left(-\frac{1}{2\sigma^2}\mathrm{U}\right)} \right\rangle_{q(\boldsymbol{u})}.
\end{aligned}$$

Because this equation is a non-normalized KL divergence, its value is minimized when the whole expression is equal to zero. Therefore:

$$q(\boldsymbol{u}) = p(\boldsymbol{u})\exp\left(-\frac{1}{2\sigma^2}\left[\boldsymbol{u}\boldsymbol{K}_u^{-1}\boldsymbol{\Psi}_2\boldsymbol{K}_u^{-1}\boldsymbol{u}^\mathsf{T} - 2\boldsymbol{y}\boldsymbol{\Psi}_1\boldsymbol{K}_u^{-1}\boldsymbol{u}^\mathsf{T}\right]\right)\cdot C. \tag{A.3}$$

For some normalizing constant C. Therefore:

$$\begin{aligned}
q(\boldsymbol{u}) &\propto \mathcal{N}(\boldsymbol{u} \mid \boldsymbol{0}, \boldsymbol{K}_u)\exp\left(-\frac{1}{2\sigma^2}\left[\boldsymbol{u}\boldsymbol{K}_u^{-1}\boldsymbol{\Psi}_2\boldsymbol{K}_u^{-1}\boldsymbol{u}^\mathsf{T} - 2\boldsymbol{y}\boldsymbol{\Psi}_1\boldsymbol{K}_u^{-1}\boldsymbol{u}^\mathsf{T}\right]\right) \\
&\propto (2\pi)^{-\frac{m_u}{2}}|\boldsymbol{K}_u|^{-\frac{1}{2}}\exp\left(-\frac{1}{2}\boldsymbol{u}\boldsymbol{K}_u^{-1}\boldsymbol{u}^\mathsf{T}\right)\exp\left(-\frac{1}{2\sigma^2}\left[\boldsymbol{u}\boldsymbol{K}_u^{-1}\boldsymbol{\Psi}_2\boldsymbol{K}_u^{-1}\boldsymbol{u}^\mathsf{T} - 2\boldsymbol{y}\boldsymbol{\Psi}_1\boldsymbol{K}_u^{-1}\boldsymbol{u}^\mathsf{T}\right]\right) \\
&\propto \exp\left(-\frac{1}{2}\boldsymbol{u}\boldsymbol{K}_u^{-1}\boldsymbol{u}^\mathsf{T} - \frac{1}{2\sigma^2}\boldsymbol{u}\boldsymbol{K}_u^{-1}\boldsymbol{\Psi}_2\boldsymbol{K}_u^{-1}\boldsymbol{u}^\mathsf{T} + \frac{1}{\sigma^2}\boldsymbol{y}\boldsymbol{\Psi}_1\boldsymbol{K}_u^{-1}\boldsymbol{u}^\mathsf{T}\right) \\
&\propto \exp\left(-\frac{1}{2}\boldsymbol{u}\left[\boldsymbol{K}_u^{-1} - \frac{1}{\sigma^2}\boldsymbol{K}_u^{-1}\boldsymbol{\Psi}_2\boldsymbol{K}_u^{-1}\right]\boldsymbol{u}^\mathsf{T} + \frac{1}{\sigma^2}\boldsymbol{y}\boldsymbol{\Psi}_1\boldsymbol{K}_u^{-1}\boldsymbol{u}^\mathsf{T}\right).
\end{aligned}$$

By completing the square (BISHOP, 2006), it follows that:

$$q(\boldsymbol{u}) = \mathcal{N}\left(\boldsymbol{u} \mid \boldsymbol{y}\boldsymbol{\Psi}_1\left[\sigma^2\boldsymbol{K}_u + \boldsymbol{\Psi}_2\right]^{-1}, \boldsymbol{K}_u\left[\sigma^2\boldsymbol{K}_u + \boldsymbol{\Psi}_2\right]^{-1}\boldsymbol{K}_u\right).$$

