University of Sheffield

Variational Optimisation for Non-conjugate Likelihood Gaussian Process Models

Juan José Giraldo Gutiérrez

Supervisor: Mauricio A. Álvarez, PhD

A thesis submitted in partial fulfilment of the requirements
for the degree of Doctor of Philosophy

in the

Department of Computer Science

August 6, 2021

Declaration

All sentences or passages quoted in this document from other people's work have been specifically acknowledged by clear cross-referencing to author, work and page(s). Any illustrations that are not the work of the author of this report have been used with the explicit permission of the originator and are specifically acknowledged. I understand that failure to do this amounts to plagiarism and will be considered grounds for failure.

Name: _____

Signature: _____

Date: _____

Abstract

In this thesis we address the problems associated to non-conjugate likelihood Gaussian process models, i.e., probabilistic models where the likelihood function and the Gaussian process priors are non-conjugate. Such problems include intractability, scalability, and poor local optima solutions for the parameters and hyper-parameters of the models. Particularly, in this thesis we address the aforementioned issues in the context of probabilistic models, where the likelihood's parameters are modelled as latent parameter functions drawn from correlated Gaussian processes. We study three ways to generate such latent parameter functions: 1. from a linear model of coregionalisation; 2. from convolution processes, i.e., a convolution integral between smoothing kernels and Gaussian process priors; and 3. using variational inducing kernels, an alternative form to generate the latent parameter functions through the convolution processes formalism, by using a double convolution integral. We borrow ideas from different variational optimisation mechanisms, that consist on introducing a variational (or exploratory) distribution over the model so as to build objective functions that: allow us to deal with intractability as well as enabling scalability when needing to hand massive amounts of data observations. Also, such variational optimisations mechanisms grant us to perform inference of the model hyper-parameters together with the posterior's parameters through a fully natural gradient optimisation scheme; a useful scheme for tackling the problem of poor local optima solutions. Such variational optimisation mechanisms have been broadly studied in the context of reinforcement and Bayesian deep learning showing to be successful exploratory-learning tools; nonetheless, they have not been much studied in the context of Gaussian process models, so we provide a study of their performance in said context.

Contents

List of Figures

List of Tables

Acknowledgements

I would like to thank my supervisor Mauricio A. Álvarez for believing that I might be able to carry out this journey of studying a PhD. Thanks to Mauricio for encouraging me to think broadly without limiting my perspective when facing a problem, also for the detailed feedback that surely helped me to improve my writing skills. Thanks to my beloved wife Liza María Gonzalez who has strongly provided the support I have needed during this doctorate journey, also thanks for deciding to leave our home country to experience a completely new culture and language. Thanks to Chunchao, Fariba, Senee, Zhuo, Chao, Donya, Lili, George, Wil Ward, Mike Smith, Tianqui, Luka, Yan Ge, August and Nada for the relevant mathematical discussions, the occasional lunch or dinner times and, the pints and chats in a pub; I'll treasure in my heart a bit of your lives and cultures. Thanks to all people in the office 136 in Regent Court, 211 Portobello, Sheffield, UK. Thanks to Eleni Vasilaki for all the recommendations and comments regarding my research work. Thanks to Hernán Felipe García, Cristian Torres and Pablo Alvarado for being like a Colombian family in a foreign country. Thanks to Pablo Moreno Muñoz for all the discussions and help with regard to the Gaussian processes models for heterogeneous outputs, and of course thanks for the amazing times we could share when visiting Sheffield. Also, I want to thank Julián Gil for inviting me to develop the work of correlated Gaussian processes for datasets with multiple annotators, I believe we both benefited from several discussions about it.

Thanks to my family, my siblings Paula and Diego, and my father Hernando Giraldo for their continuous support and trust in me. Thanks to God and his wisdom for making firm my steps (Psalm 37:23).

Thanks to Peter and Anita Lines for their continuous support, prayers and love while living in the UK; thanks for the delicious english meals you cooked for my wife and I.

I would like to dedicate this work to my mother Luz Nidia Gutierrez, who passed away one month before starting this PhD journey; she left her light of love, kindness and compassion to be treasured in our lives.

Symbols and abbreviations

Generalities

P	dimensionality of the input space
D	number of outputs
M	number of inducing points per latent function $u_q(\cdot)$
N	number of data observations
Q	number of latent functions $u_q(\cdot)$
J_d	number of latent parameter functions chained to the d-th likelihood
J	total number of latent parameter functions chained to the likelihoods' parameters, $J = \sum_{d=1}^{D} J_d$
R	number of independent and identically distributed samples drawn per q-th latent function $u_q(\cdot)$ or $\iota_q(\cdot)$
$\tilde{\alpha}_t$, $\tilde{\beta}_t$, $\tilde{\gamma}_t$, $\tilde{\upsilon}_t$	positive step-size parameters
α_t, β_t, γ_t, υ_t	positive step-size parameters, where $\alpha_t = \tilde{\alpha}_t/(1-\tilde{\gamma}_t)$, $\beta_t = \tilde{\beta}_t/(1-\tilde{\upsilon}_t)$, $\gamma_t = \tilde{\gamma}_t/(1-\tilde{\gamma}_t)$ and $\upsilon_t = \tilde{\upsilon}_t/(1-\tilde{\upsilon}_t)$
$\mathbf{X}$	input training data, $\mathbf{X} = \{\mathbf{x}_n\}_{n=1}^{N}$
$\mathbf{y}$	output training data, $\mathbf{y} = \{\mathbf{y}_d\}_{d=1}^{D}$, where $\mathbf{y}_d = [y_{d,1}, ..., y_{d,N}]^\top$
$\mathbf{Z}_q$	set of inducing points per latent function $u_q(\cdot)$, $\mathbf{Z}_q = \{\mathbf{z}_q^{(m)}\}_{m=1}^{M}$
$\mathbf{Z}_{d,j}$	set of inducing points per latent function $f_{d,j}(\cdot)$, $\mathbf{Z}_{d,j} = \{\mathbf{z}_{d,j}^{(m)}\}_{m=1}^{M}$
$\mathbf{z}$	vector that stacks all inducing points, $\{\mathbf{Z}_q\}_{q=1}^{Q}$ or $\{\mathbf{Z}_{d,j}\}_{d=1,j=1}^{D,J_d}$ as per the model that applies

Operators

$\text{Cov}[\cdot, \cdot]$	covariance function
$\text{E}[\cdot]$	expected value
$\mathbf{A} \odot \mathbf{B}$	Hadamard product between matrices $\mathbf{A}$ and $\mathbf{B}$
$\text{diag}(\cdot)$	operator that acts over a diagonal matrix or vector. For instance, if applied to a diagonal matrix, $\mathbf{M} \in \mathbb{R}^{P \times P}$, the operator maps the diagonal of the matrix into a vector $\mathbf{m} \in \mathbb{R}^{P \times 1}$, i.e., $\mathbf{m} = \text{diag}(\mathbf{M}) = \mathbf{M}\mathbf{1}_P$, where $\mathbf{1}_P$ is a vector of ones with length P. If applied to a vector $\mathbf{m} \in \mathbb{R}^{P \times 1}$, the operator maps the vector to a diagonal matrix, i.e., $\mathbf{M} = \text{diag}(\mathbf{m})$, where the diagonal matrix becomes, $\mathbf{M} \in \mathbb{R}^{P \times P}$, with the elements of $\mathbf{m}$ in its diagonal.

Functions

$u_q(\mathbf{x})$	q-th latent function evaluated at $\mathbf{x}$
$u_q^i(\cdot)$	i-th sample of $u_q(\cdot)$ drawn independent and identically distributed
u	represents a vector of functions that stacks all R independent and identically distributed samples $u_q^i(\cdot)$ of all groups Q, i.e., $u = [{u_1^1}^\top, ..., {u_Q^1}^\top, ..., {u_1^R}^\top, ..., {u_Q^R}^\top]^\top$
$f_{d,j}(\mathbf{x})$	j-th latent function belonging to the d-th output evaluated at $\mathbf{x}$
$f_j(\mathbf{x})$	alternative notation of $f_{d,j}(\mathbf{x})$ used for a single-output model ($D = 1$), i.e., $f_j(\mathbf{x}) := f_{1,j}(\mathbf{x})$
$\check{u}_{d,j}(\mathbf{x})$	additional evaluation of the latent function $f_{d,j}$ at $\mathbf{x}$, i.e., $\check{u}_{d,j}(\mathbf{x}) := f_{d,j}(\mathbf{x})$
$\check{u}_j(\mathbf{x})$	alternative notation of $\check{u}_{d,j}(\mathbf{x})$ used for a single-output model ($D = 1$), i.e., $\check{u}_j(\mathbf{x}) := \check{u}_{1,j}(\mathbf{x})$
$\iota_q(\mathbf{x})$	q-th inducing function evaluated at $\mathbf{x}$
$\iota_q^i(\cdot)$	i-th sample of $\iota_q(\cdot)$ drawn independent and identically distributed
ι	continuous inducing function infinitely computed at all possible $\mathbf{x} \in \mathbb{R}^{P\times 1}$, it is built as $\iota = [{\iota_1^1}^\top, ..., {\iota_Q^1}^\top, ..., {\iota_1^R}^\top, ..., {\iota_Q^R}^\top]^\top$
$\psi_{d,j}(\cdot)$	j-th parameter belonging to the d-th likelihood function.
$\psi_j(\cdot)$	alternative notation of $\psi_{d,j}(\cdot)$ used for a single-output model ($D = 1$), i.e., $\psi_j(\cdot) := \psi_{1,j}(\cdot)$
$\phi(\cdot)$	general notation of a link function
$\phi_{d,j}(\cdot)$	link function associated to the likelihood's parameter $\psi_{d,j}(\cdot)$.
$\phi_j(\cdot)$	alternative notation of $\phi_{d,j}(\cdot)$ used for a single-output model ($D = 1$), i.e., $\phi_j(\cdot) := \phi_{1,j}(\cdot)$
$k_q(\mathbf{x}, \mathbf{x}')$	Gaussian process covariance function of $u_q(\mathbf{x})$
$k_{f_{d,j},u_q}(\mathbf{x}, \mathbf{x}')$	cross-covariance between latent functions $f_{d,j}(\mathbf{x})$ and $u_q(\mathbf{x}')$
$k_{f_{d,j},f_{d',j'}}(\mathbf{x}, \mathbf{x}')$	covariance between latent functions $f_{d,j}(\mathbf{x})$ and $f_{d',j'}(\mathbf{x}')$
$k_{f_j,u_q}(\mathbf{x}, \mathbf{x}')$	cross-covariance between latent functions $f_j(\mathbf{x})$ and $u_q(\mathbf{x}')$
$k_{f_j,f_{j'}}(\mathbf{x}, \mathbf{x}')$	covariance between latent functions $f_j(\mathbf{x})$ and $f_{j'}(\mathbf{x}')$

Vectors and Matrices

$\mathbf{f}_{d,j}$	$f_{d,j}(\mathbf{x})$ evaluated at $\mathbf{X}$, $\mathbf{f}_{d,j} = \left[f_{d,j}\left(\mathbf{x}_1\right), ..., f_{d,j}\left(\mathbf{x}_N\right)\right]^\top$
$\mathbf{f}_j$	$f_j(\mathbf{x})$ evaluated at $\mathbf{X}$, $\mathbf{f}_j = \left[f_j\left(\mathbf{x}_1\right), ..., f_j\left(\mathbf{x}_N\right)\right]^\top$
$\mathbf{f}$	vectors $\{\mathbf{f}_{d,j}\}_{d=1,j=1}^{D,J_d}$ stacked in a column vector
$\mathbf{K}_{\mathbf{f}_{d,j}\mathbf{f}_{d,j}}$	covariance matrix with entries $k_{f_{d,j},f_{d,j}}\left(\mathbf{x}, \mathbf{x}'\right)$ evaluated at $\mathbf{X}$
$\mathbf{K}_{\mathbf{f}_j\mathbf{f}_j}$	covariance matrix with entries $k_{f_j,f_j}\left(\mathbf{x}, \mathbf{x}'\right)$ evaluated at $\mathbf{X}$
$\mathbf{u}_q$	$u_q(\mathbf{x})$ evaluated at $\mathbf{Z}_q$, $\mathbf{u}_q = [u_q(\mathbf{z}_q^{(1)}), \ldots, u_q(\mathbf{z}_q^{(M)})]^\top$
$\mathbf{u}$	vectors $\{\mathbf{u}_q\}_{q=1}^{Q}$ stacked in a column vector
$\mathbf{K}_{\mathbf{u}_q\mathbf{u}_q}$	covariance matrix with entries $k_q\left(\mathbf{x}, \mathbf{x}'\right)$ evaluated at $\mathbf{Z}_q$
$\mathbf{K}_{\mathbf{u}\mathbf{u}}$	block-diagonal covariance matrix built with blocks $\mathbf{K}_{\mathbf{u}_q\mathbf{u}_q}$
$\mathbf{K}_{\mathbf{f}_{d,j}\mathbf{u}_q}$	cross-covariance matrix with entries $k_{f_{d,j},u_q}\left(\mathbf{x}, \mathbf{x}'\right)$ evaluated between $\mathbf{X}$ and $\mathbf{Z}_q$
$\mathbf{K}_{\mathbf{f}_{d,j}\mathbf{u}}$	cross-covariance matrix constructed with blocks $\mathbf{K}_{\mathbf{f}_{d,j}\mathbf{u}_q}$, i.e., $\mathbf{K}_{\mathbf{f}_{d,j}\mathbf{u}} = [\mathbf{K}_{\mathbf{f}_{d,j}\mathbf{u}_1}, ..., \mathbf{K}_{\mathbf{f}_{d,j}\mathbf{u}_Q}]$
$\mathbf{K}_{\mathbf{f}_j\mathbf{u}_q}$	cross-covariance matrix with entries $k_{f_j,u_q}\left(\mathbf{x}, \mathbf{x}'\right)$ evaluated between $\mathbf{X}$ and $\mathbf{Z}_q$
$\mathbf{K}_{\mathbf{f}_j\mathbf{u}}$	cross-covariance matrix constructed with blocks $\mathbf{K}_{\mathbf{f}_j\mathbf{u}_q}$, i.e., $\mathbf{K}_{\mathbf{f}_j\mathbf{u}} = [\mathbf{K}_{\mathbf{f}_j\mathbf{u}_1}, ..., \mathbf{K}_{\mathbf{f}_j\mathbf{u}_Q}]$
$\check{\mathbf{u}}_{d,j}$	$\check{u}_{d,j} := f_{d,j}(\mathbf{x})$ evaluated at $\mathbf{Z}_{d,j}$, $\check{\mathbf{u}}_{d,j} = [f_{d,j}(\mathbf{z}_{d,j}^{(1)}), ..., f_{d,j}(\mathbf{z}_{d,j}^{(M)})]^\top$
$\check{\mathbf{u}}_j$	$\check{u}_j(\mathbf{x}) := f_j(\mathbf{x})$ evaluated at $\mathbf{Z}_j$, $\check{\mathbf{u}}_j = [f_j(\mathbf{z}_j^{(1)}), ..., f_j(\mathbf{z}_j^{(M)})]^\top$
$\check{\mathbf{u}}$	vectors $\{\check{\mathbf{u}}_{d,j}\}_{d=1,j=1}^{D,J_d}$ stacked in a column vector
$\mathbf{V}_q$	covariance matrix that belongs to the Gaussian variational posterior $q(\mathbf{u}_q)$
$\mathbf{m}_q$	mean vector that belongs to the Gaussian variational posterior $q(\mathbf{u}_q)$
$\mathbf{V}_{d,j}$	covariance matrix that belongs to the Gaussian variational posterior $q(\check{\mathbf{u}}_{d,j})$
$\mathbf{m}_{d,j}$	mean vector that belongs to the Gaussian variational posterior $q(\check{\mathbf{u}}_{d,j})$
$\mathbf{V}$	block-diagonal covariance matrix built with blocks $\mathbf{V}_q$
$\mathbf{m}$	vectors $\{\mathbf{m}_q\}_{q=1}^{Q}$ stacked in a column vector
$\boldsymbol{\Sigma}$	covariance matrix that belongs to the Gaussian exploratory distribution $q(\boldsymbol{\theta})$ for the CCGP model ($D = 1$)
$\boldsymbol{\mu}$	mean vector that belongs to the Gaussian exploratory distribution $q(\boldsymbol{\theta})$ for the CCGP model ($D = 1$)

$\boldsymbol{\eta}$	set of mean-parameters that belong to the distribution $q(\boldsymbol{\theta})$, i.e., $\boldsymbol{\eta} = \{\boldsymbol{\mu}, \boldsymbol{\mu}\boldsymbol{\mu}^\top + \boldsymbol{\Sigma}\}$
$\boldsymbol{\xi}$	set of natural parameters that belong to the distribution $q(\boldsymbol{\theta})$, i.e., $\boldsymbol{\xi} = \{\boldsymbol{\Sigma}^{-1}\boldsymbol{\mu}, -\frac{1}{2}\boldsymbol{\Sigma}^{-1}\}$
$\boldsymbol{\rho}_q$	set of mean-parameters that belong to the distribution $q(\mathbf{u}_q)$, i.e., $\boldsymbol{\rho}_q = \{\mathbf{m}_q, \mathbf{m}_q\mathbf{m}_q^\top + \mathbf{V}_q\}$
$\boldsymbol{\mu}^D$	mean vector that belongs to the Gaussian exploratory distribution $q(\boldsymbol{\theta})$ for a Multi-Output GP model with an LMC
$\boldsymbol{\Sigma}^D$	covariance matrix that belongs to the Gaussian exploratory distribution $q(\boldsymbol{\theta})$ for a Multi-Output GP model with an LMC
$\boldsymbol{\eta}^D$	set of mean-parameters that belong to the distribution $q(\boldsymbol{\theta})$ for a Multi-Output GP model with an LMC, i.e., $\boldsymbol{\eta}^D = \{\boldsymbol{\mu}^D, \boldsymbol{\mu}^D\boldsymbol{\mu}^{D\top} + \boldsymbol{\Sigma}^D\}$
$\boldsymbol{\mu}^C$	mean vector that belongs to the Gaussian exploratory distribution $q(\boldsymbol{\theta})$ for a Multi-Output GP model with a CPM using inducing variables $\check{u}_{d,j}(\cdot)$
$\boldsymbol{\Sigma}^C$	covariance matrix that belongs to the Gaussian exploratory distribution $q(\boldsymbol{\theta})$ for a Multi-Output GP model with an CPM using inducing variables $\check{u}_{d,j}(\cdot)$
$\boldsymbol{\eta}^C$	set of mean-parameters that belong to the distribution $q(\boldsymbol{\theta})$ for a Multi-Output GP model with a CPM using inducing variables $\check{u}_{d,j}(\cdot)$, i.e., $\boldsymbol{\eta}^C = \{\boldsymbol{\mu}^C, \boldsymbol{\mu}^C\boldsymbol{\mu}^{C\top} + \boldsymbol{\Sigma}^C\}$

$\boldsymbol{\iota}_q^i$	$\iota_q^i(\cdot)$ evaluated at $\mathbf{Z}_q$, i.e., $\boldsymbol{\iota}_q^i = [\iota_q^i(\mathbf{z}_q^{(1)}), \ldots, \iota_q^i(\mathbf{z}_q^{(M)})]^\top$
$\boldsymbol{\iota}$	vectors $\{\boldsymbol{\iota}_q^i\}_{q=1,i=1}^{Q,R}$ stacked in a column vector, i.e., $\boldsymbol{\iota} = [\boldsymbol{\iota}_1^{1\top}, ..., \boldsymbol{\iota}_Q^{1\top}, ..., \boldsymbol{\iota}_1^{R\top}, ..., \boldsymbol{\iota}_Q^{R\top}]^\top$
$\mathbf{K}_{\mathbf{f}_j\boldsymbol{\iota}_q^i}$	is a cross-covariance matrix with entries computed using the covariance function, $\text{Cov}\left[f_j(\mathbf{x}), \iota_q^i(\mathbf{z})\right]$, between $\mathbf{X}$ and $\mathbf{Z}_q$
$\mathbf{K}_{\mathbf{f}_j\boldsymbol{\iota}}$	is a covariance matrix constructed as $\mathbf{K}_{\mathbf{f}_j\boldsymbol{\iota}} = [\mathbf{K}_{\mathbf{f}_j\boldsymbol{\iota}_1^1}, ..., \mathbf{K}_{\mathbf{f}_j\boldsymbol{\iota}_Q^1}, ..., \mathbf{K}_{\mathbf{f}_j\boldsymbol{\iota}_1^R}, ..., \mathbf{K}_{\mathbf{f}_j\boldsymbol{\iota}_Q^R}]$
$\mathbf{K}_{\mathbf{f}\boldsymbol{\iota}}$	is as a cross covariance matrix built with blocks $\mathbf{K}_{\mathbf{f}_j\boldsymbol{\iota}}$, i.e., $\mathbf{K}_{\mathbf{f}\boldsymbol{\iota}} = [\mathbf{K}_{\mathbf{f}_1\boldsymbol{\iota}}^\top, ..., \mathbf{K}_{\mathbf{f}_J\boldsymbol{\iota}}^\top]^\top$
$\mathbf{K}_{\mathbf{ff}}$	is a covariance matrix built with evaluations of $\text{Cov}[f_j(\mathbf{x}), f_{j'}(\mathbf{x}')]$ for all J Latent Parameter Functions, between all pairs $\mathbf{X}$
$\mathbf{K}_{\boldsymbol{\iota}_q^i\boldsymbol{\iota}_q^i}$	is a covariance matrix which entries are calculated with $k_{\iota_q^i}(\mathbf{z}, \mathbf{z}') = \text{Cov}\left[\iota_q^i(\mathbf{z}), \iota_q^i(\mathbf{z}')\right]$ between all pairs $\mathbf{Z}_q$
$\mathbf{K}_{\boldsymbol{\iota}\boldsymbol{\iota}}$	is a block-diagonal matrix built with blocks $\mathbf{K}_{\boldsymbol{\iota}_q^i\boldsymbol{\iota}_q^i}$

Abbreviations

GP	Gaussian Process
VO	Variational Optimisation
VI	Variational Inference
SVI	Stochastic Variational Inference
SGD	Stochastic Gradient Descent
ADAD	Ada-Delta
AGM	Adaptive Gradient Method
MDA	Mirror Descent Algorithm
VAN	Variational Adaptive Newton
LMC	Linear Model of Coregionalisation
LPF	Latent Parameter Function
LCC	Linear Combination Coefficient
SLFM	Semiparametric Latent Factor Model
CPM	Convolution Processes Model
NG	Natural Gradient
FNG	Fully Natural Gradient
GN	Gauss-Newton
ELBO	Evidence Lower Bound
NELBO	Negative ELBO
NLPD	Negative Log Predictive Density
LL	Log Likelihood
NLL	Negative Log Likelihood
CGP	Chained Gaussian Processes Model
CCGP	Correlated Chained Gaussian Processes Model
ACCGP	Augmented-output CCGP
MOGP	Multi-Output Gaussian Processes
HetMOGP	Heterogeneous Multi-Output Gaussian Processes Model
VIK	Variational Inducing Kernel
ZIP	Zero-inflated Poisson
HG	Heteroscedastic-Gaussian likelihood
Bt	Beta likelihood
Ga	Gamma likelihood
IF	Inducing Function
IID	Independent and Identically Distributed

Chapter 1

Introduction

Gaussian Processes (GPs) are flexible non-parametric distributions broadly used to provide prior information over non-linear functions (Rasmussen, 2006). They have been broadly applied in different scenarios, for instance: for improving sensor networks with missing signals (Osborne et al., 2008b; Osborne, 2013); in motion capture data for completing a sequence of missing frames (Zhao and Sun, 2016); in robotics, for estimating the system dynamics and provide a predictive distribution with uncertainty quantifications for planning movement trajectories (Chen et al., 2019); for natural language processing, where annotating linguistic data is often a complex and time consuming task, and consequently the GPs can learn from the outputs of multiple annotators and carry out said tasks (Cohn and Specia, 2013). They have been also used in computer emulation, where a Multiple-Output emulator can be used as a substitute of a computationally expensive deterministic model (Conti et al., 2009; Conti and O'Hagan, 2010); for learning the couplings between multiple time series and helping to enhance their forecasting capabilities (Boyle and Frean, 2005); and for modelling the temporal or spatial changes in gene expression sequences (BinTayyash et al., 2020).

Particularly, Gaussian Processes are a robust alternative for modelling parameters as latent functions in the context of probabilistic models. For instance, in a Generalised Linear Model (GLM) (Murphy, 2013) the likelihood's mean parameter can be chained to a GP latent function for providing a non-parametric modelling flexibility over such parameter (Duvenaud et al., 2011; Adam et al., 2016; Nguyen and Bonilla, 2014). Also, their use has been extended to modelling not only the likelihood's mean parameter, but each of the likelihood's parameters by chaining multiple GP latent functions; thus allowing to improve the predictive capabilities of the model due to appropriately capturing heteroscedasticity (Gujarati and Porter, 2009); i.e., the possibly non-constant

standard deviations throughout the data (Saul, 2016). There exist different ways to generate such GP latent functions, in this thesis we study three of them: 1. when each latent function follows an independent GP prior (Saul et al., 2016); 2. when each latent function is generated from a linear model of coregionalisation (LMC), i.e., a weighted sum of GP priors (Álvarez et al., 2012; Moreno-Muñoz et al., 2018); and 3. from convolution processes, i.e., a convolution integral between smoothing kernels and GP priors (Boyle and Frean, 2005; Álvarez et al., 2010; Álvarez and Lawrence, 2011). The above generative alternatives for the latent functions have been broadly used to model either a single or multiple outputs in diverse application scenarios (Álvarez et al., 2012); particularly, the independent GP priors have been used in applications that require the modelling of only a single output (Saul, 2016).

A probabilistic model based on any of the above generative GP latent functions involves issues of intractability for computing the posterior distribution, this is due to the general non-conjugate relationship between the likelihood and the GP priors. Since a posterior distribution is proportional to the likelihood's distribution multiplied by the prior distribution, such a non-conjugate relationship happens when the multiplication does not result in the form of an already known distribution. Therefore we can not directly access the posterior distribution of the model in a closed form. A way to deal with such intractability issues includes the use of variational inference, a method that transforms our model into an optimisation problem, it consists on finding a surrogate posterior distribution, also known as a variational posterior distribution, that best maximises a bound to the Log marginal likelihood. Though, using the variational inference approach for a GP model implies a necessity of dealing with issues regarding poor local optima solutions. Also, a single-output GP model presents problems of scalability caused by its high computational complexities, this is due to the need to invert a covariance matrix $\in \mathbb{R}^{N\times N}$, where N represents the number of data observation associated to the model. Commonly, the way to scale a GP model bases on the inducing variables framework (Quiñonero-Candela and Rasmussen, 2005; Snelson and Ghahramani, 2006). This is a framework that relies on the idea of augmenting the GP prior probability space, through the inclusion of a set of inducing points that change the full GP covariance matrix by a low-rank approximation (Snelson and Ghahramani, 2006; Titsias, 2009). Such inducing points help reducing significantly the GP's computational costs from $\mathcal{O}(N^3)$ to $\mathcal{O}(NM^2)$ and storage from $\mathcal{O}(N^2)$ to $\mathcal{O}(NM)$, where $M \ll N$ represents the number of inducing points (Rasmussen, 2006; Álvarez and Lawrence, 2011). Though the inducing variables framework significantly reduces the computa-

tional complexities, the use of a low-rank GP model can still be prohibitive in a context of large data observations. Likewise, in a multi-output Gaussian processes (MOGPs) context, such inducing points help reducing significantly the MOGP's computational costs from $\mathcal{O}(D^3N^3)$ to $\mathcal{O}(DNM^2)$ and storage from $\mathcal{O}(D^2N^2)$ to $\mathcal{O}(DNM)$, where D represents the number of outputs (Rasmussen, 2006; Álvarez and Lawrence, 2011).

The adequate performance of a GP model based on variational inference depends on a suitable optimisation process able to find rich local optima solutions for maximising a bound to the Log marginal likelihood. Variational GP models generally suffer from strong conditioning between the variational posterior distribution, the multiple hyper-parameters of the GP prior and the inducing points (Van der Wilk, 2018). For instance, a GP model based on LMC or convolutions processes depends on mathematical operations that include Q latent functions, where each latent function demands a treatment based on the inducing variables framework. Therefore, such strong conditionings are enhanced even more due to the dependence of inducing points per underlying latent function; and by the presence of additional hyper-parameters associated to the type of generative model for the GP latent functions that are chained to the likelihood's parameters. For the case of a MOGP model, the presence of multiple likelihoods augment such conditioning issues, even more when the outputs follow different statistical data types, i.e., we need to deal with the strong conditionings of the multiple parameters and hyper-parameters of a GP model with multiple non-conjugate heterogeneous likelihoods (Moreno-Muñoz et al., 2018). Accordingly, during the inference process of those types of GP models, stochastic gradient updates in combination with adaptive gradient methods (AGMs, e.g. Adam) tend to drive the optimisation to poor local minima.

With the purpose to overcome the optimisation problems present in variational GP models, there has recently been a growing interest in alternative optimisation schemes that adopt the natural gradient (NG) direction (Amari, 1998). For instance, in Hensman et al. (2013) the authors derived a mathematical analysis that suggested we can make better progress when optimising a variational GP along the NG direction, but without providing any experimental results of its performance. The authors in Khan et al. (2015) propose to linearise the non-conjugate terms of the model for admitting closed-form updates which are equivalent to optimising in the natural gradient direction. The work by Khan and Lin (2017) shows how to convert inference in non-conjugate models as it is done in the conjugate ones, by way of expressing the posterior distribution in the mean-parameter space. Furthermore, it shows that by means of ex-

ploiting the mirror descent algorithm (MDA) one can arrive to NG updates for tuning the variational posterior distribution. Those works coincide in improvements of training and testing performance, and also fast convergence rates. Nonetheless, they only show results in a full GP model where the kernel hyper-parameters are fixed using a grid search. On the other hand, the work by Salimbeni et al. (2018) does show a broad experimental analysis of the NG method for sparse GPs. The authors conclude that the NG is not prone to suffer from ill-conditioning issues in comparison to the AGMs. Also the NG has been used to ease optimisation of the variational posterior over the latent functions of a deep GP model (Salimbeni et al., 2019). However, in those two latter cases the NG method only applies for the latent functions' posterior parameters, while an Adam method performs a cooperative optimisation for dealing with the hyper-parameters and inducing points. The authors in Salimbeni et al. (2018) call this strategy a hybrid between NG and Adam, and termed it NG+Adam.

In this thesis we address the problems of intractability, scalability, and poor local optima solutions associated to non-conjugate likelihood Gaussian process models. Particularly, in this thesis we address the aforementioned issues in a context where the likelihood's parameters are modelled as latent parameter functions drawn from correlated Gaussian processes. We focus on three ways to generate such latent parameter functions: 1. from a linear model of coregionalisation (Journel and Huijbregts, 1979); 2. from convolution processes (Boyle and Frean, 2005), i.e., a convolution integral between smoothing kernels and Gaussian process priors; and 3. using variational inducing kernels (Álvarez et al., 2010), an alternative form to generate the latent parameter functions through the convolution processes formalism, by using a double convolution integral. We borrow ideas from different variational optimisation mechanisms like Staines and Barber (2013); Khan et al. (2017a), and Khan and Lin (2017); Khan et al. (2018), that consist on introducing a variational (or exploratory) distribution over the model so as to build objective functions that: allow us to deal with intractability as well as enable scalability when handling massive amounts of data observations. Also, such optimisations mechanisms grant us to perform inference of the model hyper-parameters together with the posterior's parameters through a fully natural gradient optimisation; a useful scheme for tackling the problem of poor local optima solutions.

Outline of the thesis and contributions

- This thesis is built upon the idea of using various optimisation mechanisms for dealing with intractability issues present in non-conjugate likelihood Gaussian

process models, i.e., GP models where the likelihood function and the GP priors are non-conjugate. Such optimisation mechanisms help us to construct scalable objective functions that grant us the use of our correlated GP models in scenarios with large amounts of data observations. Likewise, we benefit from said mechanisms for improving the inference processes of the different GP models presented in this thesis. Thus, in chapter 2, we introduce such optimisation mechanisms.

- In chapter 3, we introduce a Single-Output model that assumes by construction that the likelihood's parameters follow multiple GP priors that can be correlated through a linear model of coregionalisation; we termed it as the Correlated Chained Gaussian Processes (CCGP) model. Our approach bases on the so-called inducing variables framework and scales by means of stochastic variational inference. We run experiments in different real databases, we show that our method, based on an LMC, generally achieves richer predictive distributions that better quantify the uncertainty than the classical setting of a Chained Gaussian Processes model that builds upon independent GP priors.

- Also, in chapter 3, we propose a strategy of model training that augments a single-output GP in order to treat it as a multi-output one. We found that such strategy enhances the generalisation properties of the model accomplishing a high predictive performance.

- In chapters 3, 4 and 5 we propose a fully natural gradient (FNG) scheme for jointly tuning the hyper-parameters, inducing points and variational posterior parameters of the single-output CCGP model, and its multi-output version when having heterogeneous outputs. To this end, we borrow ideas from different variational optimisation (VO) mechanisms (or strategies) like Staines and Barber (2013); Khan et al. (2017a) and Khan et al. (2018), by introducing an exploratory distribution over the hyper-parameters and inducing points. Such VO strategies have shown to be successful exploratory-learning tools able to avoid poor local optima solutions; they have been broadly studied in the context of reinforcement and Bayesian deep learning, but not much in the context of GPs.

- In chapter 5, we provide an extension of the Heterogeneous MOGP (HetMOGP) based on a Convolution Processes model (CPM), rather than an LMC approach as in the original model by Moreno-Muñoz et al. (2018). This is a novel contribution since there are no former MOGP models with convolution processes that involve stochastic variational inference (SVI), nor a model of heterogeneous

outputs that relies on convolution processes. Likewise, we provide a FNG scheme for optimising the new model extension, the HetMOGP with CPM.

- To the best of our knowledge the NG method has not been performed over any MOGP model before. Hence, in this work we also contribute to show how a NG method used in a full scheme over the MOGP's parameters and kernel hyper-parameters alleviates the strong conditioning problems. This, by achieving better local optima solutions with higher test performance rates than Adam and stochastic gradient descent. Also, we explore for the first time in a MOGP model the behaviour of the hybrid strategy NG+Adam, and provide comparative results to our proposed scheme.

- In chapter 6, we apply the correlated chained GP models based on a linear model of coregionalisation and convolution processes for modelling the citizens mobility in the Chinese city of Guangzhou, through the use of a ZIP likelihood. To the best of our knowledge, a ZIP likelihood has not been previously implemented together with a GP model. Unlike previous works based on GPs that mainly model the mean parameter of the likelihood with a unique GP prior, here we propose that each of those likelihood's parameters are modelled as Latent Parameter Functions that follow correlated GPs as detailed in chapters 3, 4 and 5; thus, allowing a higher flexibility to model heteroscedasticity.

- Also in chapter 6, we derive an SVI framework that allow us to use two types of convolution process models in the context of large datasets: 1. CCGP with a convolution processes model, and 2. CCGP with Variational Inducing Kernels. Former works have not developed GP models based on a Convolution Processes Model and Variational Inducing Kernels for other type of likelihoods beyond a Gaussian. In this work, we derive equations that can be used for any type of likelihood. Particularly, we provide results for both CCGP models based on a Convolution Processes Model and Variational Inducing Kernels for ZIP and Poisson likelihoods.

List of publications

The main contributions of this thesis have been presented in the following papers:

(i) J. Giraldo and M. A. Álvarez. "A Fully Natural Gradient Scheme for Improving Inference of the Heterogeneous Multi-Output Gaussian Process Model". IEEE

Transactions on Neural Networks and Learning Systems (IEEE TNNLS), **Submitted on November 2019**. See reprint on Appendix K.

(ii) J. Giraldo, J. Zhang, and M. A. Álvarez. "Correlated Chained Gaussian Processes for Modelling Citizens Mobility using a Zero-Inflated Poisson Likelihood". IEEE Transactions on Intelligent Transportation Systems (IEEE TITS), **Submitted on March 2021**. See reprint on Appendix M.