And through a long calculation, the value of C can be determined:

$$C = \exp\left(-\frac{1}{2}\ln|\boldsymbol{K}_u| + \frac{1}{2}\ln|\sigma^2\boldsymbol{K}_u + \boldsymbol{\Psi}_2| - \frac{m_v}{2}\ln(\sigma^2) - \frac{1}{2\sigma^2}\boldsymbol{y}\boldsymbol{\Psi}_1\left[\sigma^2\boldsymbol{K}_u + \boldsymbol{\Psi}_2\right]^{-1}\boldsymbol{\Psi}_1^\mathsf{T}\boldsymbol{y}^\mathsf{T}\right).$$

Finally, the non-normalized KL divergence can be computed:

$$
\begin{aligned}
-\left\langle \ln \frac{q(\boldsymbol{u})}{p(\boldsymbol{u})\exp\left(-\frac{1}{2\sigma^2}\mathrm{U}\right)} \right\rangle_{q(\boldsymbol{u})} &= \left\langle \ln \frac{p(\boldsymbol{u})\exp\left(-\frac{1}{2\sigma^2}\mathrm{U}\right)}{q(\boldsymbol{u})} \right\rangle_{q(\boldsymbol{u})} \\
&\leq \ln \left\langle \exp\left(-\frac{1}{2\sigma^2}\mathrm{U}\right) \right\rangle_{p(\boldsymbol{u})} && \text{(By Jensen's Inequality.)} \\
&\leq \ln \int_{\boldsymbol{u}} \exp\left(-\frac{1}{2\sigma^2}\mathrm{U}\right) p(\boldsymbol{u}) \\
&\leq \ln \int_{\boldsymbol{u}} q(\boldsymbol{u}) \frac{1}{C} && \text{(Due to Eq A.3.)} \\
&\leq \ln \frac{1}{C} \\
&\leq \ \frac{1}{2}\ln|\boldsymbol{K}_u| - \frac{1}{2}\ln|\sigma^2\boldsymbol{K}_u + \boldsymbol{\Psi}_2| + \frac{m_v}{2}\ln(\sigma^2) && \text{(A.4)} \\
&\quad + \frac{1}{2\sigma^2}\boldsymbol{y}\boldsymbol{\Psi}_1\left[\sigma^2\boldsymbol{K}_u + \boldsymbol{\Psi}_2\right]^{-1}\boldsymbol{\Psi}_1^\intercal \boldsymbol{y}^\intercal.
\end{aligned}
$$

A.1.3 Putting the ELBO together

By combining Eqs. A.2, A.1, and A.4, the following ELBO is obtained:

$$
\begin{aligned}
p(\boldsymbol{y}) \geq & -\frac{n}{2}\log(2\pi) - \frac{n-m_u}{2}\log\left(\sigma^2\right) - \frac{1}{2\sigma^2}\psi_0 \\
& + \frac{1}{2}\log\left(|\sigma^2\boldsymbol{K}_u + \boldsymbol{\Psi}_2|\right) + \log(|\boldsymbol{K}_u|) + \frac{1}{2\sigma^2}\mathrm{Tr}\left(\boldsymbol{K}_u^{-1}\boldsymbol{\Psi}_2\right) \\
& - \frac{1}{2\sigma^2}\boldsymbol{y}\boldsymbol{y}^\intercal + \frac{1}{2\sigma^2}\boldsymbol{y}\boldsymbol{\Psi}_1\left(\sigma^2\boldsymbol{K}_u + \boldsymbol{\Psi}_2\right)^{-1}\boldsymbol{\Psi}_1^\intercal\boldsymbol{y}^\intercal \\
& - \frac{1}{2}\sum_{q,d}^{Q,D}\left[\mathrm{Tr}\left[\boldsymbol{K}_v^{(q)^{-1}}\check{\boldsymbol{\Sigma}}_{vq}\right] + \check{\boldsymbol{\mu}}_{qd}^\intercal\boldsymbol{K}_v^{(q)^{-1}}\check{\boldsymbol{\mu}}_{qd} + \ln\left|\boldsymbol{K}_v^{(q)}\right| - \ln\left|\check{\boldsymbol{\Sigma}}_{vq}\right|\right] + \frac{m_v Q D}{2}.
\end{aligned}
$$