(iii) J. Gil, J. Giraldo, A. Álvarez-Meza, A. Orozco-Gutiérrez, and M. A. Álvarez. "Correlated Chained Gaussian Processes for Datasets with Multiple Annotators". IEEE Transactions on Neural Networks and Learning Systems (IEEE TNNLS), **Submitted on January 2021**. See reprint on Appendix L.

In papers (i) and (ii), Giraldo had the main responsibility for writing the principal drafts and coding the software. The co-authors provided revisions of the drafts which were incorporated in the final versions by Giraldo. For paper (iii), Gil was responsible for writing the first draft and build the mathematical likelihood expressions for multiple annotators, Giraldo contributed with the mathematical correlated Gaussian process priors structure and revision of the draft; both Gil and Giraldo developed the software for the paper: Gil programmed the software for the multiple annotators' likelihood function with its mathematical derivatives and Giraldo coded the software for the Gaussian Process latent parameter functions with its mathematical derivatives. Paper (i) was supervised by Álvarez, and Paper (ii) was supervised by Zhang and Álvarez.

Chapter 2

Mechanisms for Optimisation

This chapter introduces various mechanisms for improving optimisation of an objective function. We introduce the Variational Optimisation method as a mechanism for introducing exploration in the parameter space of an objective function by means of a free parametrised variational distribution (Staines and Barber, 2013). Also, we show how Variational Inference (VI) can be seen as a particular case of variational optimisation. Another optimisation mechanism is the Mirror Descent Algorithm (Khan and Lin, 2017), which allows to easily derive natural gradient updates (Amari, 1998) of the variational posterior parameters by solving iterative sub-problems in the mean-space of the variational distribution. Likewise, we describe the Variational Adaptive-Newton (VAN) (Khan et al., 2017a), a method that benefits from a Gaussian posterior distribution to easily express the parameters updates in the NG direction. And we also present the concept of *natural-momentum* (Khan et al., 2018), which takes advantage of the KL divergence for providing extra memory information to the iterative sub-problems of the mirror descent algorithm.

2.1 Variational Optimisation

The goal in optimisation is to find a proper set of parameters that minimise a possibly non-convex function $g(\boldsymbol{\theta})$ by solving, $\boldsymbol{\theta}^* = \arg\min_{\boldsymbol{\theta}} g(\boldsymbol{\theta})$, where $\boldsymbol{\theta}^*$ represents the set of parameters that minimise the function. The classical way to deal with the above optimisation problem involves deriving the objective w.r.t $\boldsymbol{\theta}$ and solving in a closed-form, or through a gradient descent method. Usually, gradient methods tend to converge to the closest local minima from the starting point without exploring much the space of solutions (Chong and Zak, 2013) (see Appendix A for an example when

using the optimisation Newton's method). Alternatively the variational optimisation method proposes to solve the same problem (Staines and Barber, 2013), but introducing exploration in the parameter space of a variational (or exploratory) distribution $q(\boldsymbol{\theta}|\boldsymbol{\xi})$ by bounding the function $g(\boldsymbol{\theta})$ as follows:

$$\min_{\boldsymbol{\theta}} g(\boldsymbol{\theta}) \leq \mathbb{E}_{q(\boldsymbol{\theta}|\boldsymbol{\xi})}[g(\boldsymbol{\theta})] := \tilde{\mathcal{L}}(\boldsymbol{\xi}), \tag{2.1}$$

where $\boldsymbol{\xi}$ represents a set of variables that parametrise the distribution $q(\boldsymbol{\theta}|\boldsymbol{\xi})$, and $\tilde{\mathcal{L}}(\boldsymbol{\xi})$ is an upper bound to the function $g(\boldsymbol{\theta})$. Therefore the main goal is to minimize the above equation w.r.t the new set $\boldsymbol{\xi}$. If we start an optimisation process to solve the last problem using an exploratory Gaussian distribution $q(\boldsymbol{\theta}|\boldsymbol{\mu}, \boldsymbol{\Sigma})$, where the set $\boldsymbol{\xi}$ is composed by the mean $\boldsymbol{\mu}$ and covariance $\boldsymbol{\Sigma}$, we would expect the following: at the beginning of such optimisation process our covariance initialisation should be $\boldsymbol{\Sigma} \neq \mathbf{0}$, meaning that the space of solutions can be explored around the initial mean $\boldsymbol{\mu}$ (Wierstra et al., 2014); after the optimisation time elapses, the mean $\boldsymbol{\mu}$ will be approaching to a local minima $\boldsymbol{\theta}^*$ that best reduces the expectation in Eq. (2.1), while the covariance $\boldsymbol{\Sigma}$ will be collapsing to zero ($\boldsymbol{\Sigma} \to \mathbf{0}$). Thereby the exploratory distribution will become a Dirac's delta $q(\boldsymbol{\theta}) = \delta(\boldsymbol{\theta} - \boldsymbol{\mu})$, where $\boldsymbol{\mu} = \boldsymbol{\theta}^*$ (Hensman et al., 2015b).

With the aim to better understand the behaviour of the exploratory distribution, let us introduce an experiment inspired by the one in Khan et al. (2017a); we define $g(\theta) = 2\exp(-0.09\theta^2)\sin(4.5\theta)$, a function with multiple local minima, and an exploratory distribution over θ, $q(\theta) = \mathcal{N}(\theta|\mu, \sigma^2)$, with parameters mean μ and variance σ^2. We built the graph of Figure 2.1 to show what happens at each iteration of the optimisation process for $g(\theta)$; we present three perspectives of such an experiment, where we initialise the parameters $\theta = \mu = -3.0$ and $\sigma = 3.0$. We can notice from Figure 2.1 that the initial value of $\theta = \mu = -3.0$ is close the the poor minimum at $\theta \approx -3.114$ and far away from better minima solutions like the one at $\theta \approx -1.729$ and $\theta \approx -0.346$ (the global minimum). Looking at the third row in Figure 2.1, we realise that, when the inference process starts, the exploratory distribution $q(\theta)$ modifies its variance and moves its mean towards a better region in the space of θ. We can also see that $q(\theta)$ initially behaves as a broad distribution (in light-gray colour) with a mean located at $\mu = -3.0$, while the iterations elapse, the distribution $q(\theta)$ modifies its shape in order to reach a better local minima solution (at $\mu \approx -1.729$). The distribution presents such behaviour in spite of being closer to other poor local minima like the one between the interval $(-4, -3)$. Additionally, when the mean μ is close to the local minimum at $\theta \approx -1.729$, the variance parameter reduces constantly making the distribution look narrower, which means the variance parameter tends to collapse to zero ($\sigma^2 \to 0$)

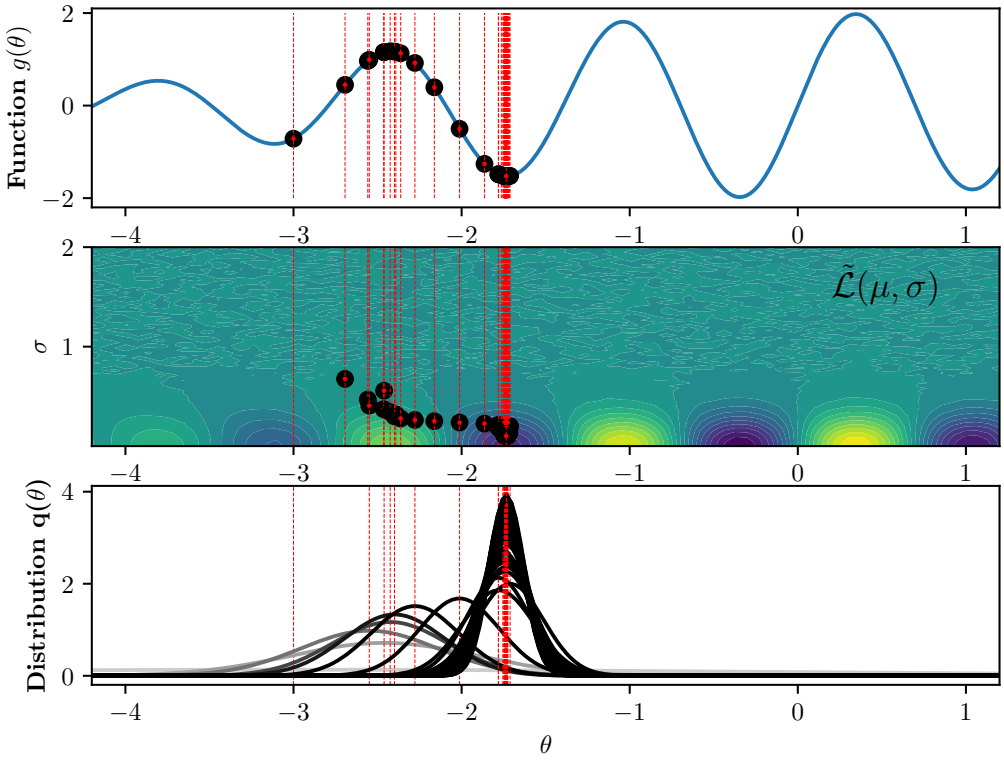


Figure 2.1: *First row shows what happens from the perspective of the original function* $g(\theta) = 2\exp(-0.09\theta^2)\sin(4.5\theta)$*, the black dots represent the position of* $\theta = \mu$ *at each iteration. Second row shows a contour graph of the space of solutions w.r.t* σ *and* μ*, here the black dots refer to the position of* σ *and* μ *at each iteration, and the low and high colour intensities relate to low and high values of* $\mathbb{E}_{q(\theta)}[g(\theta)]$*. Third row shows* $q(\theta)$*'s behaviour, for each Gaussian bell we use a colour code from light-gray to black for representing initial to final stages of the inference. All sub-graphs present vertical lines for aligning iterations, i.e., from left to right the lines represent the occurrence of an iteration. To avoid excessive overlapping, the third row only shows* $q(\theta)$ *every two iterations.*

increasing the certainty of the solution. This behaviour implies that in the long term the distribution will become a Dirac's delta $q(\theta) = \delta(\theta - \mu)$, where $\mu = \theta$. Therefore, a feasible minima solution for the original objective function $g(\theta)$ is $\theta = \mathbb{E}_{q(\theta)}[\theta] = \mu$. This can be seen in the first sub-graph where at each iteration $\theta = \mu$; in fact, at the end of the optimisation, μ is fairly close to the value $\theta \approx -1.729$, a local minima. Though we could notice, in Figure 2.1, an exploratory behaviour of $q(\theta)$ that helped avoiding the poor local minima at $\theta \approx -3.114$, the rapid collapsing effect of the variance parameter limits the exploration of θ's space. In the next subsection we will describe how to reduce such a collapsing effect of $q(\theta)$ and gain additional exploration by introducing a Kullback-Leibler (KL) diverge penalisation to Eq. (2.1) (see Appendix A where we reproduced the same example of the function $g(\theta) = 2\exp(-0.09\theta^2)\sin(4.5\theta)$, but optimising with the Newton's method).

2.2 Variational Optimisation with Penalisation

The work of VO by Staines and Barber (2013) does not introduce the KL term in the equation (2.1), i.e. $\tilde{\mathcal{L}}(\boldsymbol{\xi}) = \mathbb{E}_{q(\boldsymbol{\theta}|\boldsymbol{\xi})}[g(\boldsymbol{\theta})]$, this implies that during an inference process, the exploratory distribution is free to collapse to zero becoming a Dirac's delta $q(\boldsymbol{\theta}) = \delta(\boldsymbol{\theta} - \boldsymbol{\mu})$, where $\boldsymbol{\mu} = \boldsymbol{\theta}^*$ and $\boldsymbol{\mu}$ represents the $q(\boldsymbol{\theta})$'s mean (Wierstra et al., 2014; Hensman et al., 2015b). The covariance's collapsing behaviour is an indicator of how the exploration reduces while the objective is converging to a local minimum. Nevertheless, this collapsing effect limits the exploration of $\boldsymbol{\theta}$'s space. To gain wider exploration, we can avoid $\boldsymbol{\Sigma}$ to collapse by imposing a regularization term to the latter bound in Eq. (2.1):

$$\tilde{\mathcal{L}}(\boldsymbol{\xi}) = \mathbb{E}_{q(\boldsymbol{\theta}|\boldsymbol{\xi})}[g(\boldsymbol{\theta})] + \mathbb{D}_{KL}\big(q(\boldsymbol{\theta}|\boldsymbol{\xi})||p(\boldsymbol{\theta})\big), \tag{2.2}$$

where $\mathbb{D}_{KL}(\cdot||\cdot)$ is a Kullback-Leibler divergence that forces the exploratory distribution $q(\boldsymbol{\theta}|\boldsymbol{\xi})$ to trade-off between minimising the expectation $\mathbb{E}_{q(\boldsymbol{\theta}|\boldsymbol{\xi})}[g(\boldsymbol{\theta})]$ and not going far away from the imposed $p(\boldsymbol{\theta})$ penalization (Khan et al., 2017b). Indeed, the KL term in Eq. (2.2) reduces the collapsing effect of $q(\boldsymbol{\theta})$ and helps to gain additional exploration when an inference process is carried out.

In order to better understand the behaviour when introducing the KL divergence, we follow the same experiment used in the previous section to optimise the function $g(\theta) = 2\exp(-0.09\theta^2)\sin(4.5\theta)$, a function with multiple local minima. Figure 2.2 shows three perspectives of a such experiment, where we initialise the parameters

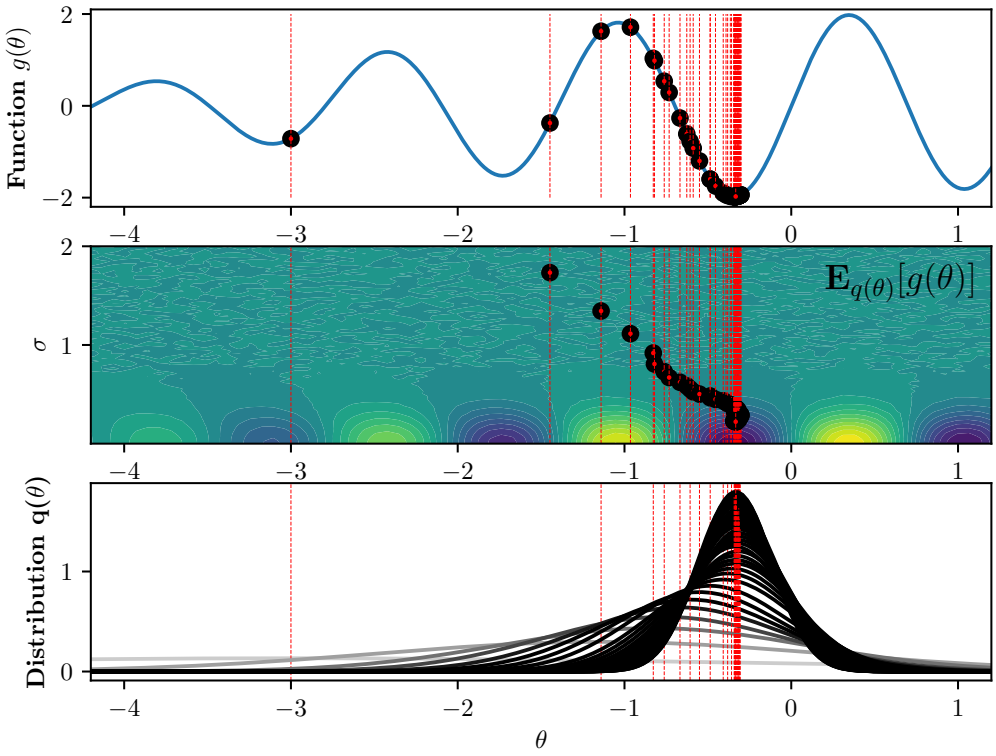


Figure 2.2: *First row shows what happens from the perspective of the original function* $g(\theta) = 2\exp(-0.09\theta^2)\sin(4.5\theta)$*, the black dots represent the position of* $\theta = \mu$ *at each iteration. Second row shows a contour graph of the space of solutions w.r.t* σ *and* μ*, here the black dots refer to the position of* σ *and* μ *at each iteration, and the low and high colour intensities relate to low and high values of* $\mathbb{E}_{q(\theta)}[g(\theta)]$*, notice that here we do not include the KL term information for easing the visualisation of the multiple local minima. Third row shows* $q(\theta)$*'s behaviour, for each Gaussian bell we use a colour code from light-gray to black for representing initial to final stages of the inference. All sub-graphs present vertical lines for aligning iterations, i.e., from left to right the lines represent the occurrence of an iteration. To avoid excessive overlapping, the third row only shows* $q(\theta)$ *every two iterations.*

$\theta = \mu = -3.0$ and $\sigma = 3.0$, and $p(\theta) = \mathcal{N}(\theta|0, \lambda^{-1})$ with $\lambda = 1.0$. We can notice from Figure 2.2 that the initial value of $\theta = \mu = -3.0$ is far away from $g(\theta)$'s global minimum at $\theta \approx -0.346$. When the inference process starts, the exploratory distribution $q(\theta)$ modifies its variance and moves its mean towards a better region in the space of θ. From the third row we can also see that $q(\theta)$ initially behaves as a broad distribution (in light-gray colour) with a mean located at $\mu = -3.0$, while the iterations elapse, the distribution $q(\theta)$ modifies its shape in order to reach a better local minima solution (at $\mu \approx -0.346$). The distribution presents such behaviour in spite of being closer to other poor local minima like the ones between the intervals $(-4, -3)$ and $(-2, -1)$. Additionally, when the mean μ is close to $\theta \approx -0.346$ (the global minimum), the variance parameter reduces constantly making the distribution look narrower, which means it is increasing the certainty of the solution. This behaviour implies that in the long term $q(\theta)$'s mean will be much closer to θ^*. Therefore, a feasible minima solution for the original objective function $g(\theta)$ is $\theta = \mathbb{E}_{q(\theta)}[\theta] = \mu$. This can be seen in the first sub-graph where at each iteration $\theta = \mu$, in fact, at the end μ is fairly close to the value $\theta \approx -0.346$.[1]

2.3 Variational Inference: VO for the Negative Log Likelihood

A common way to build a probabilistic model for a set of observations $\mathbf{X} = \{\mathbf{x}_n\}_{n=1}^N \in \mathbb{R}^{N \times P}$ is to assume that each observation is drawn independently and identically distributed (IID) from a probability distribution $p(\mathbf{X}|\boldsymbol{\theta})$, commonly known as a likelihood. Fitting the model consists on finding the parameter $\boldsymbol{\theta}$ that makes the distribution appropriately explain the data. This inference process is called maximum likelihood estimation, given that is equivalent to the optimisation problem of maximising the log likelihood function $\log p(\mathbf{X}|\boldsymbol{\theta})$, i.e., minimising the negative log likelihood (NLL) function $-\log p(\mathbf{X}|\boldsymbol{\theta})$ (Murphy, 2013). From a Bayesian perspective, we can introduce a prior distribution $p(\boldsymbol{\theta})$ over the parameter of interest, which implies that there also exists a posterior distribution, $p(\boldsymbol{\theta}|\mathbf{X}) \propto p(\mathbf{X}|\boldsymbol{\theta})p(\boldsymbol{\theta})$, over such parameter, useful to

[1] Given that in practise we usually do not have any idea about the landscape of the objective functions that we are interested in optimising, then our suggestion for a practitioner is to initialise the penalisation distribution $p(\theta) = \mathcal{N}(\theta|0, \lambda^{-1})$ with $\lambda = 1.0$; this value presents a stable performance in diverse types of landscapes, or use even a smaller value if more aggressive exploration is desired. (see Appendix B for a detailed analysis of the influence of λ during optimisation).

render future predictions of the model. When the likelihood and prior are conjugate, the posterior distribution can be computed in closed form, but that is not always the case. Hence, if the likelihood and prior are non-conjugate, it is necessary to approximate the posterior (Bishop, 2006). Variational inference is a powerful framework broadly used in machine learning, that allows to estimate the posterior distribution by minimising the KL divergence $\mathbb{D}_{KL}\big(q(\boldsymbol{\theta}|\boldsymbol{\xi})||p(\boldsymbol{\theta}|\mathbf{X})\big)$ between an approximate variational posterior $q(\boldsymbol{\theta}|\boldsymbol{\xi})$ and the posterior distribution $p(\boldsymbol{\theta}|\mathbf{X})$ (Blei et al., 2017). Since we do not have access to the posterior, minimising such KL divergence is equivalent to maximising a lower bound to the marginal likelihood. It emerges from the equality: $\log \mathbb{E}_{q(\boldsymbol{\theta}|\boldsymbol{\xi})}\left[\frac{p(\mathbf{X}|\boldsymbol{\theta})p(\boldsymbol{\theta})}{q(\boldsymbol{\theta}|\boldsymbol{\xi})}\right] = \log p(\mathbf{X})$, in which, after applying the Jensen's inequality we arrive to,

$$-\tilde{\mathcal{L}}(\boldsymbol{\xi}) = \mathbb{E}_{q(\boldsymbol{\theta}|\boldsymbol{\xi})}\left[\log \frac{p(\mathbf{X}|\boldsymbol{\theta})p(\boldsymbol{\theta})}{q(\boldsymbol{\theta}|\boldsymbol{\xi})}\right] \leq \log p(\mathbf{X}), \tag{2.3}$$

where $\log p(\mathbf{X})$ represents the log marginal likelihood and $-\tilde{\mathcal{L}}(\boldsymbol{\xi})$ is an evidence lower bound (ELBO) (Jordan et al., 1999). It is noteworthy that if we replace $g(\boldsymbol{\theta}) = -\log p(\mathbf{X}|\boldsymbol{\theta})$ in Eq. (2.2), we end up with exactly the same lower bound of Eq. (2.3). Therefore, VI can be seen as a particular case of VO with a KL divergence penalisation, where the objective $g(\boldsymbol{\theta})$ is nothing but the NLL. We can distinguish from two perspectives when using VO for maximum likelihood: from the Bayesian perspective we are not only interested in a point estimate for the parameter $\boldsymbol{\theta}$, but in the uncertainty codified in $q(\boldsymbol{\theta})$'s (co)variance for making future predictions; and from the non-Bayesian perspective the main goal in maximum likelihood estimation is to optimise the function $g(\boldsymbol{\theta}) = -\log p(\mathbf{X}|\boldsymbol{\theta})$. For this case, if $q(\boldsymbol{\theta}|\boldsymbol{\mu}, \boldsymbol{\Sigma})$ is a Gaussian distribution, we can make use of only the posterior's mean $\mathbb{E}_{q(\boldsymbol{\theta}|\boldsymbol{\xi})}[\boldsymbol{\theta}] = \boldsymbol{\mu}$ as a feasible solution for $\boldsymbol{\theta}^*$ without taking into account the uncertainty. This is also known as the maximum a posteriori (MAP) solution in the context of VI, due to the fact that $\boldsymbol{\theta}_{\text{MAP}} = \arg\max_{\boldsymbol{\theta}} p(\boldsymbol{\theta}|\mathbf{X}) \approx q(\boldsymbol{\theta}|\boldsymbol{\mu}, \boldsymbol{\Sigma})$, where the maximum of the distribution $q(\boldsymbol{\theta}|\boldsymbol{\mu}, \boldsymbol{\Sigma})$ is located at its mean, thereby $\boldsymbol{\theta}_{\text{MAP}} = \boldsymbol{\mu}$ (Bishop, 2006) (see Appendix H for details on MAP in the context of VI).

2.4 Exploiting the Mirror Descent Algorithm

Direct update equations for the parameters of a (posterior) distribution using natural gradients involve the inversion of a Fisher information matrix, which in general it

is computationally complex to do. The purpose of this section is to show how an alternative formulation of the NG updates can be derived from the MDA (Khan and Lin, 2017). We describe the Variational Adaptive-Newton, a method that benefits from a Gaussian posterior distribution to easily express the parameters updates in the NG direction. And we also detail the concept of *natural-momentum* which takes advantage of the KL divergence for providing extra memory information to the iterative sub-problems of the MDA (Khan et al., 2018).

2.4.1 Connection between Natural-Gradient and Mirror Descent

The NG allows to solve an optimisation problem like the one in Eq. (2.2), where the goal consists on finding an optimal distribution $q(\boldsymbol{\theta})$ that best minimises the objective bound (Amari, 1998). The method takes advantage of the inverse Fisher information matrix, $\mathbf{F}^{-1}$, associated to the random variable $\boldsymbol{\theta}$, by iteratively weighting the gradient updates, $\boldsymbol{\xi}_{t+1} = \boldsymbol{\xi}_t - \alpha_t \mathbf{F}_t^{-1} \hat{\nabla}_{\boldsymbol{\xi}} \tilde{\mathcal{L}}_t$, where α_t is a positive step-size parameter and $\boldsymbol{\xi}_t$ represents the natural (or canonical) parameters of the distribution $q(\boldsymbol{\theta})$. Such natural parameters can be better noticed by expressing the distribution in the general form of the exponential family,

$$q(\boldsymbol{\theta}) = h(\boldsymbol{\theta}) \exp\big(\langle \boldsymbol{\xi}, \phi(\boldsymbol{\theta}) \rangle - A(\boldsymbol{\xi})\big),$$

where $A(\boldsymbol{\xi})$ is the log-partition function, $\phi(\boldsymbol{\theta})$ is a vector of sufficient statistics and $h(\boldsymbol{\theta})$ is a scaling constant (Murphy, 2013). Such exponential family relies on the equation above in order to gather a parametric set of probability distributions, it includes the distributions: Gamma, Exponential, Beta, Bernoulli, Normal (or Gaussian), Poisson, etc (Bishop, 2006). For instance, one might express the Normal distribution, $q(\boldsymbol{\theta}) = \mathcal{N}(\boldsymbol{\theta}|\boldsymbol{\mu}, \boldsymbol{\Sigma})$, in said exponential family by having a set of natural parameters $\boldsymbol{\xi} = \{\boldsymbol{\Sigma}^{-1}\boldsymbol{\mu}, -\frac{1}{2}\boldsymbol{\Sigma}^{-1}\}$; a log-partition function $A(\boldsymbol{\xi}) = \frac{1}{2}\left(\boldsymbol{\mu}^T\boldsymbol{\Sigma}^{-1}\boldsymbol{\mu} + \ln|\boldsymbol{\Sigma}|\right)$; a vector of sufficient statistics $\phi(\boldsymbol{\theta}) = \{\boldsymbol{\theta}, \boldsymbol{\theta}\boldsymbol{\theta}^T\}$; and a scaling constant $h(\boldsymbol{\theta}) = \frac{1}{(2\pi)^{P/2}}$. With respect to the NG updates, $\boldsymbol{\xi}_{t+1} = \boldsymbol{\xi}_t - \alpha_t \mathbf{F}_t^{-1} \hat{\nabla}_{\boldsymbol{\xi}} \tilde{\mathcal{L}}_t$, it is worth mentioning that they are expensive due to involving the computation of the inverse Fisher matrix at each iteration. Since an exponential-family distribution has an associated set of mean-parameters $\boldsymbol{\eta} = \mathbb{E}[\boldsymbol{\phi}(\boldsymbol{\theta})]$, then an alternative way to induce the NG updates consists on formulating a MDA in such mean-parameter space. Hence, the algorithm bases on

solving the following iterative sub-problems:

$$\boldsymbol{\eta}_{t+1} = \arg\min_{\boldsymbol{\eta}} \langle \boldsymbol{\eta}, \hat{\nabla}_{\boldsymbol{\eta}} \tilde{\mathcal{L}}_t \rangle + \frac{1}{\alpha_t} \mathbb{D}_{KL}(q(\boldsymbol{\theta}) || q_t(\boldsymbol{\theta})), \tag{2.4}$$

where $\boldsymbol{\eta}$ is the set of $q(\boldsymbol{\theta})$'s mean-parameters, $\tilde{\mathcal{L}}$ is a VO bound of a function $g(\boldsymbol{\theta})$, $\hat{\nabla}_{\boldsymbol{\eta}} \tilde{\mathcal{L}}_t := \hat{\nabla}_{\boldsymbol{\eta}} \tilde{\mathcal{L}}(\boldsymbol{\eta}_t)$ denotes a stochastic gradient, $q_t(\boldsymbol{\theta}) := q(\boldsymbol{\theta}|\boldsymbol{\eta}_t)$ and α_t is a positive step-size parameter (Khan and Lin, 2017). The intention of the above formulation is to exploit the parametrised distribution's structure by controlling its divergence w.r.t its older state $q_t(\boldsymbol{\theta})$. Replacing the distribution $q(\boldsymbol{\theta})$ in its exponential-form, in the above KL divergence, and setting Eq. (2.4) to zero, let us express,

$$\langle \boldsymbol{\eta}, \hat{\nabla}_{\boldsymbol{\eta}} \tilde{\mathcal{L}}_t \rangle + \frac{1}{\alpha_t} \big[\langle \boldsymbol{\xi}, \boldsymbol{\eta} \rangle - A(\boldsymbol{\xi}) - \langle \boldsymbol{\xi}_t, \boldsymbol{\eta} \rangle + A(\boldsymbol{\xi}_t) \big] = 0,$$

and by deriving w.r.t $\boldsymbol{\eta}$, we arrive to $\boldsymbol{\xi}_{t+1} = \boldsymbol{\xi}_t - \alpha_t \hat{\nabla}_{\boldsymbol{\eta}} \tilde{\mathcal{L}}_t$, where $\boldsymbol{\xi}_{t+1} := \boldsymbol{\xi}$ and $\hat{\nabla}_{\boldsymbol{\eta}} \tilde{\mathcal{L}}_t = \mathbf{F}^{-1} \hat{\nabla}_{\boldsymbol{\xi}} \tilde{\mathcal{L}}_t$ as per the work in Raskutti and Mukherjee (2015), where the authors provide a formal proof of such equivalence. The formulation in Eq. (2.4) is advantageous since it is easier to compute derivatives w.r.t $\boldsymbol{\eta}$ than computing the inverse Fisher information matrix $\mathbf{F}^{-1}$. Therefore, the MDA for solving iterative sub-problems in the mean-parameter space is equivalent to updating the canonical parameters in the NG direction (see Appendix C for more details).

2.4.2 Variational Adaptive-Newton and Natural-Momentum

The VAN method aims to solve the problem in Eq. (2.4) using a Gaussian distribution $q(\boldsymbol{\theta}) := q(\boldsymbol{\theta}|\boldsymbol{\mu}, \boldsymbol{\Sigma})$ as the exploratory mechanism for optimisation (Khan et al., 2017a). This implies that if $\boldsymbol{\mu}$ and $\boldsymbol{\Sigma}$ represent the mean and covariance, respectively, then $q(\boldsymbol{\theta})$'s mean-parameters are $\boldsymbol{\eta} = \{\boldsymbol{\mu}, \boldsymbol{\Sigma} + \boldsymbol{\mu}\boldsymbol{\mu}^\top\}$, and also its analogous natural-parameters are $\boldsymbol{\xi} = \{\boldsymbol{\Sigma}^{-1}\boldsymbol{\mu}, -\frac{1}{2}\boldsymbol{\Sigma}^{-1}\}$. When linking these parametrisations and solving for the MDA in Eq. (2.4), we end up with the following updates:

$$\begin{aligned} \boldsymbol{\Sigma}_{t+1}^{-1} &= \boldsymbol{\Sigma}_t^{-1} + 2\alpha_t \hat{\nabla}_{\boldsymbol{\Sigma}} \tilde{\mathcal{L}}_t, \\ \boldsymbol{\mu}_{t+1} &= \boldsymbol{\mu}_t - \alpha_t \boldsymbol{\Sigma}_{t+1} \hat{\nabla}_{\boldsymbol{\mu}} \tilde{\mathcal{L}}_t, \end{aligned}$$

where $\boldsymbol{\mu}_t$ and $\boldsymbol{\Sigma}_t$ are the mean and covariance parameters at the instant t respectively; the stochastic gradients are $\hat{\nabla}_{\boldsymbol{\mu}} \tilde{\mathcal{L}}_t := \hat{\nabla}_{\boldsymbol{\mu}} \tilde{\mathcal{L}}(\boldsymbol{\mu}_t, \boldsymbol{\Sigma}_t)$ and $\hat{\nabla}_{\boldsymbol{\Sigma}} \tilde{\mathcal{L}}_t := \hat{\nabla}_{\boldsymbol{\Sigma}} \tilde{\mathcal{L}}(\boldsymbol{\mu}_t, \boldsymbol{\Sigma}_t)$. These latter updates represent a NG descent algorithm for exploring the space of solutions of

the variable $\boldsymbol{\theta}$ through a Gaussian distribution (Khan and Lin, 2017). It is possible to keep exploiting the structure of the distribution $q(\boldsymbol{\theta})$, this by including an additional KL divergence term in the MDA of Eq. (2.4) as follows:

$$\boldsymbol{\eta}_{t+1} = \arg\min_{\boldsymbol{\eta}} \langle \boldsymbol{\eta}, \hat{\nabla}_{\boldsymbol{\eta}} \tilde{\mathcal{L}}_t \rangle + \frac{1}{\tilde{\alpha}_t} \mathrm{KL}(\boldsymbol{\theta})_t - \frac{\tilde{\gamma}_t}{\tilde{\alpha}_t} \mathrm{KL}(\boldsymbol{\theta})_{t-1}, \tag{2.5}$$

where $q_t(\boldsymbol{\theta}) := q(\boldsymbol{\theta}|\boldsymbol{\mu}_t, \boldsymbol{\Sigma}_t)$ represents the exploratory distributions $q(\boldsymbol{\theta})$ with the parameters obtained at time t, and $\mathrm{KL}(\cdot)_t := \mathbb{D}_{KL}(q(\cdot)||q_t(\cdot))$. Such additional KL term, called as a *natural-momentum* by Khan et al. (2018), provides extra memory information to the MDA for potentially improving its convergence rate. This momentum can be controlled by the relation between the positive step-sizes $\tilde{\alpha}_t$ and $\tilde{\gamma}_t$. When solving for Eq. (2.5), we arrive to the following NG update equations:

$$\boldsymbol{\Sigma}_{t+1}^{-1} = \boldsymbol{\Sigma}_t^{-1} + 2\alpha_t \hat{\nabla}_{\boldsymbol{\Sigma}} \tilde{\mathcal{L}}_t \tag{2.6}$$

$$\boldsymbol{\mu}_{t+1} = \boldsymbol{\mu}_t - \alpha_t \boldsymbol{\Sigma}_{t+1} \hat{\nabla}_{\boldsymbol{\mu}} \tilde{\mathcal{L}}_t + \gamma_t \boldsymbol{\Sigma}_{t+1} \boldsymbol{\Sigma}_t^{-1} (\boldsymbol{\mu}_t - \boldsymbol{\mu}_{t-1}), \tag{2.7}$$

where $\alpha_t = \tilde{\alpha}_t/(1-\tilde{\gamma}_t)$ and $\gamma_t = \tilde{\gamma}_t/(1-\tilde{\gamma}_t)$ are positive step-size parameters (Khan et al., 2017b, 2018).

2.5 Summary

In this chapter, we have introduced various mechanisms for improving optimisation of an objective function, like the variational optimisation method as a mechanism for inducing exploration in the parameter space of an objective function, by means of a free parametrised variational distribution. Also, we showed how variational inference can be seen as a particular case of variational optimisation. Another optimisation mechanism was the mirror descent algorithm, which allows to easily derive natural gradient updates of the variational posterior parameters by solving iterative sub-problems in the mean-space of the variational distribution. Likewise, we introduced the variational adaptive-Newton, a method that benefits from a Gaussian posterior distribution to easily express the parameters updates in the NG direction. And we also introduced the concept of *natural-momentum*, which takes advantage of the KL divergence for providing an extra memory information to the MDA. In the following chapters, we will apply the different optimisation mechanisms over Gaussian process models, for instance using VO we will derive their objective functions. Usually, training the objective function of such GP models involves problems associated to poor local optima solutions, due to strong conditionings between the parameters and hyper-parameters of the model.

Therefore, with the aim to improve the inference of said Gaussian process models and to deal with the issues of poor local optima solutions, we will make use of the optimisation mechanisms MDA and VAN for deriving model parameters and hyper-parameters updates in the direction of the natural gradient.

Chapter 3

Correlated Chained Gaussian Processes Model

A Gaussian process is a non-parametric stochastic process that extends a multivariate normal probability distribution from finite dimensional vectors to functions (Rasmussen, 2006). They have become a robust alternative for modelling parameters as latent functions in the context of probabilistic models. For instance, in a Generalised Linear Model (Murphy, 2013) a GP can be plugged to the likelihood's mean parameter through a link function so as to provide strong prior information over such parameter. Also, with the aim to improve the predictive capabilities, their use has been extended to modelling the likelihood's parameters by means of multiple GP priors. Such multi-GPs idea has been motivated by the notion that a function can be formed through the addition of multiple underlying components (Duvenaud et al., 2011; Adam et al., 2016), with the aim to find strong posterior approximations (Nguyen and Bonilla, 2014; Adam, 2017; Adam et al., 2018), and/or with the purpose of inducing higher modelling flexibilities to capture heteroscedasticity when chaining (or linking) GP priors to each parameter of the likelihood function (Saul et al., 2016). All these multi-GPs frameworks rely on constructing a joint probabilistic model where the GP latent functions are assumed independent a priori, thereby lacking of a correlation structure betweeen GPs. In general, modeling the likelihood's parameters as independent is unrealistic, because in practice one would expect the parameters to be correlated as part of a latent process that affects the model parameters. In this chapter we introduce the Correlated Chained Gaussian Processes model; this type of model introduces correlations between the likelihood's parameters based on ideas from the context of a Multi-Output GPs regression (Álvarez et al., 2012). In such a model, introducing correlations between the

outputs has shown to improve predictions of one output given the others (Álvarez and Lawrence, 2009; Osborne et al., 2008a; Boyle and Frean, 2005). In our case, we do not have multiple outputs, but multiple latent functions chained to the likelihood's parameters. Therefore, in order to introduce correlations between those latent functions, we assume by construction that each latent function is derived from an LMC, i.e, a linear combination of Q functions u_q, where each function u_q follows a GP (Journel and Huijbregts, 1979). We show how the model scales using the so-called inducing variables framework (Titsias, 2009; Álvarez et al., 2010) together with stochastic variational inference (Blei et al., 2017).

Furthermore, since the multiple latent functions chained to likelihood's parameters, make the model suffer from poor local optima solutions due to a strong conditioning between the variational posterior distribution, the multiple hyper-parameters of the GP prior and the inducing points (Van der Wilk, 2018); in this chapter we make use of the mechanisms for optimisation introduced in chapter 2, by deriving a fully natural gradient scheme that allows us to jointly fit the parameters and hyper-parameters of the CCGP model for improving its inference process. We provide comparisons between our proposed optimisation scheme and adaptive gradient methods. Also, we propose a model training strategy that consists on augmenting the CCGP model by duplicating its output and train it as a multi-output model. The strategy benefits from stochastic inference where we randomly sample mini-batches of data observations per training iteration. The intention is that at every iteration of the stochastic inference process, we can distribute mini-batches of the data thinking of those mini-batches as partially different input-output collections, as it happens in a multi-output context. This strategy present similarities to previous works that consist on distributing data for building products of GP experts (Cao and Fleet, 2014; Deisenroth and Ng, 2015). Though our motivation relies on improving the model performance by inducing correlations between mini-batches, instead of scaling up processing.

3.1 Gaussian Process

A Gaussian process is a non-parametric stochastic process that extends a multivariate normal probability distribution from finite dimensional vectors to functions (Rasmussen, 2006). Let us define a collection of N data observations with a matrix of inputs $\mathbf{X} = [\mathbf{x}_1, ..., \mathbf{x}_N]^\top \in \mathbb{R}^{N \times P}$ and a vector of outputs $\mathbf{y} = [y_1, ..., y_N]^\top$, where, for instance, each $\mathbf{x}_n$ might represent a spatio-temporal observation associated to a

measurement y_n. Usually, for a regression model, each observation y_n is modelled as a noisy version of a latent function evaluated at the n-th input observation, $f(\mathbf{x}_n)$. All data observations can be modelled by means of a likelihood function,

$$p(\mathbf{y}|\mathbf{f}, \mathbf{X}) = \prod_{n=1}^{N} p(y_n|f(\mathbf{x}_n)),$$

where the latent function follows a GP prior, i.e., $f(\mathbf{x}) \sim \mathcal{GP}(m(\mathbf{x}), k(\mathbf{x}, \mathbf{x}'))$, and consequently, $\mathbf{f} = [f(\mathbf{x}_1), ..., f(\mathbf{x}_N)]^\top$. The GP is characterised by a mean function $m(\mathbf{x}) = \mathbb{E}[f(\mathbf{x})]$ and a kernel covariance function $k(\mathbf{x}, \mathbf{x}') = \text{Cov}[f(\mathbf{x}), f(\mathbf{x}')] = \mathbb{E}[f(\mathbf{x})f(\mathbf{x}')] - \mathbb{E}[f(\mathbf{x})]\mathbb{E}[f(\mathbf{x}')]$; here $\text{Cov}[\cdot, \cdot]$ represents a covariance function. Such a kernel determines the nature of the latent functions involved in a GP model, for instance, through the kernel we can induce latent functions with: smoothness, periodicity, stationarity, non-stationarity, etc. It is important to emphasize that a kernel is a covariance function that depends on a set of hyper-parameters, that generally have to be fitted during an optimisation process when training the model. For example, a popular covariance option is the exponentiated quadratic kernel, $k(\mathbf{x}, \mathbf{x}') = \sigma_f^2 \exp\left(-\frac{\|\mathbf{x}-\mathbf{x}'\|^2}{2l^2}\right)$, which depends on the hyper-parameters σ_f^2 and l that control the amplitude and length-scale of the latent functions, respectively (Álvarez et al., 2012).

3.2 Likelihood Parametrisation Using Chained GPs

Using Gaussian processes for modelling an input-output data collection $\mathcal{D} = \{\mathbf{X}, \mathbf{y}\}$, where $\mathbf{X} = \{\mathbf{x}_n\}_{n=1}^N$ is the input data with $\mathbf{x}_n \in \mathbb{R}^P$ and $\mathbf{y} = \{y_n\}_{n=1}^N$ is the output data, consists on constructing a joint distribution between an arbitrary likelihood function and one (or multiple) Gaussian process priors (Nguyen and Bonilla, 2014; Saul et al., 2016). In such model, the likelihood's parameters are chained to the GP priors as follows,

$$p(\mathbf{y}, \mathbf{f}|\mathbf{X}) = \prod_{n=1}^{N} p(y_n|\psi_1(\mathbf{x}_n), ..., \psi_J(\mathbf{x}_n)) \prod_{j=1}^{J} \mathcal{N}(\mathbf{f}_j|\mathbf{0}, \mathbf{K}_{\mathbf{f}_j\mathbf{f}_j}), \tag{3.1}$$

where each $\mathbf{f}_j = [f_j(\mathbf{x}_1), ..., f_j(\mathbf{x}_N)]^\top$ represents a vector of latent parameter functions (LPFs) that follows a GP prior, J represents the number of such latent functions, and each $\psi_j(\mathbf{x}_n) = \phi_j(f_j(\mathbf{x}_n))$ represents a likelihood's parameter chained to the GP priors through a link function $\phi_j(\cdot)$. For instance, if the likelihood is a Heterocedastic Gaussian then its parameters mean and variance are respectively chained as $\psi_1(\mathbf{x}_n) =$

$f_1(\mathbf{x}_n)$ and $\psi_2(\mathbf{x}_n) = \exp(f_2(\mathbf{x}_n))$; or if the likelihood is a Gamma its parameters are linked as $\psi_1(\mathbf{x}_n) = \exp(f_1(\mathbf{x}_n))$ and $\psi_2(\mathbf{x}_n) = \exp(f_2(\mathbf{x}_n))$. Also, in the equation above, $\mathbf{K}_{\mathbf{f}_j\mathbf{f}_j}$ represents the kernel matrix built from evaluations between all pairs of data observations $\mathbf{X}$ in the j-th covariance function $\text{Cov}[f_j(\cdot), f_j(\cdot)] = k_j(\cdot,\cdot)$ that belongs to the j-th GP prior (Adam et al., 2018). In general, this model is known as chained Gaussian processes (CGP) model (Saul et al., 2016).

3.3 Inducing Variables Framework

The non-parametric formulation of a GP introduces computational loads through the inference process. For example, given a dataset that contains N samples, GP regression involves a computational complexity of $\mathcal{O}(N^3)$ for inverting the covariance matrix $\mathbf{K}_{\mathbf{f}_j\mathbf{f}_j}$ (Rasmussen, 2006). An approach to reduce such computational complexity is to augment the GP prior with a set of *inducing variables* $\check{\mathbf{u}}_j = [f_j(\mathbf{z}_j^{(1)}), ..., f_j(\mathbf{z}_j^{(M)})]^\top$, with $\check{u}_j := f_j$, that represent additional function evaluations at some unknown inducing points $\mathbf{Z}_j = [\mathbf{z}_j^{(1)}, ..., \mathbf{z}_j^{(M)}]^\top \in \mathbb{R}^{M\times P}$, thereby allowing a complexity reduction to $\mathcal{O}(NM^2)$, where $M << N$ (Snelson and Ghahramani, 2006; Titsias, 2009). We can write the augmented GP prior as

$$p(\mathbf{f}_j, \check{\mathbf{u}}_j) = \mathcal{N}\left(\begin{bmatrix}\mathbf{f}_j \\ \check{\mathbf{u}}_j\end{bmatrix}\middle|\mathbf{0}, \begin{bmatrix}\mathbf{K}_{\mathbf{f}_j\mathbf{f}_j} & \mathbf{K}_{\mathbf{f}_j\check{\mathbf{u}}_j} \\ \mathbf{K}_{\check{\mathbf{u}}_j\mathbf{f}_j} & \mathbf{K}_{\check{\mathbf{u}}_j\check{\mathbf{u}}_j}\end{bmatrix}\right), \tag{3.2}$$

where, by applying Gaussian properties to condition the distribution over functions $\mathbf{f}_j$ to the inducing variables $\check{\mathbf{u}}_j$, we can express such an augmented GP prior as $p(\mathbf{f}_j, \check{\mathbf{u}}_j) = p(\mathbf{f}_j|\check{\mathbf{u}}_j)p(\check{\mathbf{u}}_j)$, with:

$$p(\mathbf{f}_j|\check{\mathbf{u}}_j) = \mathcal{N}(\mathbf{f}_j|\mathbf{K}_{\mathbf{f}_j\check{\mathbf{u}}_j}\mathbf{K}^{-1}_{\check{\mathbf{u}}_j\check{\mathbf{u}}_j}\check{\mathbf{u}}_j, \mathbf{K}_{\mathbf{f}_j\mathbf{f}_j} - \mathbf{K}_{\mathbf{f}_j\check{\mathbf{u}}_j}\mathbf{K}^{-1}_{\check{\mathbf{u}}_j\check{\mathbf{u}}_j}\mathbf{K}^{\top}_{\mathbf{f}_j\check{\mathbf{u}}_j}), \tag{3.3}$$

$$p(\check{\mathbf{u}}_j) = \mathcal{N}(\check{\mathbf{u}}_j|\mathbf{0}, \mathbf{K}_{\check{\mathbf{u}}_j\check{\mathbf{u}}_j}), \tag{3.4}$$

for which $\mathbf{K}_{\mathbf{f}_j\check{\mathbf{u}}_j}$ is the cross covariance matrix formed by computing $\text{cov}\,[f_j(\cdot), \check{u}_j(\cdot)] = k_j(\cdot,\cdot)$ between inputs $\mathbf{X}$ and $\mathbf{Z}_j$; and $\mathbf{K}_{\check{\mathbf{u}}_j\check{\mathbf{u}}_j}$ is the covariance matrix built from evaluations of the covariance function $\text{cov}\,[\check{u}_j(\cdot), \check{u}_j(\cdot)] = k_j(\cdot,\cdot)$ between all pairs of inducing points $\mathbf{Z}_j$ (Hensman et al., 2015a). It is worth mentioning that the inducing variables are not always additional function evaluations of $f_j(\cdot)$, but additional evaluations of a GP function, say $u_j(\cdot)$, that covariates with the function $f_j(\cdot)$, i.e., $f_j(\cdot) \neq u_j(\cdot)$, but there exist a covariance function $\text{Cov}[f_j(\cdot), u_j(\cdot)]$ that allows us to construct the matrix

$\mathbf{K}_{\mathbf{f}_j\mathbf{u}_j}$. We will henceforth refer to $\check{u}_j(\cdot)$ as an inducing variable that strictly represents additional evaluations of $f_j(\cdot)$, and to $u_j(\cdot)$ to the alternative case. Therefore, the Gaussian properties to condition the distribution over functions $\mathbf{f}_j$ to the inducing variables $\mathbf{u}_j$ grants us the application of Eq. (3.3) and (3.4) for the alternative case of $u_j(\cdot)$ by making a variable change of $\check{u}_j(\cdot)$ to $u_j(\cdot)$. We will study this latter case in section 3.5, where the number J of latent functions $f_j(\cdot)$ is not necessarily the same number of latent functions $u_j(\cdot)$.

3.4 Evidence Lower Bound for the Multi-GP setting

For non-Gaussian likelihoods, posterior inference is analytically intractable and approximations are needed instead. To overcome this issue, an inducing variable framework combined with the VI mechanism described in section 2.3, allow us to build a tractable objective bound when using multiple GPs (Hensman et al., 2013).[1] Let $p(\mathbf{y}, \mathbf{f}, \check{\mathbf{u}}) = p(\mathbf{y}|\mathbf{f})p(\mathbf{f}, \check{\mathbf{u}})$ be a joint distribution with likelihood function $p(\mathbf{y}|\mathbf{f}) = \prod_{n=1}^{N} p(y_n|\psi_1(\mathbf{x}_n), ..., \psi_J(\mathbf{x}_n))$ and augmented GP prior

$$p(\mathbf{f}, \check{\mathbf{u}}) = \prod_{j=1}^{J} p(\mathbf{f}_j|\check{\mathbf{u}}_j)p(\check{\mathbf{u}}_j), \tag{3.5}$$

as the one already introduced in Eq. (3.3) and (3.4). Also, let $\mathbf{f} = [\mathbf{f}_1^\top, ..., \mathbf{f}_J^\top]^\top$ and $\check{\mathbf{u}} = [\check{\mathbf{u}}_1^\top, ..., \check{\mathbf{u}}_J^\top]^\top$ be vectors that group the LPFs and inducing variables, respectively. In order to approximate the true posterior we introduce a free parametrised variational distribution $q(\mathbf{f}, \check{\mathbf{u}}) \approx p(\mathbf{f}, \check{\mathbf{u}}|\mathbf{y})$. Such approximation consists on optimising the variational posterior's parameters by minimising a Kullback-Leibler (KL) divergence between the distribution $q(\mathbf{f}, \check{\mathbf{u}})$ and the true posterior $p(\mathbf{f}, \check{\mathbf{u}}|\mathbf{y})$. However, in practice we cannot compute that KL divergence, but equivalently maximise an evidence lower bound, i.e., a lower bound to the log marginal likelihood. That ELBO can be derived as follows:

$$\mathcal{L} = \mathbb{E}_{q(\mathbf{f},\check{\mathbf{u}})}\left[\log \frac{p(\mathbf{y}|\mathbf{f})p(\mathbf{f}, \check{\mathbf{u}})}{q(\mathbf{f}, \check{\mathbf{u}})}\right], \tag{3.6}$$

[1] We avoid the full notation $p(\mathbf{y}, \mathbf{f}, \check{\mathbf{u}}, \mathbf{X}, \mathbf{Z})$ excluding the variables $\mathbf{X}$ and $\mathbf{Z}$ to ease the writing.

where the variational posterior distribution is defined as

$$q(\mathbf{f}, \check{\mathbf{u}}) = \prod_{j=1}^{J} p(\mathbf{f}_j|\check{\mathbf{u}}_j)q(\check{\mathbf{u}}_j),$$

where $p(\mathbf{f}_j|\check{\mathbf{u}}_j)$ was already defined in (3.3) (Saul et al., 2016). The Chained GP model assumes that the posterior $q(\check{\mathbf{u}}) = \prod_{j=1}^{J} \mathcal{N}(\check{\mathbf{u}}_j|\mathbf{m}_j, \mathbf{V}_j)$ is the product of J parametrised Gaussian distributions, each one with mean $\mathbf{m}_j$ and covariance $\mathbf{V}_j$. When solving for Eq. (3.6) the objective becomes:

$$\mathcal{L} = \sum_{n=1}^{N} \mathbb{E}_{q(\mathbf{f}_1)\cdots q(\mathbf{f}_J)}[\log p(y_n|\psi_1(\mathbf{x}_n), ..., \psi_J(\mathbf{x}_n)] - \mathbb{D}_{KL}(q(\check{\mathbf{u}})||p(\check{\mathbf{u}})),$$

where $\mathbb{D}_{KL}(\cdot||\cdot)$ is a Kullback-Leibler divergence and each distribution $q(\mathbf{f}_j)$ can be computed from

$$q(\mathbf{f}_j) := \int p(\mathbf{f}_j|\check{\mathbf{u}}_j)q(\check{\mathbf{u}}_j)d\check{\mathbf{u}}_j, \tag{3.7}$$

where the variable $\check{\mathbf{u}}_j$ is integrated out. Once we have built the objective ELBO, the goal is to optimise it w.r.t each variational parameter $\mathbf{m}_j$ and $\mathbf{V}_j$, each set of unknown inducing points $\mathbf{Z}_j$ and the kernel hyper-parameters.

3.5 Introducing Correlations over Chained Gaussian Processes

We introduce an alternative approach that explores correlations between the GP latent functions. Such correlations are induced through the inclusion of a multi-parameter GP prior where the LPFs are considered to come from a linear model of coregionalisation (Journel and Huijbregts, 1979) as follows:

$$f_j(\mathbf{x}_n) = \sum_{q=1}^{Q} \sum_{i=1}^{R_q} w_{j,q}^{i} u_q^i(\mathbf{x}_n), \tag{3.8}$$

where $u_q^i(\mathbf{x})$ are samples from $u_q(\cdot) \sim \mathcal{GP}(0, k_q(\cdot,\cdot))$ taken IID, and each $w_{j,q}^i$ is a linear combination coefficient (LCC). We will use $R_q = 1$ for simplicity. Such a model is also known as the semi-parametric latent factor model (SLFM) (Teh et al., 2005). Each latent function $f_j(\mathbf{x}_n)$ is chained to each likelihood's parameter in Eq. (3.1), as $\psi_j(\mathbf{x}_n) = \phi_j(f_j(\mathbf{x}_n))$. Hence the likelihood function depends of J latent functions

necessary for representing its parameters, since there is a LCC per LPF we can group in a vector $\mathbf{w}_q = [w_{1,q}, ..., w_{J,q}]^\top \in \mathbb{R}^{J\times 1}$ all the coefficients per function $u_q(\cdot)$; and we can cluster all vectors $\mathbf{w}_q$ in a specific vector of LCCs $\mathbf{w} = [\mathbf{w}_1^\top, ..., \mathbf{w}_Q^\top]^\top \in \mathbb{R}^{QJ\times 1}$. Now that we know that the GP latent functions f_j are a priori correlated due to the generative model in Eq. (3.8), we can express the likelihood as:

$$p(\mathbf{y}|\mathbf{f}) = \prod_{n=1}^{N} p(y_n|\psi_1(\mathbf{x}_n), ..., \psi_J(\mathbf{x}_n)),$$

and our augmented GP prior becomes,

$$p(\mathbf{f}, \mathbf{u}) = \prod_{j=1}^{J} p(\mathbf{f}_j|\mathbf{u})p(\mathbf{u}), \tag{3.9}$$

where we can use the properties of a multivariate Gaussian distribution to write:

$$p(\mathbf{f}_j|\mathbf{u}) = \mathcal{N}(\mathbf{f}_j|\mathbf{K}_{\mathbf{f}_j\mathbf{u}}\mathbf{K}_{\mathbf{u}\mathbf{u}}^{-1}\mathbf{u}, \mathbf{K}_{\mathbf{f}_j\mathbf{f}_j} - \mathbf{K}_{\mathbf{f}_j\mathbf{u}}\mathbf{K}_{\mathbf{u}\mathbf{u}}^{-1}\mathbf{K}_{\mathbf{f}_j\mathbf{u}}^\top), \tag{3.10}$$

$$p(\mathbf{u}) = \mathcal{N}(\mathbf{u}|\mathbf{0}, \mathbf{K}_{\mathbf{u}\mathbf{u}}), \tag{3.11}$$

where $\mathbf{K}_{\mathbf{u}\mathbf{u}} \in \mathbb{R}^{QM\times QM}$ is a block-diagonal matrix with blocks $\mathbf{K}_{\mathbf{u}_q\mathbf{u}_q} \in \mathbb{R}^{M\times M}$ built from evaluations of $\mathrm{Cov}\left[u_q(\cdot), u_q(\cdot)\right] = k_q(\cdot,\cdot)$ between all pairs of inducing points $\mathbf{Z}_q = [\mathbf{z}_q^{(1)}, ..., \mathbf{z}_q^{(M)}]^\top \in \mathbb{R}^{M\times P}$; the covariance matrix $\mathbf{K}_{\mathbf{f}_j\mathbf{f}_j}$ is built from evaluations of $\mathrm{Cov}\left[f_j(\cdot), f_j(\cdot)\right] = \sum_{q=1}^{Q} w_{j,q}w_{j,q}k_q(\cdot,\cdot)$ over all pairs of data $\mathbf{X}$, thus involving all Q kernel-covariances weighted by the coefficients $w_{j,q}$. It is important to highlight that in the former method each covariance matrix $\mathbf{K}_{\mathbf{f}_j\mathbf{f}_j}$ in Eq. (3.1) and (3.2) is influenced by an unique kernel-covariance $\mathrm{Cov}[f_j(\cdot), f_j(\cdot)] = k_j(\cdot,\cdot)$; and $\mathbf{K}_{\mathbf{f}_j\mathbf{u}} = [\mathbf{K}_{\mathbf{f}_j\mathbf{u}_1}, ..., \mathbf{K}_{\mathbf{f}_j\mathbf{u}_Q}]$ is a covariance matrix with blocks $\mathbf{K}_{\mathbf{f}_j\mathbf{u}_q} \in \mathbb{R}^{N\times M}$ constructed by computing $\mathrm{Cov}\left[f_j(\cdot), u_q(\cdot)\right] = w_{j,q}k_q(\cdot,\cdot)$ between $\mathbf{X}$ and $\mathbf{Z}_q$. Unlike the matrix $\mathbf{K}_{\mathbf{f}_j\check{\mathbf{u}}_j}$ in Eq. (3.2) and (3.3), here the blocks $\mathbf{K}_{\mathbf{f}_j\mathbf{u}_q}$ are additionally weighted by the coefficients $w_{j,q}$. Notice that $\mathbf{u} = [\mathbf{u}_1^\top, ..., \mathbf{u}_Q^\top]^\top \in \mathbb{R}^{QM\times 1}$ is the inducing variables vector formed by the function evaluations $\mathbf{u}_q = [u_q(\mathbf{z}_q^{(1)}), ..., u_q(\mathbf{z}_q^{(M)})]^\top$ (Álvarez et al., 2010). It is worth to distinguish from subsection 3.3 that in our model the inducing variables $\mathbf{u}$ are not additional function evaluations of $\mathbf{f}$, but evaluations of each function $u_q(\cdot)$, i.e., $\mathbf{u} \neq \check{\mathbf{u}}$. Also, Eq. (3.9) differs from Eq. (3.5) in the fact that we condition the distribution over each latent function f_j on all the Q latent functions u_q. While due to the independence assumption in former methods the distribution over each latent function f_j can only be conditioned on a unique $\check{u}_j$. We refer to the model described above as the Correlated Chained Gaussian Processes model.

Given that most of the likelihood functions that we can define for our CCGP model make it intractable, it is necessary to derive an ELBO. To this end we define a variational posterior as follows,

$$q(\mathbf{f}, \mathbf{u}) = \prod_{j=1}^{J} p(\mathbf{f}_j|\mathbf{u}) \prod_{q=1}^{Q} q(\mathbf{u}_q), \tag{3.12}$$

where each $q(\mathbf{u}_q) = \mathcal{N}(\mathbf{u}_q|\mathbf{m}_q, \mathbf{V}_q)$ is a Gaussian distribution with mean $\mathbf{m}_q$ and covariance $\mathbf{V}_q$. Following Eq. (3.6) we can derive the ELBO:

$$\mathcal{L} = \sum_{n=1}^{N} \mathbb{E}_{q(\mathbf{f}_1)\cdots q(\mathbf{f}_J)} \left[\log p\left(y_n|\psi_1(\mathbf{x}_n), ..., \psi_J(\mathbf{x}_n)\right)\right] - \mathbb{D}_{KL}(q(\mathbf{u})||p(\mathbf{u})), \tag{3.13}$$

where the posteriors over each LPF can be computed from:

$$q(\mathbf{f}_j) := \int p(\mathbf{f}_j|\mathbf{u})q(\mathbf{u})d\mathbf{u}. \tag{3.14}$$

In the equation above, all posteriors $q(\mathbf{u}) = \prod_{q=1}^{Q} q(\mathbf{u}_q)$ influence the construction of each LPF's posterior $q(\mathbf{f}_j)$, while that does not happen in previous methods, as it can be seen in Eq. (3.7). We solve for Eq. (3.14) arriving to:

$$q(\mathbf{f}_j) = \mathcal{N}(\mathbf{f}_j|\mathbf{K}_{\mathbf{f}_j\mathbf{u}}\mathbf{K}_{\mathbf{u}\mathbf{u}}^{-1}\mathbf{m}, \mathbf{K}_{\mathbf{f}_j\mathbf{f}_j} + \mathbf{K}_{\mathbf{f}_j\mathbf{u}}\mathbf{K}_{\mathbf{u}\mathbf{u}}^{-1}(\mathbf{V} - \mathbf{K}_{\mathbf{u}\mathbf{u}})\mathbf{K}_{\mathbf{u}\mathbf{u}}^{-1}\mathbf{K}_{\mathbf{f}_j\mathbf{u}}^{\top}), \tag{3.15}$$

where $\mathbf{m} = [\mathbf{m}_1^{\top}, ..., \mathbf{m}_Q^{\top}]^{\top}$ is a vector of means and $\mathbf{V}$ a block-diagonal matrix with blocks given by each covariance $\mathbf{V}_q$. The ELBO in Eq. (3.13) can be easily rewriten in terms of mini-batches allowing stochastic inference as follows:

$$\hat{\mathcal{L}} = S \sum_{i_c \in C} \mathbb{E}_{q(\mathbf{f}_1)\cdots q(\mathbf{f}_J)} \left[\log p\left(y_{i_c}|\psi_1(\mathbf{x}_{i_c}), ..., \psi_J(\mathbf{x}_{i_c})\right)\right] - \mathbb{D}_{KL}(q(\mathbf{u})||p(\mathbf{u})), \tag{3.16}$$

where the variable C represents a set of indexes, where each index is uniformly sample as $i_c \sim \text{Unif}(1, ..., N)$, and $S = N/L$ is a scaling factor with L as the set's size.

In order to make predictions with the model, it is necessary to compute the following distribution:

$$p(\mathbf{y}_*|\mathbf{y}) \approx \int p(\mathbf{y}_*|\mathbf{f}_*)q(\mathbf{f}_*)d\mathbf{f}_*, \tag{3.17}$$

where $q(\mathbf{f}_*) = \prod_{j=1}^{J} q(\mathbf{f}_{j,*})$, and each $q(\mathbf{f}_{j,*})$ can be computed with Eq. (3.15) by evaluating $\mathbf{K}_{\mathbf{f}_{j,*}\mathbf{u}}$ and $\mathbf{K}_{\mathbf{f}_{j,*}\mathbf{f}_{j,*}}$ at the new inputs $\mathbf{X}_*$.

3.6 Deriving a Fully Natural Gradient Scheme for the CCGP model

This section describes how to derive the fully natural gradient updates for optimising the CCGP model. We first detail how to induce an exploratory distribution over the hyper-parameters and inducing points as per the section 2.2, then we write down the MDA for the model and derive the update equations. Later on, we get into specific details about the algorithm's implementation.

3.6.1 An Exploratory Distribution for the CCGP

In the context of sparse GPs, the kernel hyper-parameters and inducing points of the model have usually been treated as deterministic variables. Here, we use the VO perspective as a mechanism to induce randomness over such variables, this with the aim to gain exploration for finding better solutions during the inference process (Staines and Barber, 2013). To this end we define and connect random real vectors to the variables through a link function as follows: for the inducing points, $\mathbf{z} = \boldsymbol{\theta}_{\mathbf{Z}}$, for a vector that stacks all kernel hyper-parameters of the model, $\mathbf{l}_{\text{kern}} = \exp(\boldsymbol{\theta}_{\mathbf{L}})$, with $\boldsymbol{\theta}_{\mathbf{L}} = [\boldsymbol{\theta}_{\mathbf{L}_1}^\top, ..., \boldsymbol{\theta}_{\mathbf{L}_Q}^\top]^\top \in \mathbb{R}^{QP\times 1}$, and for the vector of LCC $\mathbf{w} = \boldsymbol{\theta}_{\mathbf{w}}$, that are used to generate the LPFs in Eq. (3.8). We have defined the real random vectors $\boldsymbol{\theta}_{\mathbf{Z}} \in \mathbb{R}^{QMP\times 1}$, $\boldsymbol{\theta}_{\mathbf{L}_q} \in \mathbb{R}^{P\times 1}$ and $\boldsymbol{\theta}_{\mathbf{w}} \in \mathbb{R}^{QJ\times 1}$ to link the set of inducing points, the kernel hyper-parameters per latent function $u_q(\cdot)$, and the vector $\mathbf{w}$ of LCCs. We cluster the random vectors defining $\boldsymbol{\theta} = [\boldsymbol{\theta}_{\mathbf{Z}}^\top, \boldsymbol{\theta}_{\mathbf{L}}^\top, \boldsymbol{\theta}_{\mathbf{w}}^\top]^\top \in \mathbb{R}^{(QMP+QP+QJ)\times 1}$ to refer to all the parameters in a single variable. Hence, we can specify an exploratory distribution $q(\boldsymbol{\theta}) := \mathcal{N}(\boldsymbol{\theta}|\boldsymbol{\mu}, \boldsymbol{\Sigma})$ for applying the VO approach in Eq. (2.2), though for our case the objective to bound is already derived in Eq. (3.16) for the CCGP model. Therefore our VO bound is defined as follows:

$$\tilde{\mathcal{F}} = \mathbb{E}_{q(\boldsymbol{\theta})}\big[-\hat{\mathcal{L}}\big] + \mathbb{D}_{KL}(q(\boldsymbol{\theta})||p(\boldsymbol{\theta})), \tag{3.18}$$

where $p(\boldsymbol{\theta}) = \mathcal{N}(\boldsymbol{\theta}|\mathbf{0}, \lambda_1^{-1}\mathbf{I})$ is a Gaussian distribution with precision λ_1 that forces further exploration of $\boldsymbol{\theta}$'s space (Khan et al., 2017b). It is important to highlight that the kernel hyper-parameters are guaranteed to be strictly positive by means of the exponential link function, $\mathbf{l}_{\text{kern}} = \exp(\boldsymbol{\theta}_{\mathbf{L}})$; i.e., from a Bayesian perspective the kernel hyper-parameters, $\mathbf{l}_{\text{kern}}$, are set to follow a prior Log-Normal distribution. Since our aim, from an optimisation perspective, is to induce additional exploration over such hyper-parameters of the model by using the VAN approach described in section 2.4.2.

Thereby a Log-Normal prior is just a consequence of the implementation in which our work relies by using a Gaussian exploratory distribution, $q(\boldsymbol{\theta}) := \mathcal{N}(\boldsymbol{\theta}|\boldsymbol{\mu}, \boldsymbol{\Sigma})$, and the exponential link functions to the positive hyper-parameters.

3.6.2 Mirror Descent Algorithm for the CCGP

With the purpose of minimising our VO objective in Eq. (3.18), we use the MDA described in Eq. (2.5), which additionally exploits the natural-momentum. Therefore, we make use of the mean-parameters of distributions $q(\mathbf{u}_q)$ and $q(\boldsymbol{\theta})$ by defining $\boldsymbol{\rho}_q = \{\mathbf{m}_q, \mathbf{m}_q\mathbf{m}_q^\top + \mathbf{V}_q\}$ and $\boldsymbol{\eta} = \{\boldsymbol{\mu}, \boldsymbol{\mu}\boldsymbol{\mu}^\top + \boldsymbol{\Sigma}\}$ (Khan and Lin, 2017). In this way we can write the MDA as:

$$\begin{aligned}\boldsymbol{\eta}_{t+1}, \{\boldsymbol{\rho}_{q,t+1}\}_{q=1}^{Q} = &\underset{\boldsymbol{\eta},\, \{\boldsymbol{\rho}_q\}_{q=1}^{Q}}{\arg\min} \langle \boldsymbol{\eta}, \hat{\nabla}_{\boldsymbol{\eta}} \tilde{\mathcal{F}}_t \rangle + \frac{1}{\tilde{\alpha}_t}\mathrm{KL}(\boldsymbol{\theta})_t - \frac{\tilde{\gamma}_t}{\tilde{\alpha}_t}\mathrm{KL}(\boldsymbol{\theta})_{t-1} \\ &+ \sum_{q=1}^{Q}\Big[\langle \boldsymbol{\rho}_q, \hat{\nabla}_{\boldsymbol{\rho}_q} \tilde{\mathcal{F}}_t \rangle + \frac{1}{\tilde{\beta}_t}\mathrm{KL}(\mathbf{u}_q)_t - \frac{\tilde{\upsilon}_t}{\tilde{\beta}_t}\mathrm{KL}(\mathbf{u}_q)_{t-1}\Big],\end{aligned} \tag{3.19}$$

where $\tilde{\mathcal{F}}_t := \tilde{\mathcal{F}}(\{\mathbf{m}_{q,t}\}_{q=1}^{Q}, \{\mathbf{V}_{q,t}\}_{q=1}^{Q}, \boldsymbol{\mu}_t, \boldsymbol{\Sigma}_t)$ and $\tilde{\beta}_t$, $\tilde{\alpha}_t$, $\tilde{\upsilon}_t$, and $\tilde{\gamma}_t$ are positive step-size parameters. Notice that the sub-index t in the equations above refers to a t-th instant of an iterative procedure.

3.6.3 Fully Natural Gradient Updates

We can solve for Eq. (3.19) by computing derivatives w.r.t $\boldsymbol{\eta}$ and $\boldsymbol{\rho}$, and setting to zero (Khan et al., 2017a). This way we obtain results similar to Eq. (2.6) and (2.7), we call them FNG updates:

$$\boldsymbol{\Sigma}_{t+1}^{-1} = \boldsymbol{\Sigma}_t^{-1} + 2\alpha_t \hat{\nabla}_{\boldsymbol{\Sigma}} \tilde{\mathcal{F}}_t \tag{3.20}$$

$$\boldsymbol{\mu}_{t+1} = \boldsymbol{\mu}_t - \alpha_t \boldsymbol{\Sigma}_{t+1} \hat{\nabla}_{\boldsymbol{\mu}} \tilde{\mathcal{F}}_t + \gamma_t \boldsymbol{\Sigma}_{t+1} \boldsymbol{\Sigma}_t^{-1} (\boldsymbol{\mu}_t - \boldsymbol{\mu}_{t-1}) \tag{3.21}$$

$$\mathbf{V}_{q,t+1}^{-1} = \mathbf{V}_{q,t}^{-1} + 2\beta_t \hat{\nabla}_{\mathbf{V}_q} \tilde{\mathcal{F}}_t \tag{3.22}$$

$$\begin{aligned}\mathbf{m}_{q,t+1} = &\mathbf{m}_{q,t} - \beta_t \mathbf{V}_{q,t+1} \hat{\nabla}_{\mathbf{m}_q} \tilde{\mathcal{F}}_t \\ &+ \upsilon_t \mathbf{V}_{q,t+1} \mathbf{V}_{q,t}^{-1} (\mathbf{m}_{q,t} - \mathbf{m}_{q,t-1}),\end{aligned} \tag{3.23}$$

where $\alpha_t = \tilde{\alpha}_t/(1-\tilde{\gamma}_t)$, $\beta_t = \tilde{\beta}_t/(1-\tilde{\upsilon}_t)$, $\gamma_t = \tilde{\gamma}_t/(1-\tilde{\gamma}_t)$ and $\upsilon_t = \tilde{\upsilon}_t/(1-\tilde{\upsilon}_t)$ are positive step-size parameters (see Appendix F for details on the gradients derivation).