A.2 Deriving Ψ-statistics

Before deriving the formulas for the Ψ-statistics, I will first derive an formula for the parameters of $q(\boldsymbol{W})$.

$$
\begin{aligned}
q(\boldsymbol{W}) &= \langle p(\boldsymbol{W} \mid \boldsymbol{V})\rangle_{q(\boldsymbol{V})} \\
&= \prod_{q,d}^{Q,D}\left\langle p(\boldsymbol{w}_{qd} \mid \boldsymbol{v}_{qd})\right\rangle_{q(\boldsymbol{v}_{qd})} \\
&= \prod_{q,d}^{Q,D}\mathcal{N}\left(\boldsymbol{w}_{qd} \,\middle|\, \check{\boldsymbol{\mu}}_{vqd}\boldsymbol{K}_v^{(q)^{-1}}\boldsymbol{K}_{vw}^{(q)}, \boldsymbol{K}_w^{(q)} - \boldsymbol{K}_{wv}^{(q)}\boldsymbol{K}_v^{(q)^{-1}}\left(\boldsymbol{K}_v^{(q)} + \check{\boldsymbol{\Sigma}}_{vq}\right)\boldsymbol{K}_v^{(q)^{-1}}\boldsymbol{K}_{vw}^{(q)}\right) \\
&= \prod_{q,d}^{Q,D}\mathcal{N}\left(\boldsymbol{w}_{qd} \mid \tilde{\boldsymbol{\mu}}_{wqd}, \tilde{\boldsymbol{\Sigma}}_{wq}\right).
\end{aligned}
$$

Starting with ψ_0:

$$
\begin{aligned}
\psi_0 &= \left\langle \operatorname{Tr}\left[\boldsymbol{K}_f\right] \right\rangle_{q(\boldsymbol{W},\boldsymbol{V})} \\
&= \left\langle \sum_{i}^{n} \left[\sigma_f^2 \mathrm{SRBF}\left(\boldsymbol{x}_i \boldsymbol{W}_i^\top, \boldsymbol{x}_i \boldsymbol{W}_i^\top, \right)\right] \right\rangle_{q(\boldsymbol{W},\boldsymbol{V})} \\
&= \left\langle \sum_{i}^{n} \sigma_f^2 \right\rangle_{q(\boldsymbol{W},\boldsymbol{V})} \\
&= \left\langle n\sigma_f^2 \right\rangle_{q(\boldsymbol{W},\boldsymbol{V})} \\
&= n\sigma_f^2.
\end{aligned}
$$

Moving on to $\boldsymbol{\Psi}_1$:

$$
\begin{aligned}
[\boldsymbol{\Psi}_1]_{ij} &= \left\langle \left[\boldsymbol{K}_{fu}\right]_{ij} \right\rangle_{q(\boldsymbol{W},\boldsymbol{V})} \\
&= \left\langle \sigma_f \cdot \mathrm{SRBF}\left(\boldsymbol{x}_i \boldsymbol{W}_i^\top, \mathfrak{z}_j\right) \right\rangle_{q(\boldsymbol{W},\boldsymbol{V})} \\
&= \int_{\boldsymbol{W},\boldsymbol{V}} \sigma_f \cdot \mathrm{SRBF}\left(\boldsymbol{x}_i \boldsymbol{W}_i^\top, \mathfrak{z}_j\right) q(\boldsymbol{W},\boldsymbol{V}) \\
&= \sigma_f \int_{\boldsymbol{W}} \mathrm{SRBF}\left(\boldsymbol{x}_i \boldsymbol{W}_i^\top, \mathfrak{z}_j\right) \int_{\boldsymbol{V}} q(\boldsymbol{W},\boldsymbol{V}) \\
&= \sigma_f \int_{\boldsymbol{W}} \exp\left(-\frac{1}{2}\left\|\boldsymbol{x}_i \boldsymbol{W}_i^\top - \mathfrak{z}_j\right\|^2\right) q(\boldsymbol{W}) \\
&= \sigma_f \int_{\boldsymbol{W}} \exp\left(-\frac{1}{2}\sum_{q}^{Q}\left(\boldsymbol{x}_i \boldsymbol{w}_{q:i}^\top - \mathfrak{z}_{jq}\right)^2\right) \prod_{q,d}^{Q,D} \mathcal{N}\left(\boldsymbol{w}_{qd} \mid \tilde{\boldsymbol{\mu}}_{wqd}, \tilde{\boldsymbol{\Sigma}}_{wq}\right) \\
&= \sigma_f \int_{\boldsymbol{W}} \prod_{q}^{Q} \exp\left(-\frac{1}{2}\left(\boldsymbol{x}_i \boldsymbol{w}_{q:i}^\top - \mathfrak{z}_{jq}\right)^2\right) \prod_{d}^{D} \mathcal{N}\left(\boldsymbol{w}_{qd} \mid \tilde{\boldsymbol{\mu}}_{wqd}, \tilde{\boldsymbol{\Sigma}}_{wq}\right) \\
&= \sigma_f \prod_{q}^{Q} \int_{\boldsymbol{W}_q} \exp\left(-\frac{1}{2}\left(\boldsymbol{x}_i \boldsymbol{w}_{q:i}^\top - \mathfrak{z}_{jq}\right)^2\right) \prod_{d}^{D} \mathcal{N}\left(\boldsymbol{w}_{qd} \mid \tilde{\boldsymbol{\mu}}_{wqd}, \tilde{\boldsymbol{\Sigma}}_{wq}\right) \\
&= \sigma_f \prod_{q}^{Q} \int_{\boldsymbol{w}_{q:i}} \exp\left(-\frac{1}{2}\left(\boldsymbol{x}_i \boldsymbol{w}_{q:i}^\top - \mathfrak{z}_{jq}\right)^2\right) \prod_{d}^{D} \int_{\boldsymbol{w}_{qd} \setminus w_{qdi}} \mathcal{N}\left(\boldsymbol{w}_{qd} \mid \tilde{\boldsymbol{\mu}}_{wqd}, \tilde{\boldsymbol{\Sigma}}_{wq}\right) \\
&= \sigma_f \prod_{q}^{Q} \int_{\boldsymbol{w}_{q:i}} \exp\left(-\frac{1}{2}\left(\boldsymbol{x}_i \boldsymbol{w}_{q:i}^\top - \mathfrak{z}_{jq}\right)^2\right) \prod_{d}^{D} \mathcal{N}\left(w_{qdi} \mid \tilde{\mu}_{wqdi}, \tilde{\sigma}_{wqii}\right) \\
&= \sigma_f \prod_{q}^{Q} \int_{\boldsymbol{w}_{q:i}} \exp\left(-\frac{1}{2}\left(\boldsymbol{x}_i \boldsymbol{w}_{q:i}^\top - \mathfrak{z}_{jq}\right)^2\right) \mathcal{N}\left(\boldsymbol{w}_{q:i} \mid \tilde{\boldsymbol{\mu}}_{wq:i}, \tilde{\sigma}_{wqii}\boldsymbol{I}\right) \\
&= \sigma_f \prod_{q}^{Q} \left(\tilde{\sigma}_{wqii}\boldsymbol{x}_i\boldsymbol{x}_i^\top + 1\right)^{\frac{1}{2}} \exp\left(-\frac{1}{2}\frac{\left(\boldsymbol{x}_i \tilde{\boldsymbol{\mu}}_{q:i}^\top - \mathfrak{z}_{jq}\right)^2}{\tilde{\sigma}_{wqii}\boldsymbol{x}_i\boldsymbol{x}_i^\top + 1}\right).
\end{aligned}
$$

Where the last step is identical to the derivation in section B.1 of Titsias and Lázaro-Gredilla (2013).