3.7 Implementation

In order to implement the proposed method, we have to take into account that our computational complexity depends on inverting the covariance matrix $\mathbf{\Sigma}$ in Eq. (3.20). Such complexity can be expressed as $\mathcal{O}((QMP + QP + QJ)^3)$ for the CCGP with LMC, where the terms with the number of inducing points and/or input dimensionality tend to dominate the complexity. Likewise, the gradient $\hat{\nabla}_{\mathbf{\Sigma}}\tilde{\mathcal{F}}$ involves computing the Hessian $\hat{\nabla}^2_{\boldsymbol{\theta}\boldsymbol{\theta}}\tilde{\mathcal{L}}$ which can be computationally expensive and prone to suffer from non-positive definiteness. To alleviate those complexity issues we assume $\mathbf{\Sigma} = \text{diag}(\boldsymbol{\sigma}^2)$, where $\boldsymbol{\sigma}$ is a vector of standard deviations, and $\text{diag}(\boldsymbol{\sigma}^2)$ represents a matrix with the elements of $\boldsymbol{\sigma}^2$ on its diagonal. Additionally, we estimate the Hessian by means of the Gauss-Newton (GN) approximation $\hat{\nabla}^2_{\boldsymbol{\theta}\boldsymbol{\theta}}\tilde{\mathcal{L}} \approx \hat{\nabla}_{\boldsymbol{\theta}}\tilde{\mathcal{L}} \circ \hat{\nabla}_{\boldsymbol{\theta}}\tilde{\mathcal{L}}$ (Bertsekas, 1999; Khan et al., 2017b). The authors in Khan et al. (2018) term this method as the variational RMSprop with momentum. They alternatively express Eq. (3.20) and (3.21) as:

$$\mathbf{p}_{t+1} = (1 - \alpha_t)\,\mathbf{p}_t + \alpha_t \mathbb{E}_{q(\boldsymbol{\theta})}\left[\hat{\nabla}_{\boldsymbol{\theta}}\tilde{\mathcal{L}} \circ \hat{\nabla}_{\boldsymbol{\theta}}\tilde{\mathcal{L}}\right] \tag{3.24}$$

$$\begin{aligned}\boldsymbol{\mu}_{t+1} = \boldsymbol{\mu}_t &- \alpha_t(\mathbf{p}_{t+1} + \lambda_1\mathbf{1})^{-1} \circ \hat{\nabla}_{\boldsymbol{\mu}}\tilde{\mathcal{F}} \\ &+ \gamma_t(\mathbf{p}_t + \lambda_1\mathbf{1}) \circ (\mathbf{p}_{t+1} + \lambda_1\mathbf{1})^{-1} \circ \left(\boldsymbol{\mu}_t - \boldsymbol{\mu}_{t-1}\right),\end{aligned} \tag{3.25}$$

where $\hat{\nabla}_{\boldsymbol{\mu}}\tilde{\mathcal{F}} = (\mathbb{E}_{q(\boldsymbol{\theta})}\left[\hat{\nabla}_{\boldsymbol{\theta}}\tilde{\mathcal{L}}\right] + \lambda_1\boldsymbol{\mu}_t)$, $\circ$ represents an element-wise product and we have made a variable change defining a vector $\mathbf{p}_t := \boldsymbol{\sigma}_t^{-2} - \lambda_1\mathbf{1}$, with $\mathbf{1}$ as a vector of ones. The GN approximation provides stronger numerical stability by preventing that $\boldsymbol{\sigma}^2$ becomes negative. Also, using $\text{diag}(\boldsymbol{\sigma}^2)$ we reduce the computational complexity from $\mathcal{O}((QMP + QP + QJ)^3)$ to $\mathcal{O}(QMP + QP + QJ)$, see Appendix G for a pseudo-code implementation of the algorithm.

3.8 Predictive Distribution

Making predictions with the CCGP model depends on computing the distribution: $p(\mathbf{y}_*|\mathbf{y}) \approx \int p(\mathbf{y}_*|\mathbf{f}_*)q(\mathbf{f}_*)d\mathbf{f}_*$, where $q(\mathbf{f}_*) = \prod_{j=1}^{J} q(\mathbf{f}_{j,*})$. Each distribution, $q(\mathbf{f}_{j,*})$, can be calculated from $q(\mathbf{f}_{j,*}) = \int p(\mathbf{f}_{j,*}|\mathbf{u}, \boldsymbol{\theta})q(\mathbf{u})q(\boldsymbol{\theta})d\boldsymbol{\theta}d\mathbf{u}$, where $q(\boldsymbol{\theta})$ is the exploratory (or variational) distribution induced over all hyper-parameters and inducing points of the model. Nonetheless, the fact of integrating out the distribution, $q(\boldsymbol{\theta})$, involves a high computational complexity during the predicting process. For instance, in the work by Rossi et al. (2021) this scenario emerges when studying the fact of inducing prior distributions over the hyper-parameters and inducing points of the model, i.e., building a

fully Bayesian GP model; consequently, the authors propose to perform the prediction operation by parallelizing and using GPUs for integrating out the posterior distribution over said hyper-parameters and inducing points. Nonetheless, our motivation for treating the hyper-parameters and inducing points as stochastic variables relies on the fact of improving exploration during optimisation. Indeed, in the practice we realised that $q(\boldsymbol{\theta})$'s covariance converged to very small values, in general $\text{diag}(\boldsymbol{\sigma}^2) \leq 10^{-15}$, and almost all the uncertainty information was concentrated on $q(\mathbf{u})$'s covariance. Therefore, with the aim to avoid the expensive computations involved in the integrations, we can trade-off the computation by using the MAP solution for $q(\boldsymbol{\theta})$ (see Appendix H for details on MAP in the context of VI) and completely integrate over the remaining distribution as follows: $q(\mathbf{f}_{j,*}) = \int p(\mathbf{f}_{j,*}|\mathbf{u}, \boldsymbol{\theta} = \boldsymbol{\mu})q(\mathbf{u})d\mathbf{u}$. When solving for these integrals we arrive to the same solutions in Eq. (3.15), where we just need to evaluate, at the new inputs $\mathbf{X}_*$, the matrix covariances $\mathbf{K}_{\mathbf{f}_{j,*}\mathbf{u}}$ and $\mathbf{K}_{\mathbf{f}_{j,*}\mathbf{f}_{j,*}}$.

3.9 Training Strategy: Augmenting the CCGP's Output

The objective in Eq. (3.16) shows a version of the ELBO that permits stochastic inference (Blei et al., 2017) by randomly sampling a mini-batch of data observations per training iteration. Lets us supose that we pick two mini-batches of data, if we compare them, it is reasonable to think that one mini-batch can contain information that it is not necessarily present in the other mini-batch. We can think of those two mini-batches as partially different collections of input-output data, as it happens in a multi-output context. When we look at multi-output Gaussian processes performance, we realise that their main achievement relates to providing more accurate predictions due to the correlations induced between the outputs (Álvarez and Lawrence, 2009; Osborne et al., 2008a). Therefore, with the intention that at every iteration of the inference process we can induce correlations between two mini-batches, as if each one were associated to a different output; we propose a training strategy that is based on augmenting a single-output GP model in order to treat it as a multi-output model.

Let us call our original output $\mathbf{y}_1$ and create a duplicate version of it as $\mathbf{y}_2 = \mathbf{y}_1$, thereby we can express a two-output likelihood as follows:

$$p(\mathbf{y}|\mathbf{f}) = \prod_{n=1}^{N} p(y_{1,n}|\psi_{1,1}(\mathbf{x}_n), ..., \psi_{1,J}(\mathbf{x}_n))p(y_{2,n}|\psi_{2,1}(\mathbf{x}_n), ..., \psi_{2,J}(\mathbf{x}_n)),$$

where we additionally index the LPFs indicating the association to an output, so we can express the SLFM as:

$$f_{d,j}(\mathbf{x}) = \sum_{q=1}^{Q} w_{d,j,q} u_q(\mathbf{x}_n),$$

where the linear combination coefficients $w_{d,j,q}$ include an additional index that associates them to a d-th output (Álvarez et al., 2012). With the aim to derive the objective ELBO, we define the augmented GP prior and approximate posterior respectively as,

$$p(\mathbf{f}, \mathbf{u}) = \prod_{j=1}^{J} p(\mathbf{f}_{1,j}|\mathbf{u}) p(\mathbf{f}_{2,j}|\mathbf{u}) p(\mathbf{u}),$$
$$q(\mathbf{f}, \mathbf{u}) = \prod_{j=1}^{J} p(\mathbf{f}_{1,j}|\mathbf{u}) p(\mathbf{f}_{2,j}|\mathbf{u}) \prod_{q=1}^{Q} q(\mathbf{u}_q),$$

where the distributions $p(\mathbf{u})$ and each $q(\mathbf{u}_q)$ are exactly the same ones already defined in Eq. (3.11) and (3.12) (Moreno-Muñoz et al., 2018). The distributions $p(\mathbf{f}_{d,j}|\mathbf{u})$ can simply be identified in Eq. (3.10) by making a change of variable from $\mathbf{f}_j$ to $\mathbf{f}_{d,j}$. We derive the ELBO following Eq. (3.6) to arrive to:

$$\hat{\mathcal{L}} = S_A \sum_{i_a \in A} \mathbb{E}_{q(\mathbf{f}_{1,1}),\ldots,q(\mathbf{f}_{1,J})} \left[\log p\left(y_{1,i_a}|\psi_{1,1}(\mathbf{x}_{i_a}), ..., \psi_{1,J}(\mathbf{x}_{i_a})\right)\right] \tag{3.26}$$
$$+ S_B \sum_{i_b \in B} \mathbb{E}_{q(\mathbf{f}_{2,1}),\ldots,q(\mathbf{f}_{2,J})} \left[\log p\left(y_{2,i_b}|\psi_{2,1}(\mathbf{x}_{i_b}), ..., \psi_{2,J}(\mathbf{x}_{i_b})\right)\right] - \mathbb{D}_{KL}(q(\mathbf{u})||p(\mathbf{u})),$$

where we express this ELBO in terms of two mini-batches one for $\mathbf{y}_1$ and the other for $\mathbf{y}_2$. Here the variables A and B represent sets of indexes, where each index is uniformly sample as $i_a \sim \text{Unif}(1, ..., N)$ and $i_b \sim \text{Unif}(1, ..., N)$. The summatories are scaled by the factors $S_A = N/L_A$ and $S_B = N/L_B$ with L_A and L_B as the size of A and B, respectively, i.e., the mini-batch size per output. Also, each distribution $q(\mathbf{f}_{d,j})$ can be computed from Eq. (3.15) by making a change of variable from $\mathbf{f}_j$ to $\mathbf{f}_{d,j}$. Notice that despite modelling two outputs using the augmented strategy, there is not need to make predictions for both of them since $\mathbf{y}_2$ is the duplicate of $\mathbf{y}_1$. Indeed, the LPFs' distributions are conditionally independent, so it is not necessary to make predictions using all marginal posteriors $q(\mathbf{f}_{d,j,*})$. We can simply pick the posteriors associated to one output, for instance $q(\mathbf{f}_{1,*}) = \prod_{j=1}^{J} q(\mathbf{f}_{1,j,*})$, and make predictions following Eq. (3.17).

One might think that Eq. (3.26) is simply the same as rewriting the single-output stochastic ELBO in Eq. (3.16) in terms of two summatories as follows:

$$\hat{\mathcal{L}} = S \sum_{i_a \in A} \mathbb{E}_{q(\mathbf{f}_1),...,q(\mathbf{f}_J)} \left[\log p\left(y_{i_a} | \psi_1(\mathbf{x}_{i_a}), ..., \psi_J(\mathbf{x}_{i_a})\right)\right] \tag{3.27}$$
$$+ S \sum_{i_b \in B} \mathbb{E}_{q(\mathbf{f}_1),...,q(\mathbf{f}_J)} \left[\log p\left(y_{i_b} | \psi_1(\mathbf{x}_{i_b}), ..., \psi_J(\mathbf{x}_{i_b})\right)\right] - \mathbb{D}_{KL},$$

where, unlike Eq. (3.26), here A and B are subsets of the original set of indexes C, i.e., $C = A \cup B$. Nonetheless, the main difference relies on the fact that, in the equation above, the scaling factor becomes $S = N/(L_A + L_B)$, where L_A represents the size of A and L_B the size of B, being different from the scaling factors S_A and S_B in (3.26). Furthermore, in the augmented setting, there is an extra set of coefficients $w_{d,j,q}$ that influence the gradients when updating the parameters of each variational posterior $q(\mathbf{u}_q)$,

$$\nabla_{\mathbf{m}_q} \hat{\mathcal{L}} = \sum_{j=1}^{J} \mathbf{R}_{\mathbf{f}_{1,j}\mathbf{u}_q}^{\top} \boldsymbol{\nu}_{1,j} + \sum_{j=1}^{J} \mathbf{R}_{\mathbf{f}_{2,j}\mathbf{u}_q}^{\top} \boldsymbol{\nu}_{2,j} - \nabla_{\mathbf{m}_q} \mathbb{D}_{KL},$$
$$\nabla_{\mathbf{V}_q} \hat{\mathcal{L}} = \sum_{j=1}^{J} \mathbf{R}_{\mathbf{f}_{1,j}\mathbf{u}_q}^{\top} \boldsymbol{\gamma}_{1,j} \mathbf{R}_{\mathbf{f}_{1,j}\mathbf{u}_q} + \sum_{j=1}^{J} \mathbf{R}_{\mathbf{f}_{2,j}\mathbf{u}_q}^{\top} \boldsymbol{\gamma}_{2,j} \mathbf{R}_{\mathbf{f}_{2,j}\mathbf{u}_q} - \nabla_{\mathbf{V}_q} \mathbb{D}_{KL},$$

where $\mathbf{R}_{\mathbf{f}_{d,j}\mathbf{u}_q} = \mathbf{K}_{\mathbf{f}_{d,j}\mathbf{u}_q} \mathbf{K}_{\mathbf{u}_q\mathbf{u}_q}^{-1}$ is a matrix that involves the computation of the covariance functions $\operatorname{cov}\left[f_{d,j}(\cdot), u_q(\cdot)\right] = w_{d,j,q} k_q(\cdot,\cdot)$, where such coefficients $w_{d,j,q}$ weight each kernel. Also, they directly affect $\boldsymbol{\nu}_{d,j} \in \mathbb{R}^{N\times 1}$ which is a vector with entries,

$$\mathbb{E}_{q(\mathbf{f}_{d,1})...q(\mathbf{f}_{d,J})}[\nabla_{f_{d,j}(\mathbf{x}_n)} \log p_{d,n}];$$

and $\boldsymbol{\gamma}_{d,j} \in \mathbb{R}^{N\times N}$ is a diagonal matrix with entries $\frac{1}{2}\mathbb{E}_{q(\mathbf{f}_{d,1})...q(\mathbf{f}_{d,J})}[\nabla^2_{f_{d,j}(\mathbf{x}_n) f_{d,j}(\mathbf{x}_n)} \log p_{d,n}]$ in its diagonal, and we have defined $\log p_{d,n} := \log p\left(y_{d,n} | f_{d,1}(\mathbf{x}_n), ..., f_{d,J}(\mathbf{x}_n)\right)$; it is worth noticing that the distributions $q(\mathbf{f}_{d,j})$ involve the computations of the coefficients $w_{d,j,q}$ as per Eq. (3.15).

3.10 Experiments

In this section we want to evaluate the performance of the CCGP model and the proposed training strategy of augmenting the output using our proposed fully natural gradient scheme of optimisation. First, in order to asses the performance of our FNG scheme, we compare it with the stochastic gradient descent (SGD) method, and the

adaptive gradient methods Ada-Delta (ADAD) and Adam; we report the convergence of the Negative ELBO (NELBO) from Eq. (3.16). Regarding the model performance, we use the FNG scheme for training purposes and report the negative log predictive density (NLPD) (Quiñonero-Candela et al., 2006) over a test set. We use the NLPD metric since it takes into account the uncertainty quantification of the model. We compare our proposed methods against the CGP model, which generalises the concept of likelihood parametrisation using multiple independent GPs.

3.10.1 Comparison between FNG and AGMs

We ran experiments over the following datasets: **boston** ($N = 505, P = 13$) contains housing information collected by the U.S. Census Service in the area of Boston, Massachusetts, consists of one output with the value of owner-occupied homes, we re-scaled it as $y \in [0, 1]$; **yacht** ($N = 308, P = 6$) is a register of the residuary resistance of sailing yachts at an initial design stage, the data is useful for estimating the required propulsive power, consists of one output, the residuary resistance; and **concrete** ($N = 1K, P = 8$) contains information of the concrete compressive strength as a function of age and ingredients, consists of one output: the concrete compressive strength, we re-scaled it as $y \in \mathbb{R}^+$.[2] For the boston dataset, we used a Beta likelihood, $\prod_{n=1}^{N} \text{Beta}(y_n|a_n, b_n)$; for the yacht dataset, a Heteroscedastic-Gaussian likelihood, $\prod_{n=1}^{N} \mathcal{N}(y_n|\mu_n, v_n)$; and for the concrete dataset, we used a likelihood Gamma, $\prod_{n=1}^{N} \text{Gamma}(y_n|\alpha_n, \beta_n)$. The LPFs were chained to the likelihood Heteroscedastic-Gaussian (HG) as $\mu_n = \psi_1(\mathbf{x}_n) = f_1(\mathbf{x}_n)$ and $v_n = \psi_2(\mathbf{x}_n) = \exp(f_2(\mathbf{x}_n))$; to Beta (Bt) as $a_n = \psi_1(\mathbf{x}_n) = \exp(f_1(\mathbf{x}_n))$ and $b_n = \psi_2(\mathbf{x}_n) = \exp(f_2(\mathbf{x}_n))$; and to Gamma (Ga) as $\alpha_n = \psi_1(\mathbf{x}_n) = \exp(f_1(\mathbf{x}_n))$ and $\beta_n = \psi_2(\mathbf{x}_n) = \exp(f_2(\mathbf{x}_n))$. With the aim to assess the performance of our FNG scheme in comparison to the AGMs, we randomly set 15 different initialisations of the CCGP model and ran the optimisation algorithms per initialisation. We assumed two inducing latent functions u_1 and u_2, during training used a mini-batch size of 100, and a number of $M = 20$. Figure 3.1 shows the average NELBO reached by each one of the optimisation methods. We can notice from Figure 3.1 that the FNG scheme generally converged to a lower NELBO in comparison to the other methods, except for the case of concrete dataset, where Adam achieved a slightly better solution. Adam method usually converged to better solutions than ADAD and SGD. The optimisers ADAD and SGD presented a poor performance for

[2] See http://archive.ics.uci.edu/ml/datasets.php for datasets information.
See https://www.cs.toronto.edu/~delve/data/boston/bostonDetail.html for boston.

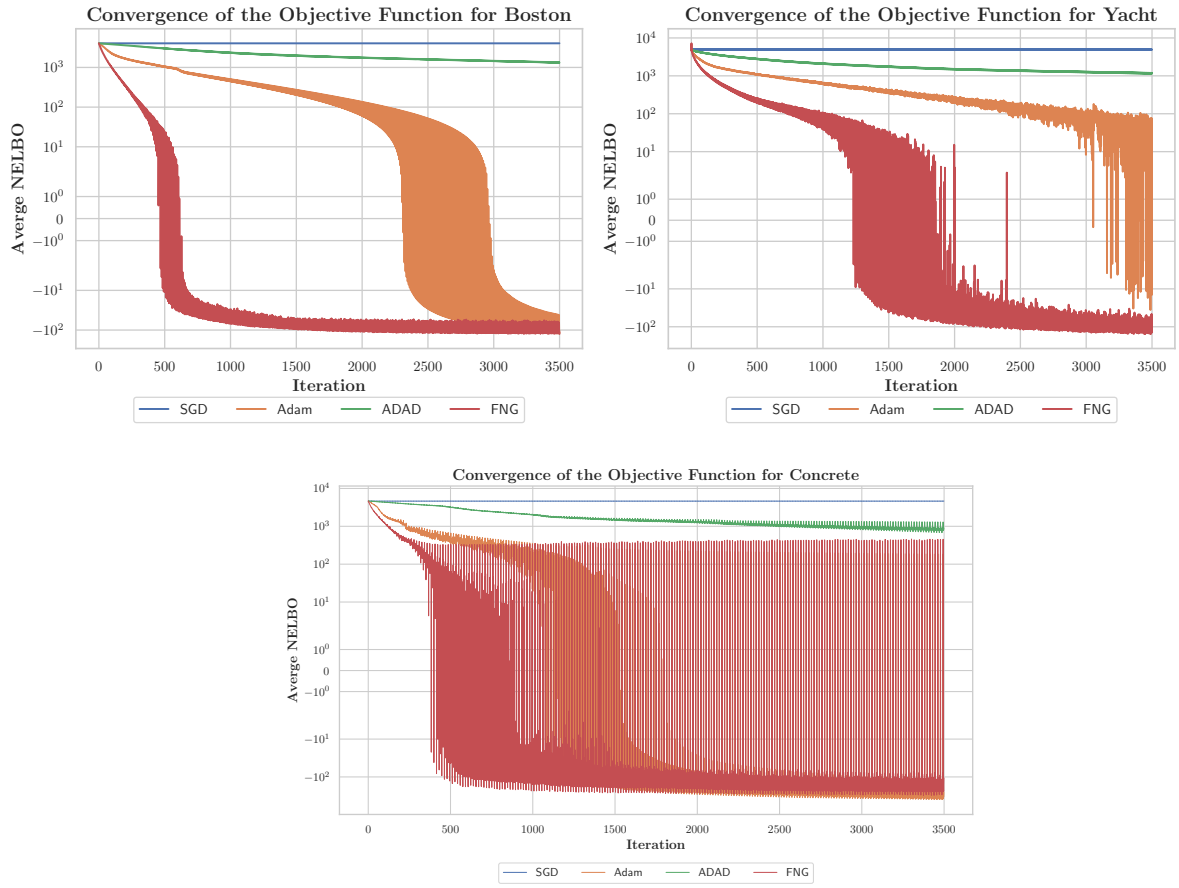


Figure 3.1: *Convergence of the Objective Functions when using the methods SGD (blue), Adam (orange), ADAD (green) and FNG (red) for training the CCGP model. The datasets are as follows: boston (top left), yacht (top right) and concrete (bottom). Each convergence graph is an average NELBO function of 15 different initialisations for the model.*

minimising the objective function for the different datasets. Particularly, the different AGMs struggled to minimise the objective function for the yacht dataset, whilst the FNG was able to converge to an appropriate minimum. Generally, the FNG scheme presented a faster convergence performance than the other methods.

3.10.2 Qualitative Assessment using the Motor Dataset

We use a one-dimensional dataset to provide a qualitative assessment of the different methods: the **motor** dataset ($N = 133, P = 1$) was built to test crash helmets and consists of one output with measurements of head acceleration in a simulated motorcycle accident.[3] We run two experiments with different percentage of data for training and testing. We define a Heteroscedastic-Gaussian likelihood $\prod_{n=1}^{N} \mathcal{N}(y_n|\mu_n, v_n)$, then each parameter associates a latent function where the mean $\mu_n = f_1(\mathbf{x}_n)$ and the variance $v_n = \exp(f_2(\mathbf{x}_n))$. All methods use two inducing latent functions u_1 and u_2, and the number of inducing points per latent funtion is $M = 20$. We use stochastic inference with a mini-batch size of 20 for training the CGP and CCGP models, while for the Augmented-output CCGP (ACCGP) strategy we use a mini-batch size of 10 per each output, this is to be fair in the amount of data used at each iteration for all methods. In the first case, we randomly split the data into 50% for training and testing respectively. In the second case, we randomly split the data into 75% for training and 25% for testing. Figures 3.2 and 3.3 show the predictive distribution reached by the methods with regard to each setting. In the graphs we include black crosses and red dots that represent the train and test data observations respectively.[4]

For the first case, we can see from Figure 3.2 that the CGP and CCGP present almost the same mean prediction, although it is more difficult for the former to quantify the uncertainty. For instance, in the time interval $(-1.75, -1.0)$ there is a flat trend in the data, but the CGP understimates such trend, while the CCGP shows a better quantification. The interval $(-1.0, 0.0)$ presents a sinusoidal behaviour without much noise, here our CCGP approach shows smaller error bars than the CGP. Both CGP and CCGP have a similar performance over the region $(0.0, 3.0)$ with slightly smaller error bars for the latter. The ACCGP approach shows a high performance for all data regions, better than CGP and CCGP for both $\mathbf{y}_1$ and $\mathbf{y}_2$. This strategy quantifies the uncertainty more appropriately in the flat zone $(-1.75, -1.0)$, also among $(-1.0, 0.0)$

[3] See http://vincentarelbundock.github.io/Rdatasets/datasets.html for motor data.

[4] Notice that in Figures 3.2 and 3.3 the data for Acceleration and Time are both standardised, i.e., zero mean and one standard deviation.

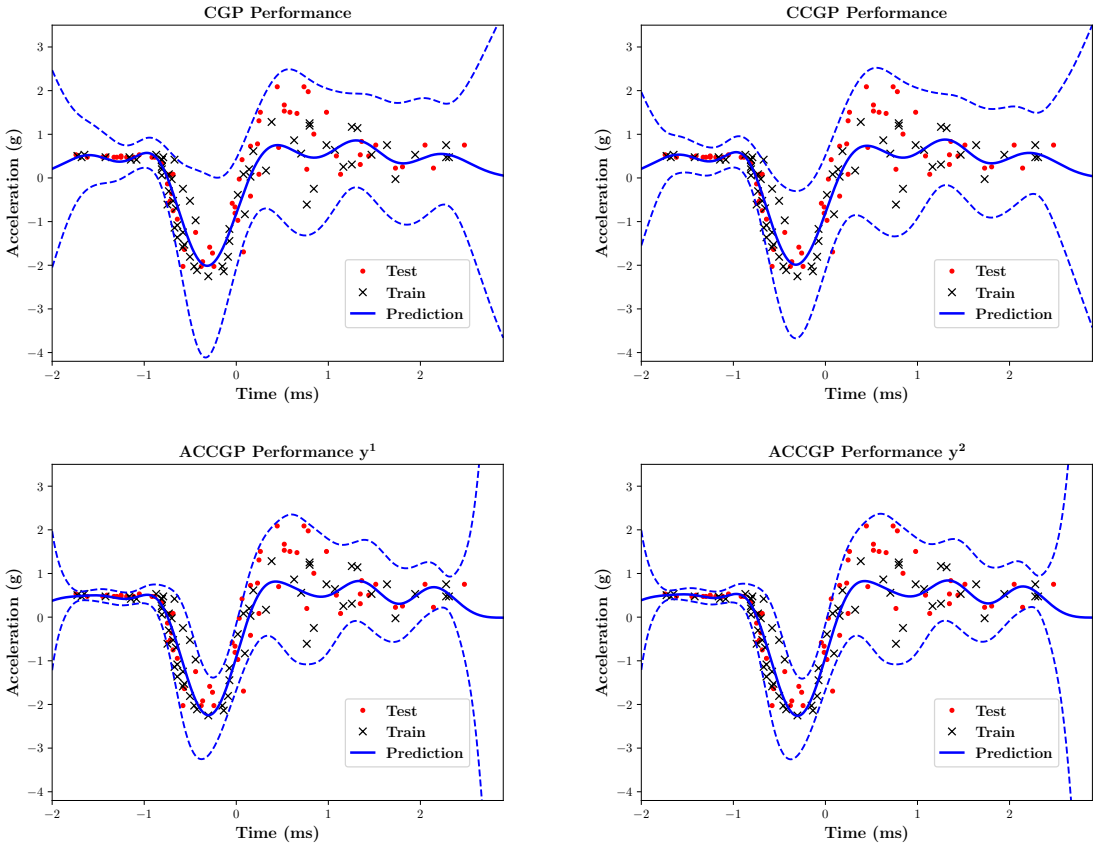


Figure 3.2: *Predictive distribution of CGP, CCGP and ACCGP ($\mathbf{y}_1$ and $\mathbf{y}_2$) over the motor dataset using a split of 50% for training and testing respectively. Each figure shows the predictive distribution; mean prediction (solid blue line) plus and minus two times the standard deviation (dashed blue line) for each input value. The black crosses represent the training data and the red dots the testing data.*

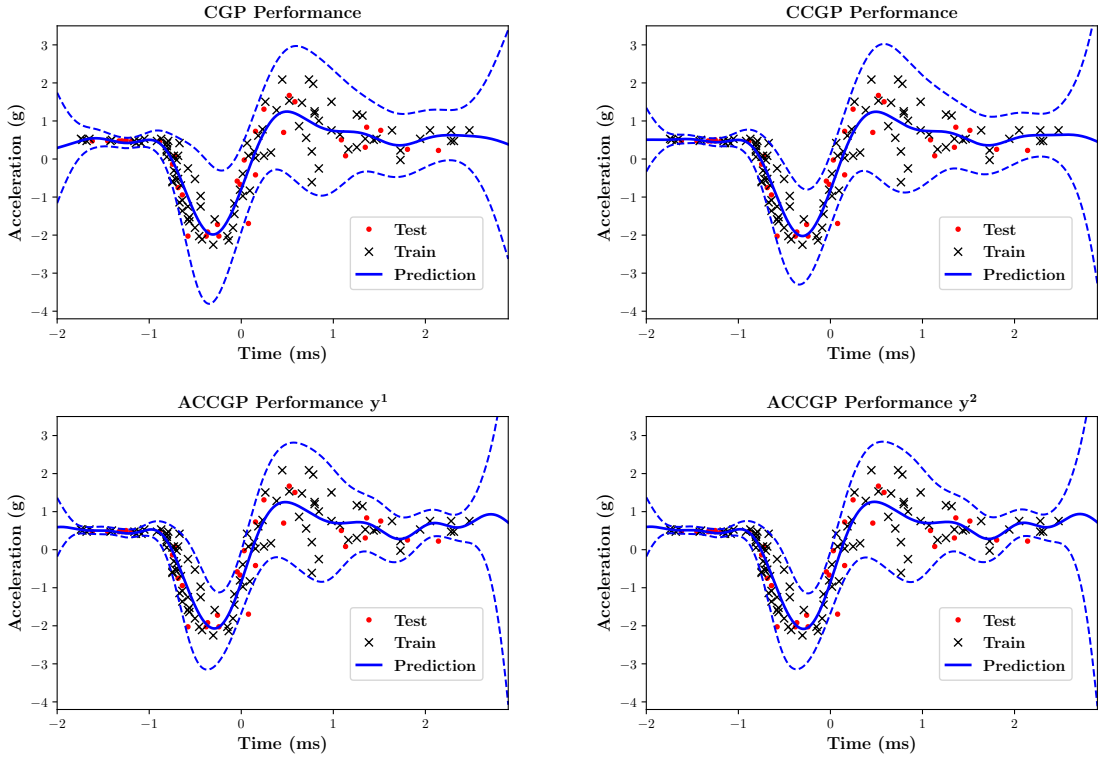


Figure 3.3: *Predictive distribution of CGP, CCGP and ACCGP ($\mathbf{y}_1$ and $\mathbf{y}_2$) over the motor dataset using a split of 75% and 25% for training and testing respectively. Each figure shows the predictive distribution; mean prediction (solid blue line) plus and minus two times the standard deviation (dashed blue line) for each input value. The black crosses represent the training data and red dots the testing data.*

and for the most noisy region of the data in $(0.0, 3.0)$. We can see from the figures that ACCGP's predictive distribution generalises the training data properly, while additionally shows to be consistent for the testing data. As an additional support to these results, Table 3.1 shows the NLPD computed over the training and testing data observations. For the second case, we can notice from Figure 3.3 that the predictive

Table 3.1: *NLPD achieved by the different methods over motor dataset for training and testing observations. Column split shows the percentage of data used for training and testing respectively. Lower values of NLPD refer to a better performance.*

	CGP		**CCGP**		**ACCGP $\mathbf{y}_1$**		**ACCGP $\mathbf{y}_2$**	
Split	**Train**	**Test**	**Train**	**Test**	**Train**	**Test**	**Train**	**Test**
50% 50%	0.491	0.601	0.444	0.536	0.219	0.418	**0.218**	**0.408**
75% 25%	0.372	0.275	0.297	0.180	**0.218**	**0.216**	0.221	0.213

distributions of the CGP and CCGP fit the data better than the first case where there was less amount of training data and more for testing. Again we can see that the predictive mean looks quite similar for CGP and CCGP approaches. Nonetheless, the main differences can be spotted in the time axis regions. In $(-1.75, -1.0)$, where data presents an almost steady behaviour, the CCGP reaches more confidence than CGP. Also, among the interval $(-1.0, 0.0)$ where there is a clear trend along the observations with low noise, the CCGP presents smaller error bars than CGP. For the remaining region in $(0.0, 3.0)$, which it is the noisiest one, both CGP and CCGP show similar deviations in the predictive distribution, though the latter attains a better uncertainty quantification at the end of the interval. With regard to the ACCGP, we can see that its predictive distribution for both $\mathbf{y}_1$ and $\mathbf{y}_2$ performs better than the first setting that had less training data, and also better than CGP and CCGP. We can see from Figure 3.3 that the mean and error bars for both ACCGP's outputs seem exactly the same. The ACCGP's predictive distribution is the most confident for the less noisy interval $(-1.75, -1.0)$, which makes sense regarding the data trend. Likewise, ACCGP approach shows smaller deviations between $(-1.0, 0.0)$ than CGP and CCGP. As per the data observations tendency in the noisy interval $(0.0, 3.0)$, ACCGP achieves a more appropriate uncertainty quantification in comparison to CGP and CCGP approaches. In order to support quantitatively these results, Table 3.1 shows the NLPD computed over the training and testing data observations.

3.10.3 Quantitative Assessment using diverse Datasets

In this subsection, we evaluate quantitatively the performance of our proposed methods reporting the NLPD over a test set. We run experiments over the following datasets: **motor** ($N = 133, P = 1$), **yacht** ($N = 308, P = 6$), **boston** ($N = 505, P = 13$) and **concrete** ($N = 1K, P = 8$) were previously introduced. Additionally, here we include **bike** ($N = 17.3K, P = 13$) that consists of an output with a daily count of rental bikes between years 2011 and 2012 in Capital bikeshare system; the dataset's features correspond to weather and seasonal information; **protein** ($N = 45.7K, P = 9$) is a dataset of physicochemical properties of protein tertiary structure, consists of one output that represents the measure of the average distance between the atoms of superimposed proteins, and nine physicochemical features; and **CTslice** ($N = 53.5K, P = 379$) contains 384 features extracted from images of computerised tomography, and consists of an output that denotes the relative location of the image slice on the axial axis of the human body.[5] For each dataset, we test three different likelihood functions:

$$\prod_{d=1}^{D}\prod_{n=1}^{N}\mathcal{N}(y_{d,n}|\mu_{d,n}, v_{d,n}), \quad \prod_{d=1}^{D}\prod_{n=1}^{N}\text{Beta}(y_{d,n}|a_{d,n}, b_{d,n}), \quad \prod_{d=1}^{D}\prod_{n=1}^{N}\text{Gamma}(y_{d,n}|\alpha_{d,n}, \beta_{d,n}),$$

where $D = 1$ for CGP and CCGP, and $D = 2$ for ACCGP. The LPFs are chained to Heterocedastic-Gaussian as $\mu_{d,n} = f_{d,1,n}$ and $v_{d,n} = \exp(f_{d,2,n})$; to Beta as $a_{d,n} = \exp(f_{d,1,n})$ and $b_{d,n} = \exp(f_{d,2,n})$; and to Gamma as $\alpha_{d,n} = \exp(f_{d,1,n})$ and $\beta_{d,n} = \exp(f_{d,2,n})$. All methods use two inducing latent functions u_1 and u_2. We re-scale the output of each datasets as per the statistical data type of the likelihood used in the experiment, i.e., $\mathbf{y} \in \mathbb{R}$ for the HG, $\mathbf{y} \in [0, 1]$ for the Bt and $\mathbf{y} \in \mathbb{R}^+$ for the Ga. We use the proposed fully natural gradient scheme for optimising all the model parameters employing a mini-batch size of 100 for the CGP and CCGP, while a mini-batch size of 50 for each output of the ACCGP strategy. We split the datasets using 75% for training and 25% for testing. We run 15 different parameters initialisations and choose the best to train the models, then we report the NLPD over the test set. Table 3.2 shows the NLPD reached by the models CGP, CCGP and strategy ACCGP (output $\mathbf{y}_1$) for each dataset with a specific likelihood (Lik) configuration. Also, it shows results for sizes of inducing points $M = 20$ and $M = 40$.

We can see from Table 3.2 that ACCGP is the most consistent method given that, in general, achieves the lowest NLPD for most of the different configurations. Regarding CCGP, Table 3.2 shows that it generally outperforms the CGP, given that

[5] See http://archive.ics.uci.edu/ml/datasets.php for datasets information.
See https://www.cs.toronto.edu/~delve/data/boston/bostonDetail.html for boston.

Table 3.2: *NLPD Achieved by the Different Methods, CGP, CCGP and ACCGP (output $\mathbf{y}_1$) over a Test set. Column Lik refers to the type of likelihood used; Heterocedastic Gaussian (HG), Beta (Bt) and Gamma (Ga). Colums with M refer to the number of inducing points used. Lower values of NLPD refer to a better performance.*

		M=20			M=40		
Lik	**Dataset**	**CGP**	**CCGP**	**ACCGP**	**CGP**	**CCGP**	**ACCGP**
HG	**motor**	0.50	0.53	**0.28**	0.98	0.74	**0.50**
	yacht	1.11	-0.45	**-0.60**	1.26	-0.44	**-0.57**
	boston	**1.27**	1.43	1.43	**1.38**	1.42	1.51
	concrete	1.15	0.98	**0.97**	1.19	1.00	**0.97**
	bike	1.52	1.45	**1.32**	1.50	1.43	**1.31**
	protein	1.27	1.25	**1.25**	**1.19**	1.20	1.25
	CTslice	1.05	1.03	**0.75**	1.03	1.37	**0.76**
Bt	**motor**	-0.60	-0.69	**-0.88**	-0.37	-0.60	**-0.65**
	yacht	-1.95	**-2.00**	-1.95	-1.83	-1.89	**-1.93**
	boston	**-0.28**	-0.23	-0.14	-0.13	**-0.26**	-0.21
	concrete	-0.01	-0.40	**-0.42**	-0.15	-0.36	**-0.40**
	bike	-0.44	-0.47	**-0.71**	-0.19	-0.50	**-0.72**
	protein	**-0.28**	-0.27	-0.26	**-0.29**	-0.27	-0.27
	CTslice	-0.23	-0.27	**-0.59**	-0.19	-0.19	**-0.60**
Ga	**motor**	-0.11	-0.16	**-0.49**	0.06	-0.28	**-0.50**
	yacht	-1.79	**-1.85**	-1.79	-1.65	-1.77	**-1.78**
	boston	**-0.28**	1.11	2.17	**-0.01**	0.12	1.90
	concrete	0.08	-0.37	**-0.39**	0.07	-0.34	**-0.37**
	bike	-0.26	-0.40	**-0.60**	-0.25	-0.42	**-0.62**
	protein	**-0.20**	-0.13	-0.13	**-0.21**	-0.14	-0.14
	CTslice	0.17	0.05	**-0.53**	0.16	-0.06	**-0.45**

in almost the 70% of the experiments reached a lower NLPD. Though, CGP showed better performance than CCGP and ACCGP for the datasets boston and protein. We notice that despite of changing the likelihood to train each dataset all methods CGP, CCGP and ACCGP tend to behave similarly. For intance, if we analyse the ranking of methods' performance on motor with HG likelihood we find that: ACCGP showed the best performance, followed by CCGP and CGP. Then, this same ranking order occurred when using Bt and Ga likelihoods. Likewise, this ranking tendency can be seen for concrete, bike and CTslice datasets. For yacht this pattern is only consistent in $M = 40$, while in $M = 20$ only for HG likelihood. So, for yacht with the other likelihoods Bt and Ga in $M = 20$, the ranking changes to CCGP best, followed by ACCGP and CGP with the same performance. For boston, the ranking is CGP with the lowest NLPD, followed by CCGP and then ACCGP for all likelihoods in $M = 20$ and $M = 40$, with the only exception of Bt likelihood in $M = 40$. For protein, the ranking is CGP best, followed by CCGP and then ACCGP for all likelihoods in $M = 20$ and $M = 40$, with the only exception of HG likelihood in $M = 20$ where ACCGP presented the lowest NLPD followed by CCGP and CGP with the same performance.

3.10.4 An Application of the CCGPs for Datasets with Multiple Annotators

Additional experiments for an application of the CCGP model using datasets with multiple annotators are presented in the paper (iii) of Appendix L. A dataset with multiple annotators is usually associated to a supervised learning task, where the labelling process is carried out by crowds with different levels of experience (Raykar et al., 2010). For instance, in a medical diagnostic, an senior doctor in medicine might conclude (or label) the presence of an illness c_1 from a computed tomography image of a patient; whilst other doctor without much expertise could label the presence of a different illness, say c_2, from the same tomography image. Therefore, the goal standard or ground truth regarding the illness becomes a hidden variable. In supervised learning, usually each input data observation, $\mathbf{x}_n$, is assigned to a single, y_n, that represents the ground truth. Though, in the context of multiple annotators, instead of having the ground truth, we have multiple labels provided by R annotators with different ranges of expertise for an n-th observation. In real-world scenarios is common to find that the each annotator r only labels $|N_r| \leq N$ samples, where $|N_r|$ is the set's cardinality, $N_r \subseteq \{1, ..., N\}$ that contains the indexes of samples labelled by the r-th annotator. Therefore, our collection of input and output data observations with multiple annota-

tor can be expressed as: $\mathbf{X} = \{\mathbf{x}_n\}_{n=1}^{N}$ and $\mathbf{y} = \{y_n^r\}_{n\in N, r\in R_n}$, respectively; for which $R_n \subseteq \{1, ..., R\}$ is a set that contains the indexes of the annotators that labelled the n-th instance, and y_n^r is the output or annotation assigned by the r-th labeller to the n-th observation (Zhu et al., 2019).

By means of the CCGP model, we can build a probabilistic framework able to code the annotators' expertise as a function of the input observations, and also exploit the correlations between the annotators' answers. Such a probabilistic framework consists of a likelihood that involves all the multiple annotations of the data observations, where the likelihood's parameters are modelled as LPFs that follow correlated GP priors. For instance, if we aim to model data observations with categorical outputs (Rodrigues et al., 2013), i.e., $y_n^r \in \{1, ..., K\}$, the likelihood function can be expressed as:

$$p(\mathbf{y}|\mathbf{f}) = \prod_{n=1}^{N}\prod_{r\in R_n}\left(\prod_{k=1}^{K}\psi_k(\mathbf{x}_n)^{\delta(y_n^r,k)}\right)^{\psi_{K+r}(\mathbf{x}_n)}\left(\frac{1}{K}\right)^{(1-\psi_{K+r}(\mathbf{x}_n))},$$

with the likelihood's parameters chained to the GP latent functions as:

$$\psi_k(\mathbf{x}_n) = \frac{\exp\left(f_k\left(\boldsymbol{x}_n\right)\right)}{\sum_{j=1}^{K}\exp\left(f_j\left(\boldsymbol{x}_n\right)\right)}, \quad \psi_{K+r}(\mathbf{x}_n) = \frac{1}{2} + \frac{1}{2}\text{sign}(f_{K+r}(\mathbf{x}_n)),$$

where $\text{sign}(\cdot)$ is a function that returns the sign of a real number; and we have also introduced an indicator function, $\delta\left(y_n^r, k\right) = 1$ if $y_n^r = k$, otherwise $\delta\left(y_n^r, k\right) = 0$. It is worth noticing that the total number of LPFs chained to the likelihood is $J = K + R$.

On the other hand, if we aim to model data observations for a regression task (Ruiz et al., 2019), we can consider that each, $y_n^r \in \mathbb{R}$, is a corrupted version of the unknown ground truth y_n, thus the likelihood function can be written as:

$$p(\mathbf{y}|\mathbf{f}) = \prod_{n=1}^{N}\prod_{r\in R_n}\mathcal{N}\left(y_n^r|y_n, v_n^r\right),$$

where, $y_n = \psi_1(\mathbf{x}_n) = f_1(\mathbf{x}_n)$, represents the ground truth (or gold standard) modelled as a LPF and $v_n^r = \psi_{r+1}(\mathbf{x}_n) = \exp(f_{r+1}(\mathbf{x}_n))$ is the r-th error-variance of the n-th observation, also modelled as a LPF. Paper (iii) in Appendix L presents a detailed explanation of the CCGP model for datasets with multiple annotators, it also includes experiments that show how such a model helps to improve the modelling of labellers' behaviour for synthetic, semi-synthetic and real-world datasets.

3.10.5 Discussion

Through the different experiments, we found that the FNG scheme usually helped to find a better solution during the inference process for training the CCGP model in comparison to the AGMs. In general, experimental results showed that CCGP attained richer predictive distributions than CGP. Introducing correlations between the latent functions allowed the CCGP model to improve its uncertainty quantification capabilities. Furthermore, we found that training the model through an augmented configuration, that we called ACCGP, permitted to enhance the generalisation properties of the model. Indeed, this ACCGP approach behaved robustly when having little training data observations. For instance, in the first experiment with the motor dataset that used only the 50% of data observations for training, the ACCGP accomplished a predictive distribution with properties quite similar to the one achieved in the case with 75% of data observations, being a bit less confident in the former case. Although, such training strategy demands additional hyper-parameters to be trained due to the augmentation of the outputs, and also additional matrix operations for computing the marginal posterior distributions, $q(\mathbf{f}_{d,j})$, during the inference process. On the other hand, we assessed the performance of our methods in the context of small and large datasets with low and high dimensionality. The results showed as per Table 3.2 that CCGP generally accomplished lower NLPD metrics in comparison to CGP, this supports the premise that our model provides robust solutions adequate for making predictions with appropriate uncertainty quantification. Furthermore, we found that in most of the configurations, it is possible to obtain higher performances for CCGP when trained using the augmenting strategy, ACCGP. We believe that the improvement attained when using the ACCGP is due to the additional set of linear coefficient parameters that influence the gradient updates of the variational posterior distributions. Also, due to the fact that the scaling factors S_A and S_B in Eq. (3.26) put additional weight over the likelihood term. For example, if the mini-batch size is just one sample per output then the scaling factors become $S_A = S_B = N$. Unlike the single-output case, if we use a mini-batch size of two samples then the scaling factor is limited to $S = N/2$ as per the two mini-batches expression in Eq. (3.27).

3.11 Summary

In this chapter, we have introduced the correlated chained Gaussian processes model. We showed that the introduction of a linear model of coregionalisation allows to exploit correlations between the likelihood's LPFs, thereby outperforming the former methods based on independent GP assumptions. We showed that the model can achieve robust predictive properties for different types of likelihoods and various types of datasets. We provided a derivation of a FNG scheme that improved the inference process of the CCGP model in comparison to AGMs. Furthermore, we proposed and tested a training strategy based on output augmentation that permits to boost predictive properties of the CCGP model. In the next chapter, we will focus on broadly addressing the issues associated to poor local optima solutions that emerge when using the CCGP model in the context of multiple heterogeneous outputs. To this end, we will particularly derive the FNG scheme for improving inference of the model when having multiple outputs, and will provide different comparative results against AGMs and a hybrid strategy (NG+Adam) (Salimbeni et al., 2019).

Chapter 4

Heterogeneous Multi-Output GPs Model with a Linear Model of Coregionalisation

In chapter 3, we introduced the CCGP model that relies on an linear model of coregionalisation for generating correlated LPFs. The extrapolation of the CCGP model to the context of multiple outputs gives rise to the Heterogeneous Multi-Output GP model (Moreno-Muñoz et al., 2018). It is worth noticing that the idea in chapter 3, of correlating the LPFs for modelling the likelihood's parameters, was motivated by the fact of building a more realistic setting for which the parameters were correlated as part of a latent stochastic process that inherently affects them jointly. Indeed, through the different experiments carried out in chapter 3, we showed how the correlation assumption between the LPFs improved the single-output model flexibility to better quantify the uncertainty in comparison to the independent case. On the other hand, the HetMOGP model is particularly built upon the assumption of knowledge transferability between the multiple outputs for which the outputs could be a mix of different statistical data types, for instance the outputs can be: continuous, categorical, binary or discrete variables that can associate different likelihood functions. Such a heterogeneity of the likelihoods and the multiple LPFs chained to their parameters, make the model suffer a strong conditioning between the variational posterior distribution, the multiple hyper-parameters of the GP prior and the inducing points (Van der Wilk, 2018).

The HetMOGP model can be seen as an extrapolation of the CCGP model, but for multiple heterogeneous outputs, it is also built upon a linear combination of Q latent functions; where each latent function demands a treatment based on the inducing

variables framework. On this model then, such strong conditionings are enhanced even more due to the dependence of inducing points per underlying latent function, and the presence of additional linear combination coefficients. Those problems hinder the AGMs to appropriately fit the parameters and hyper-parameters of the models. In order to improve the inference of the HetMOGP model, in this chapter we make use of the mechanisms for optimisation introduced in chapter 2, by deriving a fully natural gradient scheme for suitably fitting the model. We carry out experiments using toy and real datasets that involve heterogeneous outputs. To the best of our knowledge the NG method has not been performed over any MOGP model before. Hence, in this work we contribute to show how a NG method used in a full scheme over the MOGP's parameters and kernel hyper-parameters alleviates the strong conditioning problems. This, by achieving better local optima solutions with higher test performance rates than Adam and SGD. Moreover, we explore for the first time in a MOGP model the behaviour of the hybrid strategy NG+Adam proposed by Salimbeni et al. (2018), and provide comparative results to our proposed FNG scheme.

4.1 The Likelihood Function for the HetMOGP

The HetMOGP model is an extension of the Multi-Output GP that allows different kinds of likelihoods as per the statistical data type each output demands (Moreno-Muñoz et al., 2018). For instance, if we have two outputs problem, where one output is binary $y_1 \in \{0, 1\}$ while the other is a real value $y_2 \in \mathbb{R}$, we can assume our likelihood as the product of a Bernoulli and Gaussian distribution for each output respectively. In general the HetMOGP likelihood for D outputs can be written as:

$$p(\mathbf{y}|\mathbf{f}) = \prod_{n=1}^{N} \prod_{d=1}^{D} p(y_{d,n}|\psi_{d,1}(\mathbf{x}_n), ..., \psi_{d,J_d}(\mathbf{x}_n)), \tag{4.1}$$

where the vector $\mathbf{y} = [\mathbf{y}_1^\top, ..., \mathbf{y}_D^\top]^\top$ groups all the output observations and each $\psi_{d,j}(\mathbf{x}_n)$ represents the j-th parameter that belongs to the d-th likelihood. It is worth noticing that each output vector $\mathbf{y}_d$ is generated by a particular set of input observations $\mathbf{X}_d$. Though, in order to ease the explanation of the model and to be consistent with the equation above, we have assumed that all outputs $\mathbf{y}_d = [y_{d,1}, ..., y_{d,N}]^\top$ are related to the same input observations $\mathbf{X} = [\mathbf{x}_1, ..., \mathbf{x}_N]^\top \in \mathbb{R}^{N \times P}$. Each likelihoods' parameter $\psi_{d,j}(\mathbf{x}_n)$ is chained to a latent function $f_{d,j}(\cdot)$ that follows a GP prior, through a link function $\phi(\cdot)$, i.e., $\psi_{d,j}(\mathbf{x}_n) = \phi(f_{d,j}(\mathbf{x}_n))$. For instance, if we have two outputs

where the first likelihood is a Heteroscedastic Gaussian, then its parameters mean and variance are respectively chained as $\psi_{1,1}(\mathbf{x}_n) = f_{1,1}(\mathbf{x}_n)$ and $\psi_{1,2}(\mathbf{x}_n) = \exp(f_{1,2}(\mathbf{x}_n))$; if the second likelihood is a Gamma, its parameters are linked as $\psi_{2,1}(\mathbf{x}_n) = \exp(f_{2,1}(\mathbf{x}_n))$ and $\psi_{2,2}(\mathbf{x}_n) = \exp(f_{2,2}(\mathbf{x}_n))$ (Saul et al., 2016). Notice that J_d accounts for the number of latent functions necessary to parametrise the d-th likelihood, thus the total number of functions $f_{d,j}(\cdot)$ associated to the model becomes $J = \sum_{d=1}^{D} J_d$. Each $f_{d,j}(\cdot)$ is considered a LPF that comes from a LMC as follows:

$$f_{d,j}(\mathbf{x}) = \sum_{q=1}^{Q} \sum_{i=1}^{R_q} w_{d,j,q}^{i} u_q^i(\mathbf{x}), \tag{4.2}$$

where $u_q^i(\mathbf{x})$ are IID samples from GPs $u_q(\cdot) \sim \mathcal{GP}(0, k_q(\cdot,\cdot))$ and $w_{d,j,q}^{i} \in \mathbb{R}$ is a linear combination coefficient. In chapter 5, we introduce a different way to model $f_{d,j}(\mathbf{x})$ based on convolution processes. For the sake of future explanations let us assume that $R_q = 1$. In this way the number of LCCs per latent function $u_q(\cdot)$ becomes J. The coefficients per function $u_q(\cdot)$ can be grouped in a vector $\mathbf{w}_q = [w_{1,1,q}, ..., w_{1,J_1,q}, ..., w_{D,J_D,q}]^\top \in \mathbb{R}^{J\times 1}$; and we can cluster all vectors $\mathbf{w}_q$ in a specific vector of LCCs $\mathbf{w} = [\mathbf{w}_1^\top, ..., \mathbf{w}_Q^\top]^\top \in \mathbb{R}^{QJ\times 1}$.

4.2 The Inducing Points Method

A common approach for reducing computational complexity in GP models is to augment the GP prior with a set of *inducing variables*. For the specific case of the Het-MOGP model with LMC prior, the vector of *inducing variables* $\mathbf{u} = [\mathbf{u}_1^\top, ..., \mathbf{u}_Q^\top]^\top \in \mathbb{R}^{QM\times 1}$ is built from $\mathbf{u}_q = [u_q(\mathbf{z}_q^{(1)}), ..., u_q(\mathbf{z}_q^{(M)})]^\top \in \mathbb{R}^{M\times 1}$. Notice that the vector $\mathbf{u}_q$ is constructed by additional evaluations of the functions $u_q(\cdot)$ at some unknown inducing points $\mathbf{Z}_q = [\mathbf{z}_q^{(1)}, ..., \mathbf{z}_q^{(M)}]^\top \in \mathbb{R}^{M\times P}$. The vector of all inducing variables can be expressed as $\mathbf{z} = [\text{vec}(\mathbf{Z}_1)^\top, ..., \text{vec}(\mathbf{Z}_Q)^\top]^\top \in \mathbb{R}^{QMP\times 1}$ (Snelson and Ghahramani, 2006; Titsias, 2009). We can write the augmented GP prior as follows,

$$p(\mathbf{f}|\mathbf{u})p(\mathbf{u}) = \prod_{d=1}^{D}\prod_{j=1}^{J_d} p(\mathbf{f}_{d,j}|\mathbf{u}) \prod_{q=1}^{Q} p(\mathbf{u}_q), \tag{4.3}$$

where $\mathbf{f} = [\mathbf{f}_{1,1}^\top, ..., \mathbf{f}_{1,J_1}^\top, ..., \mathbf{f}_{D,J_D}^\top]^\top$ is a vector function built from the vectors $\mathbf{f}_{d,j} = [f_{d,j}(\mathbf{x}_1), ..., f_{d,j}(\mathbf{x}_N)]^\top \in \mathbb{R}^{N\times 1}$. Following the conditional Gaussian properties we can express, $p(\mathbf{f}_{d,j}|\mathbf{u}) = \mathcal{N}(\mathbf{f}_{d,j}|\mathbf{A}_{\mathbf{f}_{d,j}\mathbf{u}}\mathbf{u}, \widetilde{\mathbf{Q}}_{\mathbf{f}_{d,j}\mathbf{f}_{d,j}})$ and $p(\mathbf{u}) = \mathcal{N}(\mathbf{u}|\mathbf{0}, \mathbf{K}_{\mathbf{uu}})$, where the matrix $\mathbf{K}_{\mathbf{uu}} \in \mathbb{R}^{QM\times QM}$ is a block-diagonal with blocks $\mathbf{K}_{\mathbf{u}_q\mathbf{u}_q} \in \mathbb{R}^{M\times M}$ built from

evaluations of $\text{cov}\left[u_q(\cdot), u_q(\cdot)\right] = k_q(\cdot,\cdot)$ between all pairs of inducing points $\mathbf{Z}_q$ respectively; and we have introduced the following definitions, $\mathbf{A}_{\mathbf{f}_{d,j}\mathbf{u}} = \mathbf{K}_{\mathbf{f}_{d,j}\mathbf{u}}\mathbf{K}_{\mathbf{uu}}^{-1}$, $\widetilde{\mathbf{Q}}_{\mathbf{f}_{d,j}\mathbf{f}_{d,j}} = \mathbf{K}_{\mathbf{f}_{d,j}\mathbf{f}_{d,j}} - \mathbf{Q}_{\mathbf{f}_{d,j}\mathbf{f}_{d,j}}$, $\mathbf{Q}_{\mathbf{f}_{d,j}\mathbf{f}_{d,j}} = \mathbf{K}_{\mathbf{f}_{d,j}\mathbf{u}}\mathbf{K}_{\mathbf{uu}}^{-1}\mathbf{K}_{\mathbf{uf}_{d,j}}$, $\mathbf{K}_{\mathbf{f}_{d,j}\mathbf{u}} = \mathbf{K}_{\mathbf{uf}_{d,j}}^{\top}$. Here the covariance matrix $\mathbf{K}_{\mathbf{f}_{d,j}\mathbf{f}_{d,j}} \in \mathbb{R}^{N\times N}$ is built from the evaluation of all pairs of input data $\mathbf{X}$ in the covariance function $\text{Cov}\left[f_{d,j}(\cdot), f_{d,j}(\cdot)\right] = \sum_{q=1}^{Q} w_{d,j,q}^2 k_q\left(\cdot,\cdot\right)$; and the cross covariance matrix $\mathbf{K}_{\mathbf{f}_{d,j}\mathbf{u}} = [\mathbf{K}_{\mathbf{f}_{d,j}\mathbf{u}_1}, ..., \mathbf{K}_{\mathbf{f}_{d,j}\mathbf{u}_Q}] \in \mathbb{R}^{N\times QM}$ is constructed with the blocks $\mathbf{K}_{\mathbf{f}_{d,j}\mathbf{u}_q} \in \mathbb{R}^{N\times M}$, formed by the evaluations of $\text{Cov}\left[f_{d,j}(\cdot), u_q(\cdot)\right] = w_{d,j,q}k_q(\cdot,\cdot)$ between inputs $\mathbf{X}$ and $\mathbf{Z}_q$. Each kernel covariance $k_q\left(\cdot,\cdot\right)$ has an Exponentiated Quadratic (EQ) form as follows:

$$\mathcal{E}(\boldsymbol{\tau}|\mathbf{0},\mathbf{L}) = \frac{|\mathbf{L}|^{-1/2}}{(2\pi)^{P/2}}\exp\left[-\frac{1}{2}\boldsymbol{\tau}^{\top}\mathbf{L}^{-1}\boldsymbol{\tau}\right], \tag{4.4}$$

where $\boldsymbol{\tau} := \mathbf{x} - \mathbf{x}'$ and $\mathbf{L}$ is a diagonal matrix of length-scales. Thus, each $k_q\left(\mathbf{x},\mathbf{x}'\right) = \mathcal{E}(\boldsymbol{\tau}|\mathbf{0},\mathbf{L}_q)$.

4.3 The Evidence Lower Bound

We follow a VI derivation similar to the one used for single output GPs (Hensman et al., 2013; Saul et al., 2016). This approach allows the use of HetMOGP for large data. The goal is to approximate the true posterior $p(\mathbf{f},\mathbf{u}|\mathbf{y})$ with a variational distribution $q(\mathbf{f},\mathbf{u})$ by optimising the following negative ELBO:

$$\tilde{\mathcal{L}} = \sum_{n=1}^{N}\sum_{d=1}^{D}\mathbb{E}_{q(\mathbf{f}_{d,1})\cdots q(\mathbf{f}_{d,J_d})}\left[g_{d,n}\right] + \sum_{q=1}^{Q}\mathbb{D}_{KL}\left(q(\mathbf{u}_q)\|p(\mathbf{u}_q)\right), \tag{4.5}$$

where $g_{d,n} = -\log p(y_{d,n}|\psi_{d,1}(\mathbf{x}_n), ..., \psi_{d,J_d}(\mathbf{x}_n))$ is the NLL function associated to each output, and we have set a tractable posterior $q(\mathbf{f},\mathbf{u}) = p(\mathbf{f}|\mathbf{u})q(\mathbf{u})$, where $p(\mathbf{f}|\mathbf{u})$ is already defined in Eq. (4.3), $q(\mathbf{u}|\mathbf{m},\mathbf{V}) = \prod_{q=1}^{Q} q(\mathbf{u}_q)$, and each $q(\mathbf{u}_q) = \mathcal{N}(\mathbf{u}_q|\mathbf{m}_q,\mathbf{V}_q)$ is a Gaussian distribution with mean $\mathbf{m}_q$ and covariance $\mathbf{V}_q$ (Hensman et al., 2015a) (see Appendix D for details on the ELBO derivation). The above expectation associated to the NLL is computed using the marginal posteriors,

$$q(\mathbf{f}_{d,j}) := \mathcal{N}(\mathbf{f}_{d,j}|\tilde{\mathbf{m}}_{\mathbf{f}_{d,j}}, \tilde{\mathbf{V}}_{\mathbf{f}_{d,j}}), \tag{4.6}$$

with the following definitions, $\tilde{\mathbf{m}}_{\mathbf{f}_{d,j}} := \mathbf{A}_{\mathbf{f}_{d,j}\mathbf{u}}\mathbf{m}$, $\tilde{\mathbf{V}}_{\mathbf{f}_{d,j}} := \mathbf{K}_{\mathbf{f}_{d,j}\mathbf{f}_{d,j}} + \mathbf{A}_{\mathbf{f}_{d,j}\mathbf{u}}(\mathbf{V} - \mathbf{K}_{\mathbf{uu}})\mathbf{A}_{\mathbf{f}_{d,j}\mathbf{u}}^{\top}$, where mean $\mathbf{m} = [\mathbf{m}_1^{\top}, ..., \mathbf{m}_Q^{\top}]^{\top} \in \mathbb{R}^{QM\times 1}$ and the covariance matrix $\mathbf{V} \in \mathbb{R}^{QM\times QM}$

is a block-diagonal matrix with blocks given by $\mathbf{V}_q \in \mathbb{R}^{M\times M}$.[1] The objective function derived in Eq. (4.5) for the HetMOGP model with LMC requires fitting the parameters of each posterior $q(\mathbf{u}_q)$, the inducing points $\mathbf{z}$, the kernel hyper-parameters $\mathbf{l}_{\text{kern}} = [\text{diag}(\mathbf{L}_1)^\top, ..., \text{diag}(\mathbf{L}_Q)^\top]^\top$ and the coefficients $\mathbf{w}$. With the aim to fit said variables in a FNG scheme, later on we will apply the VO perspective on Eq. (4.5) for inducing randomness and gain exploration over $\mathbf{z}$, $\mathbf{l}_{\text{kern}}$ and $\mathbf{w}$; and by means of the MDA we will derive the inference updates for all the model's variables.

4.4 Deriving a Fully Natural Gradient Scheme for HetMOGP model with LMC

This section describes how to derive the FNG updates for optimising the HetMOGP model based on an LMC. We first detail how to induce an exploratory distribution over the hyper-parameters and inducing points, then we write down the MDA for the model and derive the update equations. Later on, we get into specific details about the algorithm's implementation.

4.4.1 An Exploratory Distribution for HetMOGP with LMC

As we previously mentioned in section 3.6.1, the kernel hyper-parameters and inducing points of the a sparse variational GP model have usually been treated as deterministic variables. In similar way to the CCGP model, here we adopt the VO perspective as a mechanism to induce stochasticity over such variables, for gaining exploration that allows us to find better solutions during the inference process (Staines and Barber, 2013). Therefore, we define and connect random real vectors to the variables through a link function $\phi(\cdot)$ as follows: for the inducing points $\mathbf{z} = \boldsymbol{\theta}_{\mathbf{z}}$, for the kernel hyper-parameters $\mathbf{l}_{\text{kern}} = \exp(\boldsymbol{\theta}_{\mathbf{L}})$ with $\boldsymbol{\theta}_{\mathbf{L}} = [\boldsymbol{\theta}_{\mathbf{L}_1}^\top, ..., \boldsymbol{\theta}_{\mathbf{L}_Q}^\top]^\top \in \mathbb{R}^{QP\times 1}$, and for the vector of LCC $\mathbf{w} = \boldsymbol{\theta}_{\mathbf{w}}$, that are used to generate the LPFs in Eq. (4.2). We have defined the real random vectors $\boldsymbol{\theta}_{\mathbf{Z}} \in \mathbb{R}^{QMP\times 1}$, $\boldsymbol{\theta}_{\mathbf{L}_q} \in \mathbb{R}^{P\times 1}$ and $\boldsymbol{\theta}_{\mathbf{w}} \in \mathbb{R}^{QJ\times 1}$ to link the set of inducing points, the kernel hyper-parameters per latent function $u_q(\cdot)$, and the vector $\mathbf{w}$ of LCCs, respectively. We cluster the random vectors defining $\boldsymbol{\theta} = [\boldsymbol{\theta}_{\mathbf{z}}^\top, \boldsymbol{\theta}_{\mathbf{L}}^\top, \boldsymbol{\theta}_{\mathbf{w}}^\top]^\top \in \mathbb{R}^{(QMP+QP+QJ)\times 1}$ to refer to all the parameters in a single variable. Having defined the previous relations, we can specify an exploratory distribution $q(\boldsymbol{\theta}) := \mathcal{N}(\boldsymbol{\theta}|\boldsymbol{\mu}^D, \boldsymbol{\Sigma}^D)$ for applying the VO framework from Eq. (2.2), though for our case the objective to

[1] Each marginal posterior derives from: $q(\mathbf{f}_{d,j}) = \int p(\mathbf{f}_{d,j}|\mathbf{u})q(\mathbf{u})d\mathbf{u}$.

bound is $\tilde{\mathcal{L}}$, already derived in Eq. (4.5) for the HetMOGP with an LMC. Therefore our VO bound is defined as follows:

$$\tilde{\mathcal{F}} = \mathbb{E}_{q(\boldsymbol{\theta})}\left[\tilde{\mathcal{L}}\right] + \mathbb{D}_{KL}(q(\boldsymbol{\theta})||p(\boldsymbol{\theta})), \tag{4.7}$$

where $p(\boldsymbol{\theta}) = \mathcal{N}(\boldsymbol{\theta}|\mathbf{0}, \lambda_1^{-1}\mathbf{I})$ is a Gaussian distribution with precision λ_1 (Khan et al., 2017b).

4.4.2 Mirror Descent Algorithm for the HetMOGP with LMC

With the aim to minimise the VO objective in Eq. (4.7), we take advantage of the MDA formulation from Eq. (2.5), which allow us to derive closed-form updates for fitting the parameters and hyper-parameters of the model. We use the parametrisation in the mean-parameter space of the distributions $q(\mathbf{u}_q)$ and $q(\boldsymbol{\theta})$ by defining $\boldsymbol{\rho}_q = \{\mathbf{m}_q, \mathbf{m}_q\mathbf{m}_q^\top + \mathbf{V}_q\}$ and $\boldsymbol{\eta}^D = \{\boldsymbol{\mu}^D, \boldsymbol{\mu}^D\boldsymbol{\mu}^{D\top} + \boldsymbol{\Sigma}^D\}$. Since the HetMOGP model can be seen as a generalisation of the CCGP model to the context of multiple heterogeneous outputs, then we can express the MDA exactly as the Eq. (3.19) previously derived for the CCGP model.

4.4.3 Fully Natural Gradient Updates

Given that the MDA for the HetMOGP with LMC is exactly the same as the one derived in Eq. (3.19) for the CCGP model, then by solving for such a MDA we obtain the following FNG updates:

$$\boldsymbol{\Sigma}^{D\,-1}_{t+1} = \boldsymbol{\Sigma}^{D\,-1}_{t} + 2\alpha_t \hat{\nabla}_{\boldsymbol{\Sigma}^D}\tilde{\mathcal{F}}_t \tag{4.8}$$

$$\boldsymbol{\mu}^D{}_{t+1} = \boldsymbol{\mu}^D{}_t - \alpha_t \boldsymbol{\Sigma}^D{}_{t+1}\hat{\nabla}_{\boldsymbol{\mu}^D}\tilde{\mathcal{F}}_t + \gamma_t \boldsymbol{\Sigma}^D{}_{t+1}\boldsymbol{\Sigma}^{D\,-1}_{t}(\boldsymbol{\mu}^D{}_t - \boldsymbol{\mu}^D{}_{t-1}) \tag{4.9}$$

$$\mathbf{V}^{-1}_{q,t+1} = \mathbf{V}^{-1}_{q,t} + 2\beta_t \hat{\nabla}_{\mathbf{V}_q}\tilde{\mathcal{F}}_t \tag{4.10}$$

$$\begin{aligned}\mathbf{m}_{q,t+1} = \mathbf{m}_{q,t} &- \beta_t \mathbf{V}_{q,t+1}\hat{\nabla}_{\mathbf{m}_q}\tilde{\mathcal{F}}_t \\ &+ \upsilon_t \mathbf{V}_{q,t+1}\mathbf{V}^{-1}_{q,t}(\mathbf{m}_{q,t} - \mathbf{m}_{q,t-1}),\end{aligned} \tag{4.11}$$

where $\alpha_t = \tilde{\alpha}_t/(1-\tilde{\gamma}_t)$, $\beta_t = \tilde{\beta}_t/(1-\tilde{\upsilon}_t)$, $\gamma_t = \tilde{\gamma}_t/(1-\tilde{\gamma}_t)$ and $\upsilon_t = \tilde{\upsilon}_t/(1-\tilde{\upsilon}_t)$ are positive step-size parameters. It is worth noticing, that the main difference when applying the equations above, either for CCGP model or for HetMOGP with LMC, relies on the influence of the gradients; the CCGP would be influenced by only one output ($D = 1$), while the HetMOGP with LMC would be influenced by the multiple heterogeneous outputs; see Appendix F for details on the gradients derivation.