Now, only $\boldsymbol{\Psi}_2$ remains:

$$\begin{aligned}
[\boldsymbol{\Psi 2}_2]_{jk} &= \left\langle [\boldsymbol{K}_{uf}\boldsymbol{K}_{fu}]_{jk} \right\rangle_{q(\boldsymbol{W},\boldsymbol{V})} \\
&= \left\langle \sum_i^n \sigma_f \cdot \mathrm{SRBF}\big(\boldsymbol{x}_i\boldsymbol{W}_i^\mathsf{T}, \mathfrak{z}_j\big)\sigma_f \cdot \mathrm{SRBF}\big(\boldsymbol{x}_i\boldsymbol{W}_i^\mathsf{T}, \mathfrak{z}_k\big) \right\rangle_{q(\boldsymbol{W},\boldsymbol{V})} \\
&= \sigma_f^2 \sum_i^n \left\langle \mathrm{SRBF}\big(\boldsymbol{x}_i\boldsymbol{W}_i^\mathsf{T}, \mathfrak{z}_j\big)\mathrm{SRBF}\big(\boldsymbol{x}_i\boldsymbol{W}_i^\mathsf{T}, \mathfrak{z}_k\big) \right\rangle_{q(\boldsymbol{W},\boldsymbol{V})} \\
&= \sigma_f^2 \sum_i^n \int_{\boldsymbol{W}} \exp\left(-\frac{1}{2} \left[\left\|\boldsymbol{x}_i\boldsymbol{W}_i^\mathsf{T} - \mathfrak{z}_j\right\|^2 + \left\|\boldsymbol{x}_i\boldsymbol{W}_i^\mathsf{T} - \mathfrak{z}_k\right\|^2 \right] \right) q(\boldsymbol{W}) \\
&= \sigma_f^2 \sum_i^n \prod_q^Q \int_{\boldsymbol{w}_{q:i}} \exp\left(-\frac{1}{2} \left[\left(\boldsymbol{x}_i\boldsymbol{w}_{q:i}^\mathsf{T} - \mathfrak{z}_{jq}\right)^2 + \left(\boldsymbol{x}_i\boldsymbol{w}_{q:i}^\mathsf{T} - \mathfrak{z}_{kq}\right)^2 \right] \right) \mathcal{N}\left(\boldsymbol{w}_{q:i} \mid \tilde{\boldsymbol{\mu}}_{wq:i}, \tilde{\sigma}_{wqii}\boldsymbol{I}\right) \\
&= \sigma_f^2 \sum_i^n \prod_q^Q \left[2\tilde{\sigma}_{wqii}\boldsymbol{x}_i\boldsymbol{x}_i^\mathsf{T} + 1\right]^{-\frac{1}{2}} \exp\left[-\frac{\left(\boldsymbol{x}_i\tilde{\boldsymbol{\mu}}_{wq:i}^\mathsf{T} - \bar{\mathfrak{z}}_{jkq}\right)^2}{2s_{qi}\boldsymbol{x}_i\boldsymbol{x}_i^\mathsf{T} + 1} - \frac{\left(\mathfrak{z}_{jq} - \mathfrak{z}_{kq}\right)^2}{4} \right],
\end{aligned}$$

where:

$$\bar{\mathfrak{z}}_{jk} = \frac{\mathfrak{z}_j + \mathfrak{z}_k}{2}.$$

Again, the last step is identical to the derivation in section B.1 of Titsias and Lázaro-Gredilla (2013).