4.5 Implementation

The implementation of the FNG scheme for the HetMOGP model with LMC follows the same procedure of chapter 3.7; where, in order to alleviate the complexity issues of inverting the covariance matrix $\mathbf{\Sigma}^D$, we assume $\mathbf{\Sigma}^D = \text{diag}(\boldsymbol{\sigma}^{D\,2})$, for which $\boldsymbol{\sigma}^D$ is a vector of standard deviations, and $\text{diag}(\boldsymbol{\sigma}^{D\,2})$ represents a matrix with the elements of $\boldsymbol{\sigma}^{D\,2}$ on its diagonal. Also, we estimate the Hessian $\hat{\nabla}^2_{\boldsymbol{\theta}\boldsymbol{\theta}}\tilde{\mathcal{L}}$, associated to the gradient $\hat{\nabla}_{\mathbf{\Sigma}^D}\tilde{\mathcal{F}}$ in Eq. (4.8), through the Gauss-Newton approximation $\hat{\nabla}^2_{\boldsymbol{\theta}\boldsymbol{\theta}}\tilde{\mathcal{L}} \approx \hat{\nabla}_{\boldsymbol{\theta}}\tilde{\mathcal{L}} \circ \hat{\nabla}_{\boldsymbol{\theta}}\tilde{\mathcal{L}}$ (Bertsekas, 1999; Khan et al., 2017b), and alternatively express the updates of Eq. (4.8) and (4.9) as,

$$\mathbf{p}^D{}_{t+1} = (1-\alpha_t)\,\mathbf{p}^D{}_t + \alpha_t \mathbb{E}_{q(\boldsymbol{\theta})}\left[\hat{\nabla}_{\boldsymbol{\theta}}\tilde{\mathcal{L}} \circ \hat{\nabla}_{\boldsymbol{\theta}}\tilde{\mathcal{L}}\right] \tag{4.12}$$

$$\begin{aligned}\boldsymbol{\mu}^D{}_{t+1} = \boldsymbol{\mu}^D{}_t &- \alpha_t(\mathbf{p}^D{}_{t+1} + \lambda_1\mathbf{1})^{-1} \circ \hat{\nabla}_{\boldsymbol{\mu}^D}\tilde{\mathcal{F}} \\ &+ \gamma_t(\mathbf{p}^D{}_t + \lambda_1\mathbf{1}) \circ (\mathbf{p}^D{}_{t+1} + \lambda_1\mathbf{1})^{-1} \circ \left(\boldsymbol{\mu}^D{}_t - \boldsymbol{\mu}^D{}_{t-1}\right),\end{aligned} \tag{4.13}$$

where $\hat{\nabla}_{\boldsymbol{\mu}^D}\tilde{\mathcal{F}} = (\mathbb{E}_{q(\boldsymbol{\theta})}\left[\hat{\nabla}_{\boldsymbol{\theta}}\tilde{\mathcal{L}}\right] + \lambda_1\boldsymbol{\mu}^D{}_t)$ and $\mathbf{p}^D{}_t := \boldsymbol{\sigma}^D{}_t^{-2} - \lambda_1\mathbf{1}$, where $\mathbf{1}$ is a vector of ones. It is worth mentioning that here, the computational complexity is reduced, from $\mathcal{O}((QMP + QP + QJ)^3)$ to $\mathcal{O}(QMP + QP + QJ)$, as in the CCGP model, but the total number of LPFs is influenced by all D heterogeneous outputs; i.e., for the HetMOGP, $J = \sum_{d=1}^{D} J_d$, where J_d accounts for the number of latent functions necessary to parametrise the d-th likelihood as per section 4.1. See Appendix G for a pseudo-code implementation of the algorithm.

4.6 Predictive Distribution

In order to make predictions with the HetMOGP model with LMC, it is necessary to compute the following distribution: $p(\mathbf{y}_*|\mathbf{y}) \approx \int p(\mathbf{y}_*|\mathbf{f}_*)q(\mathbf{f}_*)d\mathbf{f}_*$, where $q(\mathbf{f}_*) = \prod_{d=1}^{D}\prod_{j=1}^{J_d} q(\mathbf{f}_{d,j,*})$. Given that we have introduced a variational distribution $q(\boldsymbol{\theta})$ over all hyper-parameters and inducing points of the model, we could apply a fully Bayesian treatment when calculating $q(\mathbf{f}_{d,j,*})$, i.e., $q(\mathbf{f}_{d,j,*}) = \int p(\mathbf{f}_{d,j,*}|\mathbf{u},\boldsymbol{\theta})q(\mathbf{u})q(\boldsymbol{\theta})d\boldsymbol{\theta}d\mathbf{u}$ (Rossi et al., 2021). In practice, we found that $q(\boldsymbol{\theta})$'s covariance converged to very small values, in general $\text{diag}(\boldsymbol{\sigma}^{D\,2}) \leq 10^{-15}$, and almost all the uncertainty information was concentrated on $q(\mathbf{u})$'s covariance. Since making predictions with the equations above becomes computationally expensive and most of the uncertainty is represented by the distribution $q(\mathbf{u})$, we can trade-off the computation by using the MAP solution for $q(\boldsymbol{\theta})$ and completely integrating over the remaining distribution as follows: $q(\mathbf{f}_{d,j,*}) =$

$\int p(\mathbf{f}_{d,j,*}|\mathbf{u}, \boldsymbol{\theta} = \boldsymbol{\mu}^D)q(\mathbf{u})d\mathbf{u}$. When solving these integrals we arrive to exactly the same solutions in Eq. (4.6), where we simply have to evaluate the matrix covariances $\mathbf{K}_{\mathbf{f}_{d,j,*}\mathbf{u}}$ and $\mathbf{K}_{\mathbf{f}_{d,j,*}\mathbf{f}_{d,j,*}}$, all at the new inputs $\mathbf{X}_*$.

4.7 Experiments

In this section, we explore the performance of the proposed FNG method for jointly optimising all variational parameters, hyper-parameters and inducing points. We also test the hybrid (HYB) method proposed by Salimbeni et al. (2018), and compare the performance against Adam and SGD methods. We run experiments on different toy and real datasets, for all datasets we use a splitting of 75% and 25% for training and testing, respectively. The experiments consist on evaluating the method's performance when starting with 20 different initialisations of $q(\boldsymbol{\theta})$'s parameters to be optimised. We report the NELBO shown in Eq. (4.5) over the training set, and the NLPD error for the test set; this error metric takes into account the predictions' uncertainty (Quiñonero-Candela et al., 2006).

4.7.1 Optimising the HetMOGP with LMC on Toy Data

We are interested in looking at the performance of HetMOGP with LMC when increasing the number of outputs, which implies rising also the heterogeneity of the output data. Given that the inducing points have the same input space dimensionality and strongly affect the performance of sparse MOGPs, we are also interested in assessing the behaviour when increasing the input space dimensionality. For all the toy data examples we define an input space $\mathbf{X} \in [0, 1]^{N \times P}$ with $N = 2 \times 10^3$ observations, we analyse a set of different dimensions $P = \{1, 2, 3, 4, 5, 10\}$. We assume a number of $Q = 3$ with an EQ kernel $k_q(\cdot, \cdot)$, and the inducing points $\mathbf{Z}_q \in \mathbb{R}^{M \times P}$, with $M = 80$. We run the experiments using mini-batches of 50 samples at each iteration, and we use one sample to approximate the expectations w.r.t $q(\boldsymbol{\theta})$ in Eq. (4.7). Below we describe the characteristics of each toy dataset.

Toy Data 1 (T1): the first toy example consists of three outputs $D = 3$; the first output is $y_1 \in \mathbb{R}$, the second $y_2 \in [0, 1]$ and the third $y_3 \in \{0, 1\}$. We use a Heteroscedastic-Gaussian (HetGaussian), a Beta and Bernoulli distribution as the likelihoods for each output, respectively.

Toy Data 2 (T2): the second toy example consists of five outputs $D = 5$, where

the first three are exactly the same ones as T1 with the same likelihoods and the two additional ones are $y_4 \in [0, \infty]$, and $y_5 \in [0, \infty]$. We use a Gamma and an Exponential distribution for those latter outputs, respectively.

Toy Data 3 (T3): the third toy example consists of ten outputs $D = 10$, where the data type of the first five outputs $\{y_d\}_{d=1}^{5}$ is exactly the same as T2. Also, the last five outputs $\{y_d\}_{d=6}^{10}$ share the same data type of the outputs in T2. We use the following ten likelihoods: HetGaussian, Beta, Bernoulli, Gamma, Exponential, Gaussian (with $\sigma_{\text{lik}} = 0.1$), Beta, Bernoulli, Gamma and Exponential. The data of the first five outputs is not the same as the last ones since the distributions of the generative model depend on the LCCs $w_{d,j,q}$ that generate the LPFs in Eq. (4.2). [2]

In order to visualise the convergence performance of the methods, we show results for T2 which consists of five outputs, where all of them are used in T3 and three of them in T1. We focus on the example for which $P = 10$ as the dimensionality. Figure 4.1 shows the behaviour of the different algorithms over T2, where its top left sub-figure shows the average convergence of the NELBO after running 20 different initialisations. The figure shows that our FNG method tends to find a better local optima solution that minimises the NELBO followed by the HYB, Adam and SGD. The other sub-figures titled from Out1 to Out5 show the model's average NLPD achieved by each of the methods over the test set. From Figure 4.1 we can notice that the SGD method does not progress much through the inference process achieving the poorest performance along the diverse outputs. The Adam method presents a big variance along the different outputs, showing its ability to explore feasible solutions, but arriving at many different poor local minima. Particularly, for the output 3, a Bernoulli likelihood, the method hardly moves from its initial NLPD value, showing in the figure a tiny variance without much improvement. This means the method lacks exploration and rapidly becomes trapped in a very poor local minima. The HYB method in general shows smaller error bars than Adam and SGD. Indeed, it reaches low NLPD results for Gamma, HetGaussian and Exponential likelihoods, with similar behaviour to our FNG method in the two latter distributions. Although, it is difficult for HYB to achieve a proper NLPD performance on the distributions Beta and Bernoulli; though for the Beta distribution presents boxes with big variance meaning that it arrives to many different solutions, the NLPD's mean shows a trending to weak solutions. For the Bernoulli, it is deficient exploring, so it also ends up in poor solutions. Our FNG

[2] The code with all toy configurations is publicly available in the repository: https://github.com/juanjogg1987/Fully_Natural_Gradient_HetMOGP

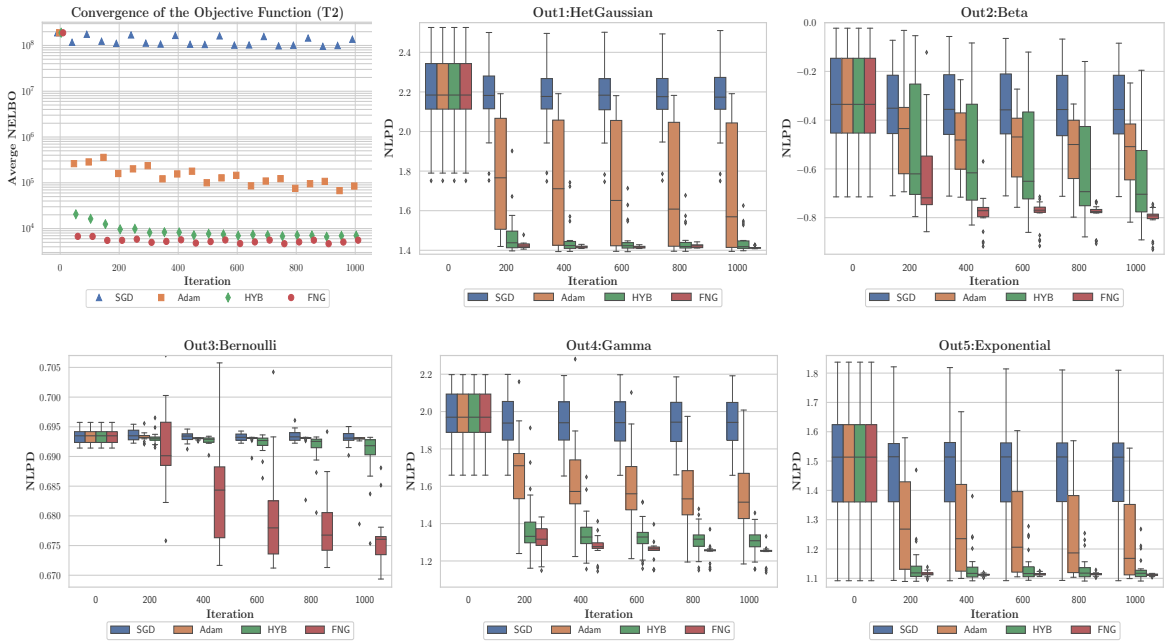


Figure 4.1: *Performance of the different inference methods on the T2 dataset for* $P = 10$ *using 20 different initialisations. The top left sub-figure shows the average NELBO convergence. The other sub-figures show the box-plot trending of the NLPD over the test set for each output. The box-plots at each iteration follow the legend's order from left to right: SGD, Adam, HYB and FNG. The isolated diamonds that appear in the outputs' graphs represent "outliers".*

method is consistent along the diverse outputs, usually tending to richer local minima solutions than the other methods. For the Beta and Gamma outputs, FNG makes a

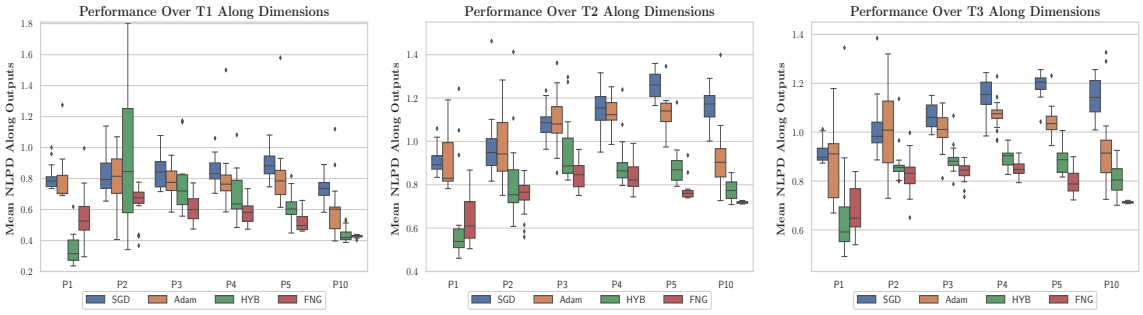


Figure 4.2: *Trending of the Mean NLPD along outputs for 20 different initialisations. Performance over: T1 (left), T2 (middle) and T3. Each sub-figure summarises the Mean NLPD of SGD, Adam, HYB and FNG methods along dimensions* $P = \{1, 2, 3, 4, 5, 10\}$. *The box-plots at each* P *follow the legend's order.*

confident progress and even shows some "outliers" below its boxes which means that our method has the ability to eventually provide better solutions than the other methods. For the Bernoulli distribution, Figure 4.1 shows that FNG presents big variance boxes, but with a tendency to much better solutions than the other methods. This big variance effect let us confirm that our proposed method actually takes advantage of the stochastic exploration induced over the model hyper-parameters for avoiding poor local minima solutions.

Figure 4.2 summarises the behaviour along the different dimensions P for each toy example. We notice from Figure 4.2 that our FNG method achieves better test performance along distinct dimensions for all toy examples, followed by the HYB, Adam and SGD methods, though HYB presents better results than FNG when $P = 1$. All methods in general tend to present large variances for T1 which consists of three outputs, although this effect is reduced when the number of outputs is increased. Our FNG in general presents the smallest variance showing its ability to find better local minima even with many outputs. When increasing the dimensionality, the methods tend to degrade their performance, but the less sensitive to such behaviour are the HYB and FNG methods, where the latter, in general achieves the lowest mean NLPD along outputs for the different toy examples. Apart from the heterogeneous toy data examples shown in this work, we also ran experiments for dimensions higher than $P = 10$, although we noticed that all methods behaved similar except for the SGD

which demands a very small step-size parameter that makes it progress slowly. We believe that the toy examples become difficult to control in such dimensions and the data observations become broadly scattered. We also explored experiments increasing the mini-batch size at each iteration, we noticed the gradient's stochasticity is reduced helping to increase the convergence rates of all methods, but the ones using NG perform better. When reducing the mini-batch size, our FNG method usually performs better than the others probably due to the fact that it additionally exploits the probability distribution $q(\boldsymbol{\theta})$, imposed over the hyper-parameters and inducing points.

4.7.2 Settings for Real Datasets Experiments

In this subsection we describe the different real datasets used for our experiments (See Appendix J for information about the web-pages where we took the datasets from).

HUMAN Dataset: the human behaviour dataset (HUMAN, $N_1, N_2 = 5 \times 10^3, N_3 = 21 \times 10^3, P = 1$, N_d associates the number of observations per output) contains information for monitoring psychiatric patients with a smartphone *app*. It consists of three outputs; the first monitors use/non-use of *WhatsApp*, $y_1 \in \{0, 1\}$, the second represents distance from the patient's home location, $y_2 \in \mathbb{R}$, and the third accounts for the number of smartphone active *apps*, we rescale it to $y_3 \in [0, 1]$. We use a Bernoulli, HetGaussian and a Beta distribution as the likelihoods for each output, respectively. We assume $Q = 5$ latent functions.

LONDON Dataset: the London dataset (LONDON, $N = 20 \times 10^3, P = 2$) is a register of properties sold in London in 2017; it consists of two outputs; the first represents house prices with $y_1 \in \mathbb{R}$ and the second accounts for the type of house. We use two types (flat/non-flat) with $y_2 \in \{0, 1\}$. We use a HetGaussian and Bernoulli distribution as the likelihood for each output respectively. We assume $Q = 3$ latent functions.

NAVAL Dataset: the naval dataset (NAVAL, $N = 11 \times 10^3, P = 15$) contains information of condition based maintenance of naval propulsion plants. It consists of two outputs: plant's compressor decay state coefficient and turbine decay state coefficient. We re-scaled both as $y_1, y_2 \in [0, 1]$, and used a Beta and Gamma distribution as the likelihood for each output respectively. We assume $Q = 4$ functions.

SARCOS Dataset: a seven degrees-of-freedom SARCOS anthropomorphic robot arm data, where the task is to map from a 21-dimensional input space (7 joint positions, 7 joint velocities, 7 joint accelerations) to the corresponding 7 joint output torques

(SARCOS, $N = 44.5 \times 10^3, P = 21, D = 7$). We use a HetGaussian distribution as the likelihood for each output and assume $Q = 3$ functions.

MOCAP7 Dataset: a motion capture data for a walking subject (MOCAP7, $N = 744, P = 1, D = 40$). We use a HetGaussian distribution as the likelihood for each output and assume $Q = 3$ functions. We refer to the dataset as MOCAP7 due to the selection of the 7-th subject for the walking experiment.

For the first three datasets, the number of inducing points per latent function is $M = 80$ and for each function $u_q(\cdot)$ we use an EQ kernel like Eq. (4.4). We run the experiments using mini-batches of 50 samples at each iteration, and we use one sample to approximate the expectations with regard to $q(\boldsymbol{\theta})$ in Eq. (4.7). For SARCOS we use mini-batches of 200 due to its large number of observations, and given that MOCAP7 is not a large dataset we use mini-batches of 5 with $M = 20$. To select Q, we applied a rule of thumb as follows: **I.** If $D <= 5$ set $Q = J$. We opted for this rule of thumb as a way to allow the HetMOGP model to have a high flexibility for modelling the data in presence of few outputs. **II.** if $D > 5$ set $Q = 3$. We chose this option for not overloading the computational complexity in presence of many outputs, though by setting $Q = 3$ we still can at least model low, medium and high length-scale resolutions from a dataset (see Appendix I for details about the rule of thumb for setting Q and the number J_d associated to each likelihood distribution).

4.7.3 Optimising the HetMOGP with LMC on Real Data

For this sub-section we explore our method's behaviour over the HetMOGP with LMC on HUMAN, LONDON, NAVAL, SARCOS and MOCAP7.

Figures 4.3 and 4.4 show the NELBO convergence over the training set, together with the average NLPD performance over the test set for HUMAN, LONDON and NAVAL data, respectively. We provide a merged NLPD along outputs for LONDON and NAVAL (see Appendix J for an analysis of each specific output). With regard to the convergence rate of the NELBO for HUMAN and LONDON datasets all methods converge similarly. Nonetheless, for the NAVAL dataset, our FNG approach presents a faster converge, followed by HYB and Adam; SGD remains without much progress along the iterations.

For the HUMAN dataset, the SGD arrives at a better minimum than Adam, but the Adam's averaged NLPD is higher across outputs. HYB reaches consistent solutions being better than Adam and SGD, not only in the training process but also in

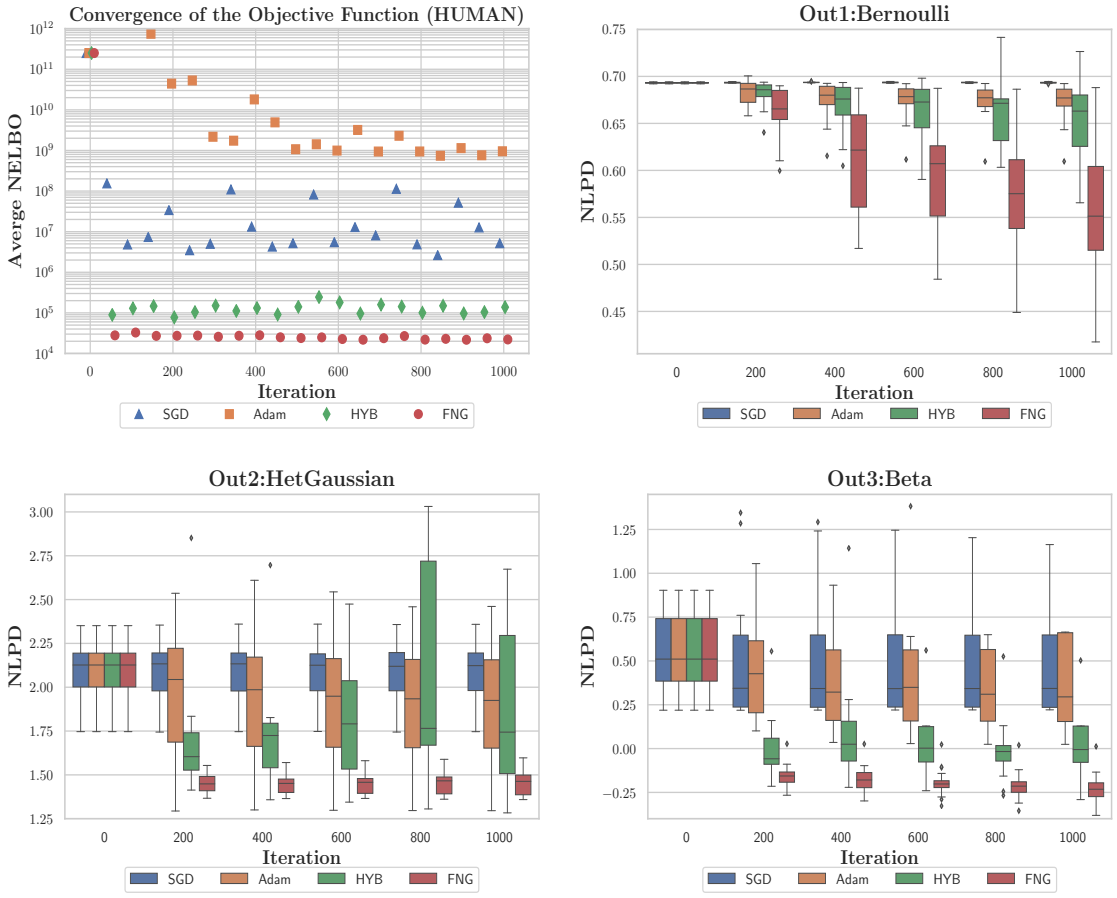


Figure 4.3: *Performance of the diverse inference methods on the HUMAN dataset using 20 different initialisations. The left sub-figure shows the average NELBO convergence of each method. The other sub-figures show the box-plot trending of the NLPD over the test set for each output. The box-plots at each iteration follow the legend's order from left to right: SGD, Adam, HYB and FNG. The isolated diamonds that appear in the outputs' graphs represent "outliers".*

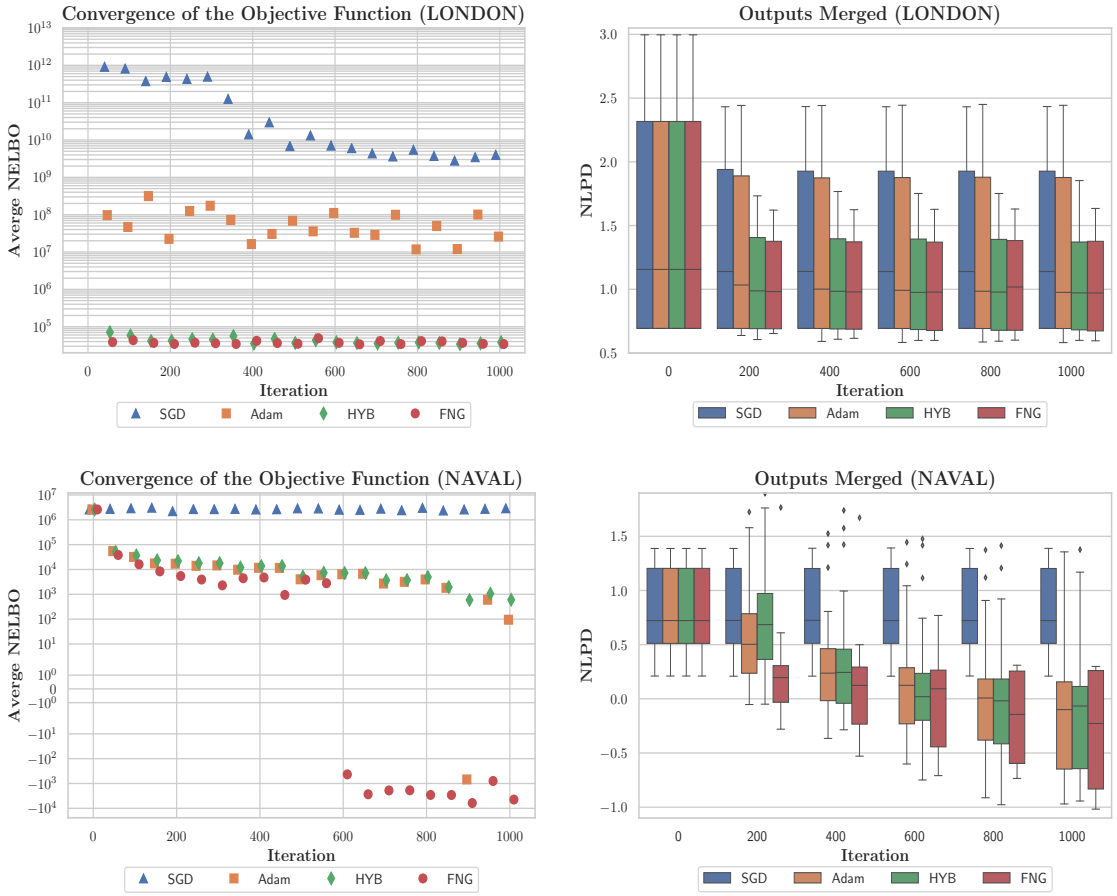


Figure 4.4: *Performance of the diverse inference methods on the LONDON and NAVAL datasets using 20 different initialisations. Sub-figures top-left and top-right correspond to LONDON; bottom-left and bottom-right refer to NAVAL. For each dataset we show the average NELBO convergence of each method and the box-plot trending of the NLPD over the test set across all output. The box-plots at each iteration follow the legend's order from left to right: SGD, Adam, HYB and FNG. The isolated diamonds that appear in the outputs' graphs represent "outliers".*

testing along the HetGaussian and Beta outputs. Though, the Bernoulli output limits the overall performance of the method since there is not much improvement along the iterations. Our FNG method also shows a steady performance along outputs, commonly arriving to solutions with lower NLPD than the other methods. Our method presents the biggest variance for the Bernoulli output, implying strong exploration of the solutions' space for such likelihood, allowing it to reach the lowest average NLPD. For the LONDON dataset, Adam converges to a richer minimum of the NELBO than SGD. Moreover, the NLPD for Adam is, on average, better than the SGD. The HYB and FNG arrive to a very similar value of the NELBO, both being better than Adam and SGD. HYB and FNG methods attain akin NLPD metrics, but the average and median trend of our approach is slightly better, being more robust to the initialisation than HYB method. The NLPD performance for the NAVAL dataset shows in Figure 4.4 that the SGD method cannot make progress. We tried to set a bigger step-size, but usually increasing it derived in numerical problems due to ill-conditioning of the covariance matrices. The methods Adam and HYB show similar NLPD boxes, but at the end, Adam attains a slightly lower median with bigger variance than HYB. Regarding the NLPD, our FNG method ends up with a larger variance than SGD, Adam and HYB, but obtaining a much better mean and median trending than the others. Also, our FNG shows that the upper bar of the NLPD box is very close to the interquartile range, while the other methods present larger upper bars, this means that our FNG method concentrates in regions that provide better predictive performance than the other methods.

Figure 4.5 shows the performance achieved by the different optimisation methods for SARCOS and MOCAP7 datasets. Since these datasets present a high number of outputs we stacked the NLPD metric along all outputs. We can notice from the SARCOS experiment, in the first two sub-figures to the left, that SGD cannot improve much during the inference process both for NELBO and NLPD. Adam and HYB converge to the same local minima achieving the same average NELBO and NLPD trend, in contrast to our FNG method which attains the lowest values showing a better performance. Particularly in the SARCOS experiment, figures show how our method changes suddenly, around iteration 600, probably escaping from the same local minima to which Adam and HYB converged. For the MOCAP7's experiment, the two sub-figures to the right show that SGD slightly improves its performance in the inference process, while Adam reaches a much better minimum for the average NELBO. Although, these former methods do not perform better than HYB and FNG. The HYB and FNG behave similar before 500 iteration, but in the long term our FNG presents the lowest average

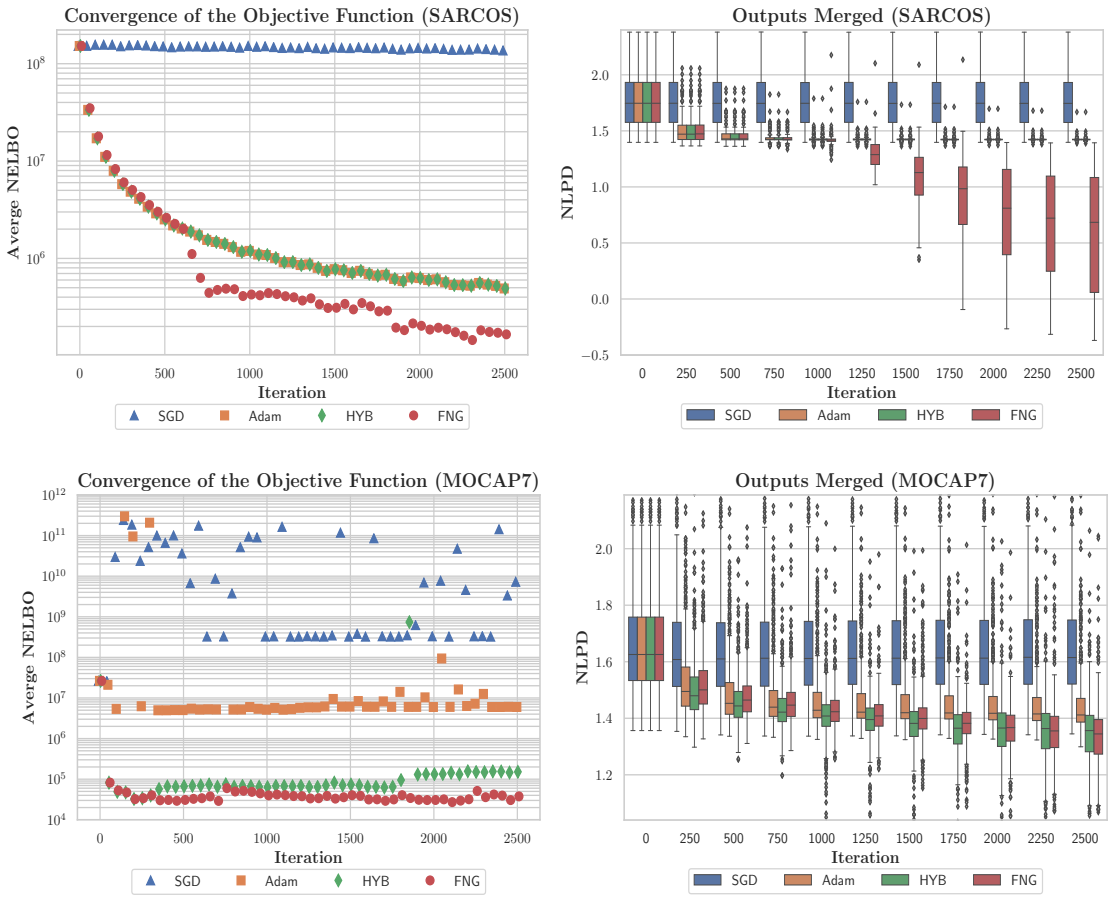


Figure 4.5: *Performance of the diverse inference methods on the SARCOS and MOCAP7 datasets using 20 different initialisations for HetMOGP with LMC. Sub-figures left and middle-left correspond to SARCOS; middle-right and right refer to MOCAP7. For each dataset we show the average NELBO convergence of each method and the box-plot trending of the NLPD over the test set across all output. The box-plots at each iteration follow the legend's order from left to right: SGD, Adam, HYB and FNG. The isolated diamonds that appear in the outputs' graphs represent "outliers".*

NELBO. Likewise, the NLPD shows that HYB presents a slightly better trend than FNG at the early stages of the inference, but at the end, our FNG finds a better NLPD metric.

4.7.4 Discussion

In practice we noticed that some likelihoods (e.g. HetGaussian, Gamma) tend to strongly influence the value of the objective function, so the optimisers HYB, Adam and SGD are prone to find solutions that focus on such kind of likelihoods, while neglecting the others with less influence, for instance a Bernoulli or Beta as shown in Figure 4.1. On the other hand, our proposed scheme presents a more consistent performance achieving richer solutions across the different types of outputs' distributions. When increasing the outputs' size our FNG presented a consistent performance for TOY and real datasets like SARCOS and MOCAP7. We realised that HYB method presents a relevant performance for low input dimensionalities in comparison to SGD and Adam, but when the input dimensionality increases its performance degrades as shown for the TOY experiments when $P > 1$ and for the SARCOS experiment with $P = 21$. So, our method is the least sensitive to reduce its performance when increasing the input dimensionality, followed by the HYB and Adam methods. When using the SGD method we had to set a very small step-size parameter, because using large step-sizes makes the model to easily become ill-conditioned.

The VO bound in Eq. (4.7) can be seen as a fully Bayesian treatment of the HetMOGP, where the model's parameters and hyper-parameters follow a prior distribution, where the positive constraint variables follow a Log-Normal distribution and the non-constraint ones follow a Gaussian distribution. Our VO bound benefits from the assumption of a Gaussian exploratory (or posterior) distribution for deriving in a closed-form our FNG optimisation scheme. This scheme helps to find solutions that directly improve the predictive capabilities of the HetMOGP model. For instance, since the inducing points' dimensionality is directly influenced by the input dimensionality, we believe that applying exploration over them helps to improve the model performance for high input dimensionalities as shown in the experiments. Although, a very high dimensionality of the input space directly influences the complexity of our scheme. It might be worth exploring alternatives to scale the algorithms for allowing the use of the full covariance matrix, $\mathbf{\Sigma}^D$, with the aim to exploit full correlations between all model's hyper-parameters, and possibly improve its predictive capabilities.

4.8 Summary

In this chapter, we have shown how a fully natural gradient scheme improves optimisation of a heterogeneous MOGP model with LMC by generally reaching better local optima solutions with higher test performance rates than HYB, Adam and SGD methods. We have shown that our FNG scheme provides rich local optima solutions, even when increasing the dimensionality of the input and/or output space. In the next chapter, we will provide an extension of the HetMOGP based on a convolution processes model, rather than on the LMC approach. This HetMOGP with convolution processes is a novel model extension since there are no former MOGP models with convolution processes that involve stochastic variational inference, nor a model of heterogeneous outputs that relies on convolution processes.

Chapter 5

Heterogeneous Multi-Output GPs Model with Convolution Processes

In chapter 3, we introduced the CCGP model based on an LMC for generating the LPFs. Then, in chapter 4, we described how the CCGP model becomes a HetMOGP model in the context of multiple heterogeneous outputs, and derived a FNG scheme for improving inference over such a model. In this chapter, we provide an extension of the HetMOGP based on a Convolution Processes model (Boyle and Frean, 2005; Álvarez and Lawrence, 2011), rather than on the LMC approach for generating the LPFs. This is a novel contribution since there are no former MOGP models with convolution processes that involve stochastic variational inference, nor a model of heterogeneous outputs that relies on convolution processes.

The CPM relies on solving convolution integrals between smoothing kernels and GP priors. The construction of the HetMOGP with CPM includes multiple smoothing kernels that involve an additional set of hyper-parameters. Thus, in comparison to the model in chapter 4 based on an LMC, here, the additional smoothing kernels' hyper-parameters make this extension of HetMOGP with CPM even more prone to suffering from a strong conditioning; i.e., a strong conditioning between the model's variational parameters, the smoothing kernels' hyper-parameters, the kernel hyper-parameters of the GP priors and the inducing points. Therefore, we also provide a derivation of the fully natural gradient scheme for optimising this new model extension.

5.1 The Convolution Processes Model

In the previous chapter, the generative process (or LMC) for correlating the LPFs relied on an static mixture of independent Gaussian Processes, i.e., a particular output function $f_{d,j}(\mathbf{x})$ simply depends on the values of the GP latent functions $\{u_q(\mathbf{x})\}_{q=1}^{Q}$ specifically evaluated at the data observation $\mathbf{x}$ (Álvarez and Lawrence, 2011). In contrast, in this section we describe the convolution processes model, a more powerful way to correlate the LPFs that can be seen as a dynamic version of the LMC (Álvarez, 2011); i.e., a particular output function $f_{d,j}(\mathbf{x})$ depends on all the continuous values of the GP latent functions, $\{u_q(\cdot)\}_{q=1}^{Q}$, as follows:

$$f_{d,j}(\mathbf{x}) = \sum_{q=1}^{Q}\sum_{i=1}^{R_q}\int_{\mathcal{X}} G_{d,j,q}^{i}(\mathbf{x}-\mathbf{r}')\,u_q^{i}(\mathbf{r}')\,\mathrm{d}\mathbf{r}',$$

where $u_q^i(\mathbf{x})$ are IID samples from Gaussian Processes $u_q(\cdot) \sim \mathcal{GP}(0, k_q(\cdot,\cdot))$ and each $G_{d,j,q}(\cdot)$ represents a smoothing kernel (Boyle and Frean, 2005). Through the use of smoothing kernels, we might, for instance, model two LPFs where one of them represents a blurring version of the other; something not possible to achieve in the static mixture of GPs performed by an LMC (Higdon, 2002). An important fact of using convolution processes is that by convolving the latent Gaussian processes $\{u_q^i(\cdot)\}_{q=1,i=1}^{Q,Rq}$ with a smoothing kernel, it allows us to obtain also a Gaussian process. By using the CPM we could model correlations between LPFs that present different length-scales, this is due to having possible convolution operations where the latent GPs, $u_q^i(\cdot)$, could be smoothed for a particular $f_{d,j}(\mathbf{x})$, but not necessarily for other (Álvarez, 2011).

We can notice from the equation above that the computation of the LPFs, $f_{d,j}(\mathbf{x})$, depends on all the possible values of $\{u_q^i(\cdot)\}_{q=1,i=1}^{Q,Rq}$ evaluated at all possible inputs $\mathbf{r}'$, that is the why we refer as a dynamic version of the LMC. In this chapter we aim to model the LPFs of the HetMOGP using convolution processes. Therefore, the HetMOGP model with convolution processes follows the same likelihood defined in Eq. (4.1), but each $f_{d,j}(\mathbf{x}_n)$ is considered a LPF that comes from a convolution processes model. We will use $R_q = 1$ as in the LMC for simplicity in the following derivations.

5.2 The Inducing Points Method

With the purpose to reduce the computational complexities involved in GPs we follow the inducing variables framework by augmenting the probability space as,

$$p(\mathbf{f}|\check{\mathbf{u}})p(\check{\mathbf{u}}) = \prod_{d=1}^{D}\prod_{j=1}^{J_d} p(\mathbf{f}_{d,j}|\check{\mathbf{u}}_{d,j})p(\check{\mathbf{u}}_{d,j}), \tag{5.1}$$

with $p(\check{\mathbf{u}}) = \prod_{d=1}^{D}\prod_{j=1}^{J_d} p(\check{\mathbf{u}}_{d,j})$, and $p(\mathbf{f}|\check{\mathbf{u}}) = \prod_{d=1}^{D}\prod_{j=1}^{J_d} p(\mathbf{f}_{d,j}|\check{\mathbf{u}}_{d,j})$, where the vector $\check{\mathbf{u}} = [\check{\mathbf{u}}_{1,1}^\top, ..., \check{\mathbf{u}}_{1,J_1}^\top, ..., \check{\mathbf{u}}_{D,J_D}^\top]^\top \in \mathbb{R}^{JM\times 1}$ is built from the inducing variables $\check{\mathbf{u}}_{d,j} = [f_{d,j}(\mathbf{z}_{d,j}^{(1)}), ..., f_{d,j}(\mathbf{z}_{d,j}^{(M)})]^\top \in \mathbb{R}^{M\times 1}$. As it can be seen, these inducing variables are additional evaluations of the functions $f_{d,j}(\cdot)$ at each set of inducing points $\mathbf{Z}_{d,j} = [\mathbf{z}_{d,j}^{(1)}, ..., \mathbf{z}_{d,j}^{(M)}]^\top \in \mathbb{R}^{M\times P}$, thus the set of all inducing points can be represented as $\mathbf{z} = [\text{vec}(\mathbf{Z}_{1,1})^\top, ..., \text{vec}(\mathbf{Z}_{1,J_1})^\top, ..., \text{vec}(\mathbf{Z}_{D,J_D})^\top]^\top \in \mathbb{R}^{JMP\times 1}$. Using the properties of Gaussian distributions, we can express, $p(\mathbf{f}_{d,j}|\check{\mathbf{u}}_{d,j}) = \mathcal{N}(\mathbf{f}_{d,j}|\mathbf{A}_{\mathbf{f}_{d,j}\check{\mathbf{u}}_{d,j}}\check{\mathbf{u}}_{d,j}, \bar{\mathbf{Q}}_{\mathbf{f}_{d,j}})$, $p(\check{\mathbf{u}}_{d,j}) = \mathcal{N}(\check{\mathbf{u}}_{d,j}|\mathbf{0}, \mathbf{K}_{\check{\mathbf{u}}_{d,j}})$, with the following definitions: $\mathbf{A}_{\mathbf{f}_{d,j}\check{\mathbf{u}}_{d,j}} = \mathbf{K}_{\mathbf{f}_{d,j}\check{\mathbf{u}}_{d,j}}\mathbf{K}_{\check{\mathbf{u}}_{d,j}}^{-1}$, $\bar{\mathbf{Q}}_{\mathbf{f}_{d,j}} = \mathbf{K}_{\mathbf{f}_{d,j}\mathbf{f}_{d,j}} - \check{\mathbf{Q}}_{\mathbf{f}_{d,j}}$, $\check{\mathbf{Q}}_{\mathbf{f}_{d,j}} = \mathbf{K}_{\mathbf{f}_{d,j}\check{\mathbf{u}}_{d,j}}\mathbf{K}_{\check{\mathbf{u}}_{d,j}}^{-1}\mathbf{K}_{\check{\mathbf{u}}_{d,j}\mathbf{f}_{d,j}}$, $\mathbf{K}_{\mathbf{f}_{d,j}\check{\mathbf{u}}_{d,j}} = \mathbf{K}_{\check{\mathbf{u}}_{d,j}\mathbf{f}_{d,j}}^\top$. Here the covariance matrix $\mathbf{K}_{\mathbf{f}_{d,j}\mathbf{f}_{d,j}} \in \mathbb{R}^{N\times N}$ is built from the evaluation of all pairs of input data $\mathbf{X} \in \mathbb{R}^{N\times P}$ in the covariance function,

$$\text{cov}\left[f_{d,j}\left(\mathbf{x}\right) f_{d',j'}\left(\mathbf{x}'\right)\right] = \sum_{q=1}^{Q}\int_{\mathcal{X}} G_{d,j,q}\left(\mathbf{x}-\mathbf{r}\right)\int_{\mathcal{X}} G_{d',j',q}\left(\mathbf{x}'-\mathbf{r}'\right) k_q(\mathbf{r},\mathbf{r}')\text{d}\mathbf{r}\text{d}\mathbf{r}',$$

where the cross covariance matrix $\mathbf{K}_{\mathbf{f}_{d,j}\check{\mathbf{u}}_{d,j}} \in \mathbb{R}^{N\times M}$ is formed by evaluations of the equation above between inputs $\mathbf{X}$ and $\mathbf{Z}_{d,j}$, and the matrix $\mathbf{K}_{\check{\mathbf{u}}_{d,j}} \in \mathbb{R}^{M\times M}$ is also built from evaluations of the equation above between all pairs of inducing points $\mathbf{Z}_{d,j}$ respectively. We can compute the above covariance function analytically for certain forms of $G_{d,j,q}(\cdot)$ and $k_q(\mathbf{r},\mathbf{r}')$. In this thesis, we follow the work by Álvarez and Lawrence (2011) by defining the kernels in the EQ form of Eq. (4.4): $k_q\left(\mathbf{x},\mathbf{x}'\right) = \mathcal{E}(\boldsymbol{\tau}|\mathbf{0},\mathbf{L}_q)$ and $G_{d,j,q}(\boldsymbol{\tau}) = S_{d,j,q}\mathcal{E}(\boldsymbol{\tau}|\mathbf{0},\boldsymbol{\kappa}_{d,j})$, where $S_{d,j,q}$ is a weight associated to the LPF indexed by $f_{d,j}(\cdot)$ and to the latent function $u_q(\cdot)$, and $\boldsymbol{\kappa}_{d,j}$ is a diagonal covariance matrix particularly associated to each $f_{d,j}(\cdot)$; $\boldsymbol{\kappa}_{d,j}$ can be seen as a matrix of length-scales in its diagonal. Therefore, when solving for the $\text{cov}\left[f_{d,j}\left(\mathbf{x}\right) f_{d',j'}\left(\mathbf{x}'\right)\right]$ above we end up with the closed-form,

$$k_{f_{d,j},f_{d',j'}}\left(\boldsymbol{\tau}\right) = \sum_{q=1}^{Q} S_{d,j,q}S_{d',j',q}\mathcal{E}(\boldsymbol{\tau}|\mathbf{0},\mathbf{P}_{d,j,d',j',q}), \tag{5.2}$$

where $\mathbf{P}_{d,j,d',j',q}$ represents a diagonal matrix of length-scales, $\mathbf{P}_{d,j,d',j',q} = \boldsymbol{\kappa}_{d,j} + \boldsymbol{\kappa}_{d',j'} + \mathbf{L}_q$.

5.3 The Evidence Lower Bound

We now introduce the negative ELBO for the HetMOGP that uses convolution processes. It follows as

$$\tilde{\mathcal{L}} = \sum_{n=1}^{N}\sum_{d=1}^{D}\mathbb{E}_{q(\mathbf{f}_{d,1})\cdots q(\mathbf{f}_{d,J_d})}\left[g_{d,n}\right] + \sum_{d=1}^{D}\sum_{j=1}^{J_d}\mathbb{D}_{KL}\left(q(\check{\mathbf{u}}_{d,j})\|p(\check{\mathbf{u}}_{d,j})\right), \tag{5.3}$$

where $g_{d,n} = -\log p(y_{d,n}|\psi_{d,1}(\mathbf{x}_n), ..., \psi_{d,J_d}(\mathbf{x}_n))$ is the NLL function associated to each output, and we have set a tractable posterior $q(\mathbf{f}, \check{\mathbf{u}}) = p(\mathbf{f}|\check{\mathbf{u}})q(\check{\mathbf{u}})$, where $p(\mathbf{f}|\check{\mathbf{u}})$ is already defined in Eq. (5.1), $q(\check{\mathbf{u}}|\mathbf{m}, \mathbf{V}) = \prod_{d=1}^{D}\prod_{j=1}^{J_d} q(\check{\mathbf{u}}_{d,j})$, and each $q(\check{\mathbf{u}}_{d,j}) = \mathcal{N}(\check{\mathbf{u}}_{d,j}|\mathbf{m}_{d,j}, \mathbf{V}_{d,j})$ is a Gaussian distribution with mean $\mathbf{m}_{d,j} \in \mathbb{R}^{M\times 1}$ and covariance $\mathbf{V}_{d,j} \in \mathbb{R}^{M\times M}$ (see Appendix E for details on the ELBO derivation). The above expectation is computed w.r.t the marginals,

$$q(\mathbf{f}_{d,j}) = \int p(\mathbf{f}_{d,j}|\check{\mathbf{u}}_{d,j})q(\check{\mathbf{u}}_{d,j})d\check{\mathbf{u}}_{d,j} = \mathcal{N}(\mathbf{f}_{d,j}|\tilde{\mathbf{m}}_{\mathbf{f}_{d,j}}, \tilde{\mathbf{V}}_{\mathbf{f}_{d,j}}), \tag{5.4}$$

with the following definitions, $\tilde{\mathbf{m}}_{\mathbf{f}_{d,j}} := \mathbf{A}_{\mathbf{f}_{d,j}\check{\mathbf{u}}_{d,j}}\mathbf{m}_{d,j}$, $\tilde{\mathbf{V}}_{\mathbf{f}_{d,j}} := \mathbf{K}_{\mathbf{f}_{d,j}\mathbf{f}_{d,j}} + \mathbf{A}_{\mathbf{f}_{d,j}\check{\mathbf{u}}_{d,j}}(\mathbf{V}_{d,j} - \mathbf{K}_{\check{\mathbf{u}}_{d,j}})\mathbf{A}^{\top}_{\mathbf{f}_{d,j}\check{\mathbf{u}}_{d,j}}$. The objective derived in Eq. (5.3) for the HetMOGP model with convolution processes requires fitting the parameters of each posterior $q(\check{\mathbf{u}}_{d,j})$, the inducing points $\mathbf{z}$, the kernel hyper-parameters $\mathbf{l}_{\text{kern}}$, the smoothing-kernels' length-scales $\boldsymbol{\kappa}_{\text{smooth}} = [\text{diag}(\boldsymbol{\kappa}_{1,1})^{\top}, ..., \text{diag}(\boldsymbol{\kappa}_{1,J_1})^{\top}, ..., \text{diag}(\boldsymbol{\kappa}_{D,J_D})^{\top}]^{\top} \in \mathbb{R}_{+}^{JP\times 1}$ and the vector of weights $\mathbf{s}_q = [S_{1,1,q}, ..., S_{1,J_1,q}, ..., S_{D,J_D,q}]^{\top} \in \mathbb{R}^{J\times 1}$ associated to each smoothing-kernel. In the interest of fitting those variables in a FNG scheme, in the following section we will explain how to apply the VO perspective over Eq. (5.3) so as to introduce stochasticity over $\mathbf{z}$, $\mathbf{l}_{\text{kern}}$, $\boldsymbol{\kappa}_{\text{smooth}}$ and $\mathbf{s}_q$; and through the MDA we will derive closed-form updates for all parameters of the model.

5.4 Deriving a Fully Natural Gradient Scheme

This section describes how to derive the FNG updates for optimising CPM scheme of the HetMOGP model. We first detail how to induce an exploratory distribution over the hyper-parameters and inducing points, then we write down the MDA for the model and derive the update equations. Later on, we get into specific details about the algorithm's implementation.

5.4.1 An Exploratory Distribution for the HetMOGP with CPM

The case of the CPM has the same kernel hyper-parameters $\mathbf{l}_{\text{kern}} = \exp(\boldsymbol{\theta}_{\mathbf{L}})$ and inducing points $\mathbf{z} = \boldsymbol{\theta}_{\mathbf{z}}$, as the LMC in chapter 4, but differs from it since the smoothing kernels involve a new set of hyper-parameters, the smoothing-kernels' length-scales. The way we define and connect the new random real vectors is as follows: $\boldsymbol{\kappa}_{\text{smooth}} = \exp(\boldsymbol{\theta}_{\boldsymbol{\kappa}})$, with $\boldsymbol{\theta}_{\boldsymbol{\kappa}} = [\boldsymbol{\theta}_{\boldsymbol{\kappa}_{1,1}}^{\top}, ..., \boldsymbol{\theta}_{\boldsymbol{\kappa}_{1,J_1}}^{\top}, ..., \boldsymbol{\theta}_{\boldsymbol{\kappa}_{D,J_D}}^{\top}]^{\top} \in \mathbb{R}^{JP\times 1}$, where $\boldsymbol{\theta}_{\boldsymbol{\kappa}_{d,j}} \in \mathbb{R}^{P\times 1}$ is a real random vector associated to each smoothing kernel $G_{d,j,q}(\cdot)$ from Eq. (5.2). Also, instead of the combination coefficients $\mathbf{w}$ of the LMC, for the CPM we have an analogous set of weights from the smoothing-kernels in Eq. (5.2), $\mathbf{s} = \boldsymbol{\theta}_{\mathbf{s}}$, where $\mathbf{s} = [\mathbf{s}_1^{\top}, ..., \mathbf{s}_Q^{\top}]^{\top} \in \mathbb{R}^{QJ\times 1}$ is a vector that groups all the weights that belong to the smoothing kernels. Thus, the real random vectors for the CPM are: $\boldsymbol{\theta}_{\mathbf{z}} \in \mathbb{R}^{JMP\times 1}$, $\boldsymbol{\theta}_{\mathbf{L}} \in \mathbb{R}^{QP\times 1}$, $\boldsymbol{\theta}_{\boldsymbol{\kappa}} \in \mathbb{R}^{JP\times 1}$, and $\boldsymbol{\theta}_{\mathbf{s}} \in \mathbb{R}^{QJ\times 1}$. We group the random vectors by defining $\boldsymbol{\theta} = [\boldsymbol{\theta}_{\mathbf{z}}^{\top}, \boldsymbol{\theta}_{\mathbf{L}}^{\top}, \boldsymbol{\theta}_{\boldsymbol{\kappa}}^{\top}, \boldsymbol{\theta}_{\mathbf{s}}^{\top}]^{\top} \in \mathbb{R}^{(JMP+QP+JP+QJ)\times 1}$. Notice that, for the CPM, the dimensionality of the real random vector $\boldsymbol{\theta}$ differs from the one for LMC, this is due to the way the inducing variables are treated in subsection 5.2 and the additional set of smoothing-kernel's hyper-parameters. In the same way as defined for the LMC, we specify an exploratory distribution $q(\boldsymbol{\theta}) := \mathcal{N}(\boldsymbol{\theta}|\boldsymbol{\mu}^{C}, \boldsymbol{\Sigma}^{C})$ and follow the VO approach in Eq. (2.2) (Staines and Barber, 2013). In this case the objective to bound is the one derived for the CPM, i.e., the new bound, $\tilde{\mathcal{F}}$, is exactly the same as Eq. (4.7), but using the corresponding $\tilde{\mathcal{L}}$ from Eq. (5.3).

5.4.2 Mirror Descent Algorithm for the HetMOGP with CPM

For the HetMOGP with CPM, we follow a similar procedure carried out for the LMC. We use the MDA in Eq. (2.5) and the mean-parameters of distributions $q(\check{\mathbf{u}}_{d,j})$ and $q(\boldsymbol{\theta})$ defining $\boldsymbol{\rho}_{d,j} = \{\mathbf{m}_{d,j}, \mathbf{m}_{d,j}\mathbf{m}_{d,j}^{\top} + \mathbf{V}_{d,j}\}$ and $\boldsymbol{\eta}^{C} = \{\boldsymbol{\mu}^{C}, \boldsymbol{\mu}^{C}\boldsymbol{\mu}^{C\top} + \boldsymbol{\Sigma}^{C}\}$ for minimising Eq. (4.7). Then, our algorithm for the CPM can be written as:

$$\begin{aligned}\boldsymbol{\eta}^{C}{}_{t+1}, \{\boldsymbol{\rho}_{d,j,t+1}\}_{d=1,j=1}^{D,J_d} = \underset{\boldsymbol{\eta}^{C}, \{\boldsymbol{\rho}_{d,j}\}_{d=1,j=1}^{D,J_d}}{\arg\min} & \langle \boldsymbol{\eta}^{C}, \hat{\nabla}_{\boldsymbol{\eta}^{C}} \tilde{\mathcal{F}}_t \rangle + \frac{1}{\tilde{\alpha}_t}\mathrm{KL}(\boldsymbol{\theta})_t - \frac{\tilde{\gamma}_t}{\tilde{\alpha}_t}\mathrm{KL}(\boldsymbol{\theta})_{t-1} \\ & + \sum_{d,j=1}^{D,J_d} \Big[\langle \boldsymbol{\rho}_{d,j}, \hat{\nabla}_{\boldsymbol{\rho}_{d,j}} \tilde{\mathcal{F}}_t \rangle + \frac{1}{\tilde{\beta}_t}\mathrm{KL}(\check{\mathbf{u}}_{d,j})_t - \frac{\tilde{\upsilon}_t}{\tilde{\beta}_t}\mathrm{KL}(\check{\mathbf{u}}_{d,j})_{t-1} \Big],\end{aligned} \tag{5.5}$$

where we have used the same variables $\tilde{\beta}_t$, $\tilde{\alpha}_t$, $\tilde{\upsilon}_t$, and $\tilde{\gamma}_t$ for the step-size parameters as in the LMC. This for the sake of a unified derivation of the FNG updates.

5.4.3 Fully Natural Gradient Updates

We can solve for Eq. (5.5) by computing derivatives w.r.t $\boldsymbol{\eta}^C$ and $\boldsymbol{\rho}_{d,j}$, and setting to zero (Khan and Lin, 2017). This way we obtain results similar to Eq. (2.6) and (2.7), and arrive to our FNG updates:

$$\boldsymbol{\Sigma}^{C\,-1}_{t+1} = \boldsymbol{\Sigma}^{C\,-1}_{t} + 2\alpha_t \hat{\nabla}_{\boldsymbol{\Sigma}^C} \tilde{\mathcal{F}}_t \tag{5.6}$$

$$\boldsymbol{\mu}^{C}{}_{t+1} = \boldsymbol{\mu}^{C}{}_{t} - \alpha_t \boldsymbol{\Sigma}^{C}{}_{t+1} \hat{\nabla}_{\boldsymbol{\mu}^C} \tilde{\mathcal{F}}_t + \gamma_t \boldsymbol{\Sigma}^{C}{}_{t+1} \boldsymbol{\Sigma}^{C\,-1}_{t} (\boldsymbol{\mu}^{C}{}_{t} - \boldsymbol{\mu}^{C}{}_{t-1}) \tag{5.7}$$

$$\mathbf{V}^{-1}_{d,j,t+1} = \mathbf{V}^{-1}_{d,j,t} + 2\beta_t \hat{\nabla}_{\mathbf{V}_{d,j}} \tilde{\mathcal{F}}_t \tag{5.8}$$

$$\begin{aligned} \mathbf{m}_{d,j,t+1} = \mathbf{m}_{d,j,t} &- \beta_t \mathbf{V}_{d,j,t+1} \hat{\nabla}_{\mathbf{m}_{d,j}} \tilde{\mathcal{F}}_t \\ &+ \upsilon_t \mathbf{V}_{d,j,t+1} \mathbf{V}^{-1}_{d,j,t} (\mathbf{m}_{d,j,t} - \mathbf{m}_{d,j,t-1}), \end{aligned} \tag{5.9}$$

where $\alpha_t = \tilde{\alpha}_t/(1-\tilde{\gamma}_t)$, $\beta_t = \tilde{\beta}_t/(1-\tilde{\upsilon}_t)$, $\gamma_t = \tilde{\gamma}_t/(1-\tilde{\gamma}_t)$ and $\upsilon_t = \tilde{\upsilon}_t/(1-\tilde{\upsilon}_t)$ are positive step-size parameters (see Appendix E for details on the gradients derivation).

5.5 Implementation

Similar to the implementation used for the CCGP and HetMOGP with LMC, here the implementation of our FNG for the HetMOGP with CPM depends on inverting the covariance matrix $\boldsymbol{\Sigma}^C$ in Eq. (5.6). Such a task involves a computational complexity of $\mathcal{O}((JMP + QP + JP + QJ)^3)$, where the terms with the number of inducing points and/or input dimensionality tend to dominate the complexity. Also, the gradient $\hat{\nabla}_{\boldsymbol{\Sigma}^C} \tilde{\mathcal{F}}$ involves computing the Hessian $\hat{\nabla}^2_{\boldsymbol{\theta}\boldsymbol{\theta}} \tilde{\mathcal{L}}$ which can be computationally expensive and prone to suffer from non-positive definiteness. In order to mitigate the complexity issues mentioned above, we assume $\boldsymbol{\Sigma}^C = \text{diag}(\boldsymbol{\sigma}^{C\,2})$, where $\boldsymbol{\sigma}^C$ is a vector of standard deviations, and $\text{diag}(\boldsymbol{\sigma}^{C\,2})$ represents a matrix with the elements of $\boldsymbol{\sigma}^{C\,2}$ on its diagonal. With the aim to adopt a stronger numerical stability by preventing that $\boldsymbol{\sigma}^{C\,2}$ becomes negative, we approximate the Hessian by means of the Gauss-Newton approximation $\hat{\nabla}^2_{\boldsymbol{\theta}\boldsymbol{\theta}} \tilde{\mathcal{L}} \approx \hat{\nabla}_{\boldsymbol{\theta}} \tilde{\mathcal{L}} \circ \hat{\nabla}_{\boldsymbol{\theta}} \tilde{\mathcal{L}}$ (Bertsekas, 1999; Khan et al., 2017b). Thus, for the implementation we use the same approach of Eq. (4.12) and (4.13), where by using $\text{diag}(\boldsymbol{\sigma}^{C\,2})$ we reduce the computational complexity from $\mathcal{O}((JMP+QP+JP+QJ)^3)$ to $\mathcal{O}(JMP + QP + JP + QJ)$. See Appendix G for a pseudo-code of the algorithm.

5.6 Predictive Distribution

In a similar way to chapter 4.6, if we want to make predictions with the HetMOGP with a CPM, it is necessary to solve: $p(\mathbf{y}_*|\mathbf{y}) \approx \int p(\mathbf{y}_*|\mathbf{f}_*)q(\mathbf{f}_*)d\mathbf{f}_*$, where $q(\mathbf{f}_*) = \prod_{d=1}^{D}\prod_{j=1}^{J_d} q(\mathbf{f}_{d,j,*})$. Given that we have introduced a variational distribution $q(\boldsymbol{\theta})$ over all hyper-parameters and inducing points of the model, we could apply a fully Bayesian treatment when calculating $q(\mathbf{f}_{d,j,*}) = \int p(\mathbf{f}_{d,j,*}|\check{\mathbf{u}}, \boldsymbol{\theta})q(\check{\mathbf{u}})q(\boldsymbol{\theta})d\boldsymbol{\theta}d\check{\mathbf{u}}$. In practice, we found that $q(\boldsymbol{\theta})$'s covariance converged to very small values, in general $\text{diag}(\boldsymbol{\sigma}^{C\,2}) \leq 10^{-15}$, and almost all the uncertainty information was concentrated on $q(\check{\mathbf{u}})$'s covariance. Since making predictions with the equations above becomes computationally expensive and most of the uncertainty is represented by the distribution $q(\check{\mathbf{u}})$, we can trade-off the computation by using the MAP solution for $q(\boldsymbol{\theta})$ and completely integrating over the remaining distribution as follows: $q(\mathbf{f}_{d,j,*}) = \int p(\mathbf{f}_{d,j,*}|\check{\mathbf{u}}, \boldsymbol{\theta} = \boldsymbol{\mu}^C)q(\check{\mathbf{u}})d\check{\mathbf{u}}$. When solving these integrals, we arrive to exactly the same solutions in Eq. (5.4), where we simply have to evaluate the matrix covariances $\mathbf{K}_{\mathbf{f}_{d,j,*}\check{\mathbf{u}}}$ and $\mathbf{K}_{\mathbf{f}_{d,j,*}\mathbf{f}_{d,j,*}}$, all at the new inputs $\mathbf{X}_*$.

5.7 Experiments

In this section, we explore the performance of the proposed FNG method for jointly optimising all variational parameters, hyper-parameters and inducing points. We also test the hybrid (HYB) method proposed by Salimbeni et al. (2018), and compare the performance against Adam and SGD methods. We run experiments on different toy and real datasets, for all datasets we use a splitting of 75% and 25% for training and testing respectively. The experiments consist on evaluating the method's performance when starting with 20 different initialisations of $q(\boldsymbol{\theta})$'s parameters to be optimised. We report the NELBO shown in Eq. (5.3), for HetMOGP model with CPM, over the training set, and the NLPD error for the test set; this error metric takes into account the predictions' uncertainty (Quiñonero-Candela et al., 2006). Information about the datasets used in this section can be consulted in section 4.7.2.

5.7.1 Optimising the HetMOGP with CPM on Real Data

In this subsection, we show the performance of our FNG over the convolved MOGP for the model with heterogeneous likelihoods. We use the datasets SARCOS and MOCAP7 with a number of outputs of $D = 7$ and $D = 40$, respectively. Figure 5.1 shows the

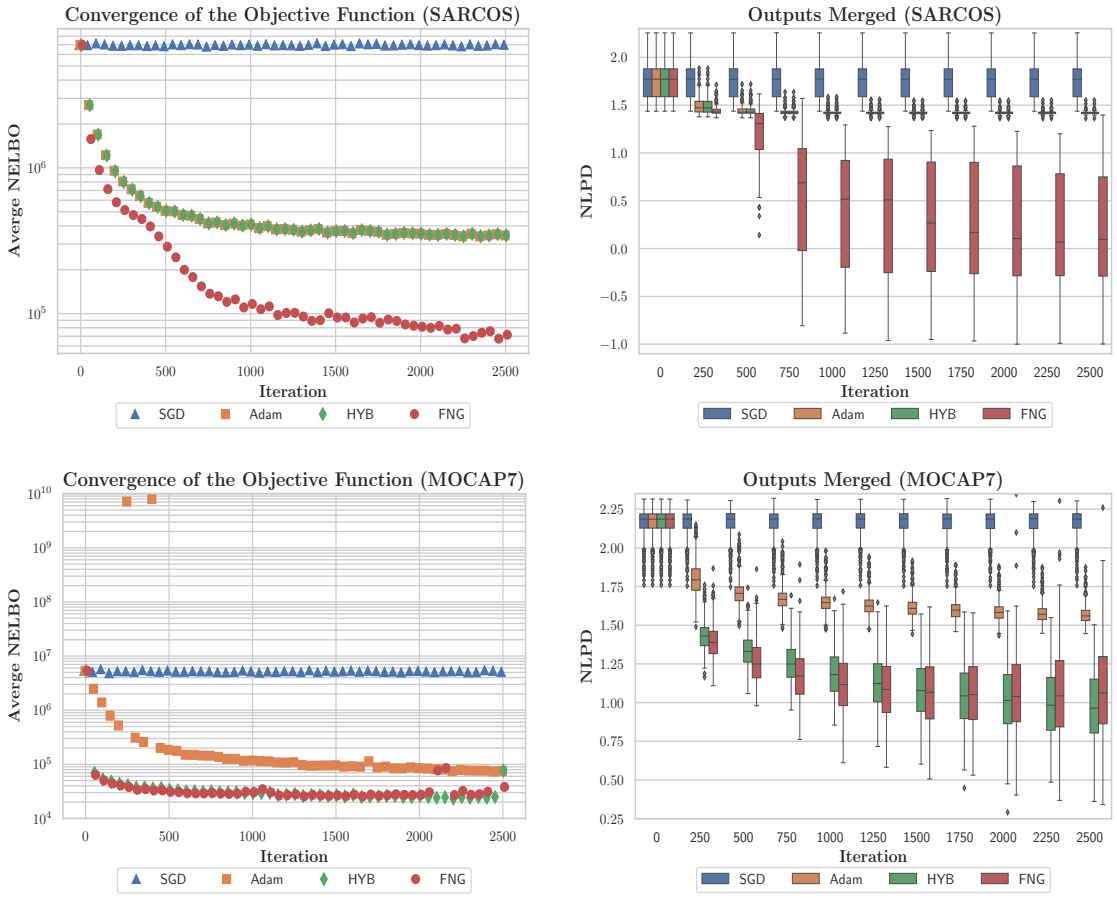


Figure 5.1: *Performance of the diverse inference methods on the SARCOS and MOCAP7 datasets using 20 different initialisations for HetMOGP with CPM. For each dataset we show the average NELBO convergence of each method and the box-plot trending of the NLPD over the test set across all outputs. The box-plots at each iteration follow the legend's order from left to right: SGD, Adam, HYB and FNG. The isolated diamonds that appear in the outputs' graphs represent "outliers".*

performance of the different optimisation methods for fitting the HetMOGP with CPM over such datasets. Similarly to Figure 4.5, we put together the NLPD metric across all outputs. The SARCOS' experiment shows that SGD does not improve much during the optimisation process. Adam and HYB seem to converge to a similar minimum value since the average NELBO and NLPD look very much alike. Otherwise, our FNG method shows to perform much better than the other methods achieving the lowest average NELBO. Also the NLPD trend exhibits a more robust performance over the test set. For MOCAP7, HYB and FNG behave similarly during the optimisation process showing almost the same average NELBO trend. Though, the former method presents a better behaviour when converging at the end. Our FNG method shows a better NLPD performance during the optimisation, but at the end HYB reaches a lower NLPD metric. Adam method accomplishes a poor minima in comparison to HYB and FNG, though a better one than SGD. We can notice from Figures 4.5 and 5.1, both experiments over SARCOS and MOCAP7, that the FNG presents similar convergence patterns in both the LMC and CPM, reaching better solutions than SGD and Adam. In comparison to HYB method, FNG was better for the SARCOS dataset, but presented a similar behaviour to HYB for the MOCAP7 dataset. The next sub-section compares the performance between these two MOGP prior schemes.

5.7.2 Comparing MOGP priors for heterogeneous likelihoods

In this subsection we compare the MOGP models for heterogeneous likelihoods: the one based on the LMC (chapter 4) and the one based on convolution processes. Table 5.1 presents the different NLPD metrics over a test set when using our proposed FNG scheme. Here, we make use of the real datasets from previous sections (see section 4.7.2 for details about the datasets), and we have additionally included two datasets for these experiments:

TRAFFIC Dataset: a record of vehicles traffic, it contains a per-day-number of vehicles passing by the main roads and streets of London city (TRAFFIC, $N = 1712, P = 3, D = 4$). We use a Poisson likelihood per each output of TRAFFIC and assume $Q = 4$ latent functions.

MOCAP9 Dataset: a motion capture data for a running subject (MOCAP9, $N = 744, P = 1, D = 20$). We use a HetGaussian distribution as the likelihood for each output and assume $Q = 3$ functions. We refer to the dataset as MOCAP9 due to the selection of the 9-th subject for the running experiment.

Table 5.1: *NLPD Performance of the Heterogeneous Schemes.*

	LMC		CPM	
Dataset	**Median**	**Mean ± Std**	**Median**	**Mean ± Std**
LONDON	1.025	1.012 ± 0.331	0.986	0.983 ± 0.396
NAVAL	-0.310	-0.318 ± 0.475	-0.429	-0.454 ± 0.527
HUMAN	0.596	0.646 ± 0.764	0.330	0.529 ± 0.807
SARCOS	0.684	0.618 ± 0.581	0.096	0.169 ± 0.605
MOCAP9	0.752	0.774 ± 0.297	1.101	1.172 ± 0.386
MOCAP7	1.344	1.344 ± 0.170	1.078	1.141 ± 0.833
TRAFFIC	72.762	69.947 ± 25.466	68.214	74.866 ± 35.775

Table 5.1 shows that the CPM in general outperforms the LMC for the different real datasets used in our experiments. The NLPD performance, for almost all datasets, shows a considerable improvement when using the convolutional approach, only for MOCAP9 the CPM did not present an improvement over the LMC. The NLPD metric for most of the datasets presents a median very close to the mean, unlike the HUMAN dataset which its mean differs much to the median, though having the median a better trend. Also, we can observe from the Table that generally the standard deviation is higher for the CPM. This is probably due to the additional hyper-parameter set, i.e., the length-scales associated to each smoothing kernel which introduce a larger parameters' space to be explored.

5.7.3 Discussion

Through the different experiments, we could observe that our FNG is a suitable scheme for training another type of MOGP model like the CPM. Indeed, our experiments showed that the CPM can also be trained under a SVI attaining better performance than a HetMOGP based on a LMC. The new HetMOGP model based on convolution processes differs from the original one based on a LMC in the way the inducing variables are introduced. For the LMC, the inducing variables were additional evaluations of the functions $u_q(\cdot)$, while for the CPM the inducing variables are additional evaluations of the functions $f_{d,j}(\cdot)$. We implemented the version of CPM using the same style of inducing variables as the LMC though, in practice, we realised that the assumption commonly used in the literature for the posterior, i.e., $q(\mathbf{f}, \mathbf{u}) = p(\mathbf{f}|\mathbf{u})q(\mathbf{u})$ is not sufficiently flexible to fit the LPFs and limits the SVI implementation. Therefore, we

opted for the inducing variables procedure where $\check{u}_{d,j}(\cdot) = f_{d,j}(\cdot)$, which does support the assumption $q(\mathbf{f}, \check{\mathbf{u}}) = p(\mathbf{f}|\check{\mathbf{u}})q(\check{\mathbf{u}})$.

5.8 Summary

In this chapter, we have provided a novel extension of a stochastic scalable HetMOGP model based on convolution processes and derived a fully natural gradient scheme for improving optimisation of such a model. We have shown in the experiments that our FNG scheme reached better local optima solutions with higher test performance rates than HYB, Adam and SGD methods for the HetMOGP model with a CPM. We have provided comparative results between the two types of GP priors for generating the LPFs: the HetMOGP with LMC versus the HetMOGP with a CPM introduced in this chapter. In the next chapter, we will focus on a practical implementation that consists on modelling the citizens mobility in the Chinese city of Guangzhou. To this end, we make use of the CCGP model in conjunction with a Zero-inflated Poisson likelihood to deal with counting data that involve problems of overdispersion.

Chapter 6

CCGP for Modelling Citizens Mobility using a Zero-Inflated Poisson Likelihood

In chapter 3, we introduced the CCGP model based on an LMC for generating the LPFs. In chapter 4, we described how the CCGP model becomes a HetMOGP model in the context of multiple heterogeneous outputs. Then, in chapter 5, we derived an extension of the model termed as the HetMOGP model with a CPM for generating such LPFs. In this chapter, we aim to apply the CCGP models based on LMC and CPM to the real problem of modelling the citizens mobility using a Zero-inflated Poisson likelihood distribution. To the best of our knowledge, a Zero-inflated Poisson likelihood has not been previously implemented together with a GP model. Unlike previous works based on GPs that mainly model the mean parameter of the likelihood with a unique GP prior (BinTayyash et al., 2020), here we propose that each of those likelihood's parameters are modelled as Latent Parameter Functions that follow correlated GPs as detailed in chapters 3, 4 and 5; thus, allowing a higher flexibility to model heteroscedasticity. Also in this chapter, we derive an SVI framework that allows us to use two types of convolution process models in the context of large datasets: 1. CCGP with a convolution processes model, here we particularly make use of the alternative inducing variables, $u_q(\cdot)$, in contrast to the chapter 5 where we used $\breve{u}_{d,j}(\cdot)$; and 2. CCGP with Variational Inducing Kernels (VIKs) (Álvarez et al., 2010), this VIKs approach is an alternative form to generate the LPFs through the convolution processes formalism, by using a double convolution integral. It is worth mentioning that former works have not developed GP models based on CPM and VIKs for other

type of likelihoods beyond a Gaussian. In this work, we derive equations that can be used for any type of likelihood. Particularly, we provide results for both CCGP models based on CPM and VIKs for Zero-inflated Poisson and Poisson likelihoods.

Modelling the mobility of persons in a city depends on counting data that inherently involve problems of overdispersion. Such overdispersion issues are caused by an excess of observations with values at zero, i.e., zero-inflated counts (Zuur et al., 2009). In order to tackle those issues, different types of machine learning models have focused on fitting the excessive dispersion in data caused by the zero-inflation. For instance, via Generalised Linear Models (GLMs) (Murphy, 2013) using likelihoods like the zero-inflated Poisson (ZIP or ZI-Poisson) (Long and Freese, 2014; Roemmele, 2019), the zero-inflated negative Binomial (ZINB) (Long and Freese, 2014), or the Tweedie distribution (Smyth and Jørgensen, 2002; Bonat et al., 2018), etc. Though these models have been useful to overcome the problems associated to the zero-inflation, they still lack of the ability to appropriately model the spatio-temporal correlations of data associated to mobility. A more powerful alternative for exploiting such spatio-temporal correlations relies on Gaussian process models; nonetheless, few works have taken advantage of their application to improve the forecasting for zero-inflated data. For instance, the work by Kahilakoski (2011) proposes the use of GPs together with a Zero-inflated Poisson likelihood for the analysis of sickness absence; the authors discuss that GP models might yield better predictive performance than hurdle models. Although they did not implement them because of numerical stability issues. The authors in Hegde et al. (2018) propose a zero-inflated formalism that consists of a Gaussian likelihood whose mean follows a latent GP, and a separate 'on-off' probit-linked GP for generating a sparse kernel that allows the model to predict zeros; this work lacks of capturing heteroscedastic noise (an inherent trait of counting data), this due to assuming a Gaussian likelihood where the noise variance is considered constant along all the observations. Also, in BinTayyash et al. (2020), the authors use a ZINB likelihood with a GP prior to model temporal and spatial counting data from RNA-sequencing experiments; this approach presents a unique latent function that models the mean of the Negative Binomial term with a GP prior, while the dispersion parameter and the so called Michalis parameter are assumed free parameters. To the best of our knowledge there are not other works based on GPs that have been concerned about solving the zero-inflation issues while exploiting the spatio-temporal correlations, but the ones mentioned before.

In this work, we aim to model the citizens mobility in the Chinese city of Guangzhou, this from counting data of persons present at a delta area of the city. Since there are

not previous works that explore the behaviour of the Zero-inflated Poisson likelihood with GP priors, here we concentrate on such a likelihood; a ZIP likelihood is more appropriate than the Gaussian likelihood used in Hegde et al. (2018), given that the statistical data type of the observations are counts, i.e., non-negative values. Also, though the work in BinTayyash et al. (2020) considers the statistical data type of counts by assuming a ZINB likelihood, it only uses a GP prior to model the mean of the Negative Binomial term while the other parameters are considered a constant, this way limiting the modelling flexibility. Unlike this latter work, here we propose that the likelihood's parameters are modelled as latent functions drawn from correlated GP priors, this way allowing a higher flexibility to model heteroscedasticity.

In the context of GP models, where each parameter of the likelihood is chained to a GP latent function, three ways to generate such latent functions include: 1. each latent function follows an independent GP prior (Saul et al., 2016) (see sections 3.1 to 3.4); 2. each latent function is generated from a linear model of coregionalisation (see section 3.5 and chapter 4), i.e., a weighted sum of GP priors (Álvarez et al., 2012; Moreno-Muñoz et al., 2018); and 3. from convolution processes, i.e., a convolution integral between smoothing kernels and GP priors (Boyle and Frean, 2005; Álvarez et al., 2010; Álvarez and Lawrence, 2011) (see chapter 5). The above generative alternatives for the latent functions have been broadly used to model either a single or multiple outputs in diverse application scenarios (Álvarez et al., 2012). In the specific ambit of modelling urban traffic in different areas of a city, the work by Rodriguez-Deniz et al. (2017) focuses on forecasting vehicles traffic speeds using an intrinsic coregionalisation model (a particular case of the LMC). Also, the work by Rodrigues et al. (2019) uses a model based on convolution processes to fit spatial and temporal patterns in crowdsourced traffic data. Nevertheless, these previous works become prohibitive in the context of a large number of data observations. To tackle such scalability issues, we additionally derive a stochastic variational inference (Hoffman et al., 2013; Blei et al., 2017) framework that allows the use of this type of models when having massive amounts of data observations. Besides the convolution processes model, we also introduce a scalable version of the variational inducing kernels approach (Álvarez et al., 2010). This VIKs approach is an alternative form to generate the LPFs through the convolution processes formalism, by using a double convolution integral; i.e., the LPF is drawn from a convolution integral between a smoothing kernel and an *inducing function* (IF), where such an IF is an artificial construction generated from another convolution integral between a smoothing kernel and a GP prior.

6.1 Correlated Chained GP with a Convolution Processes Model

This section explains the construction of a CCGP model that introduces correlations between the LPFs through the use of convolution processes. Here, we derive an alternative version of the CCGP model based on convolution process (Higdon, 2002; Boyle and Frean, 2005), it differs from chapter 5 in the way we apply the inducing variables approach. Also, we explain how such inducing variables approach allows the model to obtain tractable variational bounds suitable to SVI.

6.1.1 Convolution Processes for Generating the LPFs

A more general way to derive the latent parameter functions relies on the convolution processes model (Álvarez et al., 2010, 2012). In this type of model, the LPFs are generated by convolving Q latent processes $u_q(\cdot)$ with smoothing kernels $G_{j,q}(\cdot)$, i.e., the LPFs are drawn from $f_j(\mathbf{x}) = \sum_{q=1}^{Q} \int_{\mathcal{X}} G_{j,q}(\mathbf{x} - \mathbf{r}')\, u_q(\mathbf{r}')\, \mathrm{d}\mathbf{r}'$. Alternatively, we can express the latter equation as influenced by multiple latent functions $u_q^i(\cdot)$:

$$f_j(\mathbf{x}) = \sum_{q=1}^{Q} \sum_{i=1}^{R_q} \int_{\mathcal{X}} G_{j,q}^i(\mathbf{x} - \mathbf{r}')\, u_q^i(\mathbf{r}')\, \mathrm{d}\mathbf{r}', \tag{6.1}$$

where each $u_q^i(\cdot)$ represents a latent function drawn IID from $u_q(\cdot) \sim \mathcal{GP}(0, k_q(\cdot,\cdot))$; and R_q represents the number of said IID samples drawn per q-th latent function $u_q(\cdot)$ (Álvarez and Lawrence, 2011). Notice that the equation above is analogous to the linear model of coregionalisation presented in section 3.5, where there are usually Q groups of latent functions $u_q(\cdot)$, and each IID sample $u_q^i(\cdot)$ has the same covariance $k_q(\cdot,\cdot)$ (Journel and Huijbregts, 1979). To ease the derivations in the following sections, we will refer to R instead of R_q, i.e., the number of samples $u_q^i(\cdot)$ per q-th latent function $u_q(\cdot)$ is the same for all Q groups (see section 5.1 for additional details about the relevance of the CPM and the generative process of LPFs for $D > 1$).

6.1.2 Augmented Gaussian Process Prior

As we presented in previous chapters, a common approach to reduce the computational complexity in a GP model consists on augmenting the GP prior with a set of *inducing variables* $u(\cdot)$. Such *inducing variables* represent additional function evaluations of some unknown inducing points $\mathbf{Z} = [\mathbf{Z}_1^\top, ..., \mathbf{Z}_Q^\top]^\top \in \mathbb{R}^{QM \times P}$, with

$\mathbf{Z}_q = [\mathbf{z}_q^{(1)}, ..., \mathbf{z}_q^{(M)}]^\top \in \mathbb{R}^{M\times P}$ (Snelson and Ghahramani, 2006; Titsias, 2009). We can write the augmented GP prior as follows,

$$p(\mathbf{f}|u)p(u|\mathbf{u})p(\mathbf{u}) = \prod_{j=1}^{J} p(\mathbf{f}_j|u)p(u|\mathbf{u})p(\mathbf{u}), \tag{6.2}$$

where $u = [{u_1^1}^\top, ..., {u_Q^1}^\top, ..., {u_1^R}^\top, ..., {u_Q^R}^\top]^\top$ represents a vector of functions that stacks all R IID samples $u_q^i(\cdot)$ of all groups Q; here, u is seen as a continuous version infinitely evaluated at all possible values $\mathbf{x} \in \mathbb{R}^P$. A finite evaluation of u, for instance over the set of inducing points, is expressed as $\mathbf{u} = [{\mathbf{u}_1^1}^\top, ..., {\mathbf{u}_Q^1}^\top, ..., {\mathbf{u}_1^R}^\top, ..., {\mathbf{u}_Q^R}^\top]^\top \in \mathbb{R}^{QMR\times 1}$ with $\mathbf{u}_q^i = [u_q^i(\mathbf{z}_q^{(1)}), \ldots, u_q^i(\mathbf{z}_q^{(M)})]^\top \in \mathbb{R}^{M\times 1}$ (Álvarez and Lawrence, 2009; Álvarez et al., 2010). Particularly, the distributions of the GP prior follow the form: $p(\mathbf{f}_j|u) = \mathcal{N}(m_{f_j u}(\mathbf{X}), \mathbf{0}) = \delta(\mathbf{f}_j - m_{f_j u}(\mathbf{X}))$, where $m_{f_j u}(\mathbf{X}) = [m_{f_j u}(\mathbf{x}_1), ..., m_{f_j u}(\mathbf{x}_N)]^\top \in \mathbb{R}^{N\times 1}$ is a vector built from:

$$m_{f_j u}(\mathbf{x}) = \sum_{q=1}^{Q}\sum_{i=1}^{R} \int_{\mathcal{X}} G_{j,q}^i(\mathbf{x} - \mathbf{r}')\, u_q^i(\mathbf{r}')\, \mathrm{d}\mathbf{r}',$$

and $p(u|\mathbf{u}) = \mathcal{N}(u|k_{u\mathbf{u}}\mathbf{K}_{\mathbf{uu}}^{-1}\mathbf{u}, V_u)$ is a distribution over the vector of functions, conditioned on the finite vector of inducing variables $\mathbf{u}$; $\mathbf{K}_{\mathbf{uu}} \in \mathbb{R}^{QMR\times QMR}$ is a block-diagonal matrix with blocks $\mathbf{K}_{\mathbf{u}_q^i\mathbf{u}_q^i}$ which entries are calculated with $\mathrm{Cov}\left[u_q^i(\cdot), u_q^i(\cdot)\right] = k_q(\cdot,\cdot)$, between all pairs of inducing points $\mathbf{Z}_q$; $V_u = k_{uu} - k_{u\mathbf{u}}\mathbf{K}_{\mathbf{uu}}^{-1}k_{\mathbf{u}u}$, where $k_{uu} = \mathrm{Cov}[u,u]$ can be understood as a continuos matrix covariance infinitely evaluated, and $k_{u,\mathbf{u}} = \mathrm{Cov}[u,\mathbf{u}]$ is a cross-covariance matrix with continuous rows and finite columns; and finally $p(\mathbf{u}) = \mathcal{N}(\mathbf{u}|\mathbf{0}, \mathbf{K}_{\mathbf{uu}})$. It is worth noticing that the augmented GP prior relies on a set of inducing variables that are additional evaluations of $u_q(\cdot)$ over the inducing points; whilst in chapter 5, the inducing variables framework consisted on additional evaluations of, $\breve{u}_j(\cdot) = f_j(\cdot)$, over the inducing points.

6.1.3 The Evidence Lower Bound

As we described for the GP models in chapters 3, 4 and 5, posterior inference is analytically intractable for non-Gaussian likelihoods and approximations are needed instead. To overcome this issue, the inducing variables framework combined with the VI mechanism described in section 2.3, allow us to build a tractable objective bound. Therefore, we approximate the true posterior $p(\mathbf{f}, u, \mathbf{u}|\mathbf{y})$ with a variational distribution $q(\mathbf{f}, u, \mathbf{u})$ by optimising the following ELBO (Blei et al., 2017):

$$\mathcal{L} = \mathbb{E}_{q(\mathbf{f},u,\mathbf{u})}\Big[\log \frac{p(\mathbf{y}|\mathbf{f})p(\mathbf{f}|u)p(u|\mathbf{u})p(\mathbf{u})}{q(\mathbf{f}, u, \mathbf{u})}\Big]. \tag{6.3}$$

We set a variational posterior distribution as follows: $q(\mathbf{f}, u, \mathbf{u}) = p(\mathbf{f}|u)p(u|\mathbf{u})q(\mathbf{u})$ $= \prod_{j=1}^{J} p(\mathbf{f}_j|u)p(u|\mathbf{u})q(\mathbf{u})$ for which $q(\mathbf{u}) = \mathcal{N}(\mathbf{u}|\mathbf{m}, \mathbf{V})$ is a free parametrised distribution, with mean $\mathbf{m} \in \mathbb{R}^{QMR\times 1}$ and a block-diagonal covariance matrix $\mathbf{V} \in \mathbb{R}^{QMR\times QMR}$, with blocks given by $\mathbf{V}_{q,i} \in \mathbb{R}^{M\times M}$ (Álvarez et al., 2010; Hensman et al., 2013; Moreno-Muñoz et al., 2018). After replacing the posterior distribution at Eq. (6.3) and arranging terms, we end up with the following objective for the ELBO:

$$\mathcal{L} = \sum_{n=1}^{N} \mathbb{E}_{q(\mathbf{f})}\big[g_n\big] - \mathbb{D}_{KL}\big(q(\mathbf{u})||p(\mathbf{u})\big), \tag{6.4}$$

where $g_n = \log p\left(y_n|\psi_1\left(\mathbf{x}_n\right), ..., \psi_J\left(\mathbf{x}_n\right)\right)$ is the Log Likelihood (LL) function. The expectation above associated to the LL is computed with regard to the marginal posterior, $q(\mathbf{f}) = \int\int \prod_{j=1}^{J} p(\mathbf{f}_j|u)p(u|\mathbf{u})q(\mathbf{u})d\mathbf{u}du$. Solving for the integrals above, we arrive to:

$$q(\mathbf{f}) := \mathcal{N}(\mathbf{f}|\tilde{\mathbf{m}}_{\mathbf{fu}}, \tilde{\mathbf{V}}_{\mathbf{fu}}), \tag{6.5}$$

having the following definitions, $\tilde{\mathbf{m}}_{\mathbf{fu}} := \mathbf{A}_{\mathbf{fu}}\mathbf{m}$; $\mathbf{A}_{\mathbf{fu}} = \mathbf{K}_{\mathbf{fu}}\mathbf{K}_{\mathbf{uu}}^{-1}$; and $\tilde{\mathbf{V}}_{\mathbf{fu}} := \mathbf{K}_{\mathbf{ff}} + \mathbf{A}_{\mathbf{fu}}(\mathbf{V} - \mathbf{K}_{\mathbf{uu}})\mathbf{A}_{\mathbf{fu}}^{\top}$; where $\mathbf{K}_{\mathbf{fu}} = [\mathbf{K}_{\mathbf{f}_1\mathbf{u}}^{\top}, ..., \mathbf{K}_{\mathbf{f}_J\mathbf{u}}^{\top}]^{\top} \in \mathbb{R}^{JN\times QMR}$ is a cross covariance matrix built with blocks $\mathbf{K}_{\mathbf{f}_j\mathbf{u}} = [\mathbf{K}_{\mathbf{f}_j\mathbf{u}_1^1}, ..., \mathbf{K}_{\mathbf{f}_j\mathbf{u}_Q^1}, ..., \mathbf{K}_{\mathbf{f}_j\mathbf{u}_1^R}, ..., \mathbf{K}_{\mathbf{f}_j\mathbf{u}_Q^R}] \in \mathbb{R}^{N\times QMR}$, with $\mathbf{K}_{\mathbf{f}_j\mathbf{u}_q^i} \in \mathbb{R}^{N\times M}$ constructed with entries calculated from $\mathrm{Cov}\left[f_j(\mathbf{x}), u_q^i(\mathbf{z})\right]$ between the data observations $\mathbf{X}$ and the inducing points $\mathbf{Z}_q$; and $\mathbf{K}_{\mathbf{ff}} \in \mathbb{R}^{JN\times JN}$ is a matrix built with evaluations of the covariance function $\mathrm{Cov}\left[f_j\left(\mathbf{x}\right), f_{j'}\left(\mathbf{x}'\right)\right]$ between all pairs of data observations $\mathbf{X}$. In the following subsection, we describe the specific form of the covariance functions introduced above.

6.1.4 Covariance Functions for CCGP with CPM

For all our models we assume kernel covariance functions with the Exponentiated Quadratic form described in Eq. (4.4). Therefore, with the such a functional form we can define the following kernels for our CCGP model with CPM:

$$k_q(\mathbf{x}, \mathbf{x}') = \mathcal{E}(\boldsymbol{\tau}|\mathbf{0}, \mathbf{L}_q), \tag{6.6}$$

$$G_{j,q}^{i}(\mathbf{x}, \mathbf{x}') = S_{j,q}^{i}\mathcal{E}(\boldsymbol{\tau}|\mathbf{0}, \boldsymbol{\kappa}_j), \tag{6.7}$$

where $\mathbf{L}_q$ and $\boldsymbol{\kappa}_j$ are diagonal matrices of length-scales, and $S_{j,q}^{i}$ is a weight associated to the LPF $f_j(\cdot)$ and to the i-th sample of the latent function $u_q(\cdot)$. Having the definitions of our kernels, we can solve the convolution integrals associated to the covariance functions, $\mathrm{Cov}\left[f_j(\mathbf{x}), u_q^i(\mathbf{z})\right] = \int_{\mathcal{X}} G_{j,q}^{i}(\mathbf{x} - \mathbf{r}')k_q(\mathbf{r}', \mathbf{z})\mathrm{d}\mathbf{r}'$ and $\mathrm{Cov}\left[f_j\left(\mathbf{x}\right), f_{j'}\left(\mathbf{x}'\right)\right] =$

$\sum_{q=1,i=1}^{Q,R} \int_{\mathcal{X}} G_{j,q}^{i}(\mathbf{x}-\mathbf{r}) \int_{\mathcal{X}} G_{j',q}^{i}(\mathbf{x}'-\mathbf{r}')\, k_q(\mathbf{r},\mathbf{r}')\mathrm{d}\mathbf{r}\mathrm{d}\mathbf{r}'$. To solve such integrals above, we follow the work in Álvarez and Lawrence (2011), where the authors apply methodically an identity for the product of two Gaussian distributions. This way, we arrive to the solutions: $\mathrm{Cov}\left[f_j(\mathbf{x}), u_q^i(\mathbf{x}')\right] = S_{j,q}^{i}\mathcal{E}(\boldsymbol{\tau}|\mathbf{0}, \boldsymbol{\kappa}_j + \mathbf{L}_q)$, and $\mathrm{Cov}\left[f_j(\mathbf{x}) f_{j'}(\mathbf{x}')\right] = \sum_{q=1}^{Q} S_{j,q}^{i} S_{j',q}^{i}\mathcal{E}(\boldsymbol{\tau}|\mathbf{0}, \mathbf{P}_{j,j',q})$, where $\mathbf{P}_{j,j',q}$ represents a diagonal matrix of length-scales, $\mathbf{P}_{j,j',q} = \boldsymbol{\kappa}_j + \boldsymbol{\kappa}_{j'} + \mathbf{L}_q$.

6.1.5 Making Predictions with CCGP based on a CPM

To make predictions with our proposed model, we have to compute $p(\mathbf{y}_* \mid \mathbf{y}) \approx \int p(\mathbf{y}_* \mid \mathbf{f}_*)\, q(\mathbf{f}_*)\, d\mathbf{f}_*$, where $q(\mathbf{f}_*)$ can be computed using Eq. (6.5), but building the different covariances matrices $\mathbf{K}_{\mathbf{f}_*\mathbf{u}}$ and $\mathbf{K}_{\mathbf{f}_*\mathbf{f}_*}$ with evaluations at the new inputs $\mathbf{X}_*$ using equations from section 6.1.4.

6.2 Correlated Chained GP with Variational Inducing Kernels

This section describes the construction of the CCGP model with variational inducing kernels (Álvarez et al., 2010). Also, it explains how to obtain a variational objective of the model which is suitable for training by means of SVI (Hoffman et al., 2013; Blei et al., 2017).

6.2.1 Variational Inducing Kernels for Generating the LPFs

The concept of variational inducing kernels was proposed by Álvarez et al. (2010) as an alternative and more powerful way of defining an inducing variable (Snelson and Ghahramani, 2006; Titsias, 2009). It consists on applying a convolution of the latent function $u_q(\cdot)$ with a smoothing kernel as follows:

$$\iota_q(\mathbf{z}) = \int_{\mathcal{X}} T_q(\mathbf{z}-\mathbf{r}) u_q(\mathbf{r})\mathrm{d}\mathbf{r}, \tag{6.8}$$

where $T_q(\mathbf{z}-\mathbf{r})$ is a smoothing kernel, also known as the *inducing kernel* (IK) and $\iota_q(\mathbf{z})$ is called an *inducing function*; and the latent function is drawn from a GP, $u_q(\cdot) \sim \mathcal{GP}(0, k(\cdot,\cdot))$. This idea of inducing function and inducing kernel is closely akin to the works on sparse multi-scale Gaussian process regression by Walder et al. (2008) and inter-domain Gaussian processes by Lázaro-Gredilla and Figueiras-Vidal

(2009). The VIKs allow us to define more general inducing variables with higher approximation capacities than the inducing variables $u_q(\cdot)$ used in Eq. (6.1) for the CCGP with CPM (Álvarez et al., 2012). Though the motivation to use the VIKs in our work relies on the fact of increasing the predictive capabilities of our model, this approach is also useful to deal with possible white noise latent functions $u_q(\cdot)$ when applicable.

As we mentioned before in chapter 3 for Eq. (3.1), each latent parameter function, $f_j(\cdot)$, aims to model the j-th parameter of the likelihood, i.e., each $\psi_j(\mathbf{x}_n) = \alpha(f_j(\mathbf{x}_n))$. Unlike the convolution processes model in Eq. (6.1), which is particularly based on the inducing variables $u_q(\cdot)$, the LPFs can also be drawn from a convolution integral between a smoothing kernel and an inducing function $\iota_q(\cdot)$ as follows:

$$f_j(\mathbf{x}) = \sum_{q=1}^{Q} \sum_{i=1}^{R_q} \int_{\mathcal{X}} G_{j,q}^{i}(\mathbf{x} - \mathbf{r}')\, \iota_q^i(\mathbf{r}')\, \mathrm{d}\mathbf{r}', \tag{6.9}$$

where $G_{j,q}^{i}(\cdot)$ represents the smoothing kernel; and $\iota_q^i(\cdot)$ is an inducing function associated to the i-th sample $u_q^i(\cdot)$ taken IID from $u_q(\cdot) \sim \mathcal{GP}(0, k_q(\cdot,\cdot))$, i.e., as per Eq. (6.8): $\iota_q^i(\mathbf{z}) = \int_{\mathcal{X}} T_q(\mathbf{z}-\mathbf{r}) u_q^i(\mathbf{r})\mathrm{d}\mathbf{r}$; and R_q represents the number of IID samples drawn per q-th inducing function $\iota_q(\cdot)$ (Álvarez and Lawrence, 2011). Thereby, the equation above is an alternative approach to generate the GP priors for modelling the likelihood's parameters under the VIKs approach. As we assumed for the CPM, instead of R_q in Eq. (6.9), we will refer to the same number R of IID samples for all Q groups of inducing functions.

6.2.2 Augmented Gaussian Process Prior

We follow a similar inducing variables framework used for the model CCGP with Convolution processes. It is worth noticing that for the convolution processes model, the function $u(\cdot)$ is the one representing the inducing variable that augments the GP prior (see Eq. (6.2)). Conversely, in this case of VIKs, the vector function $\iota = [\iota_1^{1\top}, ..., \iota_Q^{1\top}, ..., \iota_1^{R\top}, ..., \iota_Q^{R\top}]^\top$ is the one used to augment the GP prior, and from which we compute additional evaluations over the set of unknown inducing points $\mathbf{Z} = [\mathbf{Z}_1^\top, ..., \mathbf{Z}_Q^\top]^\top \in \mathbb{R}^{QM\times P}$, with $\mathbf{Z}_q = [\mathbf{z}_q^{(1)}, ..., \mathbf{z}_q^{(M)}]^\top \in \mathbb{R}^{M\times P}$ (Álvarez et al., 2010). We express the augmented GP prior as follows, $p(\mathbf{f}|\iota)p(\iota|\boldsymbol{\iota})p(\boldsymbol{\iota}) = \prod_{j=1}^{J} p(\mathbf{f}_j|\iota)p(\iota|\boldsymbol{\iota})p(\boldsymbol{\iota})$, where ι, the inducing function, is a continuous function infinitely computed at all possible $\mathbf{x} \in \mathbb{R}^{P\times 1}$; whilst a finite evaluation of ι, for example over the inducing points can be expressed as $\boldsymbol{\iota} = [\boldsymbol{\iota}_1^{1\top}, ..., \boldsymbol{\iota}_Q^{1\top}, ..., \boldsymbol{\iota}_1^{R\top}, ..., \boldsymbol{\iota}_Q^{R\top}]^\top \in \mathbb{R}^{QMR\times 1}$ with $\boldsymbol{\iota}_q^i =$

$[\iota_q^i(\mathbf{z}_q^{(1)}), \ldots, \iota_q^i(\mathbf{z}_q^{(M)})]^\top \in \mathbb{R}^{M\times 1}$. The specific terms of the augmented GP prior can be written as: $p(\mathbf{f}_j|\iota) = \mathcal{N}(m_{f_j\iota}(\mathbf{X}), \mathbf{0}) = \delta(\mathbf{f}_j - m_{f_j\iota}(\mathbf{X}))$, where the mean $m_{f_j\iota}(\mathbf{X}) = [m_{f_j\iota}(\mathbf{x}_1), ..., m_{f_j\iota}(\mathbf{x}_N)]^\top \in \mathbb{R}^{N\times 1}$ is a vector built from:

$$m_{f_j\iota}(\mathbf{x}) = \sum_{q=1}^{Q}\sum_{i=1}^{R}\int_{\mathcal{X}} G_{j,q}^i(\mathbf{x}-\mathbf{r}')\,\iota_q^i(\mathbf{r}')\,\mathrm{d}\mathbf{r}',$$

and $p(\iota|\boldsymbol{\iota}) = \mathcal{N}(\iota|k_{\iota\boldsymbol{\iota}}\mathbf{K}_{\boldsymbol{\iota}\boldsymbol{\iota}}^{-1}\boldsymbol{\iota}_j, V_\iota)$, is a distribution over the continuous inducing function ι, conditioned on $\boldsymbol{\iota}$; $\mathbf{K}_{\boldsymbol{\iota}\boldsymbol{\iota}} \in \mathbb{R}^{QMR\times QMR}$ is a block-diagonal matrix with blocks $\mathbf{K}_{\boldsymbol{\iota}_q^i\boldsymbol{\iota}_q^i}$ with entries calculated with $k_{\iota_q^i}(\mathbf{z},\mathbf{z}') := \mathrm{Cov}\left[\iota_q^i(\mathbf{z}), \iota_q^i(\mathbf{z}')\right] = \int_{\mathcal{X}} T_q(\mathbf{z}-\mathbf{r})\int_{\mathcal{X}} T_q(\mathbf{z}'-\mathbf{r}')k_q(\mathbf{r},\mathbf{r}')\mathrm{d}\mathbf{r}\mathrm{d}\mathbf{r}'$ between all pairs of inducing points $\mathbf{Z}_q$; $V_\iota = k_{\iota\iota} - k_{\iota\boldsymbol{\iota}}\mathbf{K}_{\boldsymbol{\iota}\boldsymbol{\iota}}^{-1}k_{\iota\boldsymbol{\iota}}$, where $k_{\iota\iota} = \mathrm{Cov}[\iota,\iota]$ is a continuos matrix covariance infinitely evaluated, and $k_{\iota,\boldsymbol{\iota}} = \mathrm{Cov}[\iota,\boldsymbol{\iota}]$ is a cross-covariance matrix with continuous rows and finite columns; and $p(\boldsymbol{\iota}) = \mathcal{N}(\boldsymbol{\iota}|\mathbf{0},\mathbf{K}_{\boldsymbol{\iota}\boldsymbol{\iota}})$.

6.2.3 The Evidence Lower Bound

In a similar form to the ELBO derivation for the CCGP with CPM, here we approximate the true posterior $p(\mathbf{f},\iota,\boldsymbol{\iota}|\mathbf{y})$ with a variational distribution $q(\mathbf{f},\iota,\boldsymbol{\iota})$ for constructing the following ELBO (Blei et al., 2017):

$$\mathcal{L} = \mathbb{E}_{q(\mathbf{f},\iota,\boldsymbol{\iota})}\Big[\log\frac{p(\mathbf{y}|\mathbf{f})p(\mathbf{f}|\iota)p(\iota|\boldsymbol{\iota})p(\boldsymbol{\iota})}{q(\mathbf{f},\iota,\boldsymbol{\iota})}\Big]. \tag{6.10}$$

We define a variational posterior distribution with the form: $q(\mathbf{f},\iota,\boldsymbol{\iota}) = p(\mathbf{f}|\iota)p(\iota|\boldsymbol{\iota})q(\boldsymbol{\iota})$ $= \prod_{j=1}^{J} p(\mathbf{f}_j|\iota)p(\iota|\boldsymbol{\iota})q(\boldsymbol{\iota})$, where, $q(\boldsymbol{\iota}) = \mathcal{N}(\boldsymbol{\iota}|\mathbf{m},\mathbf{V})$, with mean $\mathbf{m} \in \mathbb{R}^{QMR\times 1}$ and a block-diagonal covariance matrix $\mathbf{V} \in \mathbb{R}^{QMR\times QMR}$, which blocks are given by $\mathbf{V}_q^i \in \mathbb{R}^{M\times M}$. By replacing the posterior $q(\mathbf{f},\iota,\boldsymbol{\iota})$ in Eq. (6.10), we obtain a scalable objective:

$$\mathcal{L} = \sum_{n=1}^{N}\mathbb{E}_{q(\mathbf{f})}\big[g_n\big] - \mathbb{D}_{KL}\big(q(\boldsymbol{\iota})||p(\boldsymbol{\iota})\big), \tag{6.11}$$

where $g_n = \log p\left(y_n|\psi_1(\mathbf{x}_n), ..., \psi_J(\mathbf{x}_n)\right)$ is the LL function (Moreno-Muñoz et al., 2018). In contrast to the objective function in Eq. (6.4) for the CCGP with a CPM, the expectation for the LL in the equation above is calculated with respect to the marginal posterior, $q(\mathbf{f}) = \int\int\prod_{j=1}^{J} p(\mathbf{f}_j|\iota)p(\iota|\boldsymbol{\iota})q(\boldsymbol{\iota})d\boldsymbol{\iota}d\iota$. When solving for these integrals, we obtain the following:

$$q(\mathbf{f}) := \mathcal{N}(\mathbf{f}|\tilde{\mathbf{m}}_{\mathbf{f}\boldsymbol{\iota}}, \tilde{\mathbf{V}}_{\mathbf{f}\boldsymbol{\iota}}), \tag{6.12}$$

where we have defined, $\tilde{\mathbf{m}}_{\mathbf{f}\boldsymbol{\iota}} := \mathbf{A}_{\mathbf{f}\boldsymbol{\iota}}\mathbf{m}$; $\mathbf{A}_{\mathbf{f}\boldsymbol{\iota}} = \mathbf{K}_{\mathbf{f}\boldsymbol{\iota}}\mathbf{K}_{\boldsymbol{\iota}\boldsymbol{\iota}}^{-1}$; and $\tilde{\mathbf{V}}_{\mathbf{f}\boldsymbol{\iota}} := \mathbf{K}_{\mathbf{ff}} + \mathbf{A}_{\mathbf{f}\boldsymbol{\iota}}(\mathbf{V} - \mathbf{K}_{\boldsymbol{\iota}\boldsymbol{\iota}})\mathbf{A}_{\mathbf{f}\boldsymbol{\iota}}^{\top}$; with $\mathbf{K}_{\mathbf{f}\boldsymbol{\iota}} = [\mathbf{K}_{\mathbf{f}_1\boldsymbol{\iota}}^{\top}, ..., \mathbf{K}_{\mathbf{f}_J\boldsymbol{\iota}}^{\top}]^{\top} \in \mathbb{R}^{JN\times QMR}$ as a cross covariance matrix built with blocks $\mathbf{K}_{\mathbf{f}_j\boldsymbol{\iota}} = [\mathbf{K}_{\mathbf{f}_j\boldsymbol{\iota}_1^1}, ..., \mathbf{K}_{\mathbf{f}_j\boldsymbol{\iota}_Q^1}, ..., \mathbf{K}_{\mathbf{f}_j\boldsymbol{\iota}_1^R}, ..., \mathbf{K}_{\mathbf{f}_j\boldsymbol{\iota}_Q^R}] \in \mathbb{R}^{N\times QMR}$, in which, each $\mathbf{K}_{\mathbf{f}_j\boldsymbol{\iota}_q^i} \in \mathbb{R}^{N\times M}$ has entries computed with the covariance function, $\text{Cov}\left[f_j(\mathbf{x}), \iota_q^i(\mathbf{z})\right]$, between the data observations $\mathbf{X}$ and the inducing points $\mathbf{Z}_q$; and $\mathbf{K}_{\mathbf{ff}}$ is a covariance matrix built with evaluations of $\text{Cov}[f_j(\mathbf{x}), f_{j'}(\mathbf{x}')]$, between all pairs of data observation $\mathbf{X}$. In the following subsection, we detail the form of the covariance functions introduced above.

6.2.4 Covariance Functions for CCGP with VIKs

For our CCGP model with VIKs we follows the same EQ form of the kernel covariance functions. Given that this type of model also relies on the latent function $u_q \sim \mathcal{GP}(0, k_q(\cdot,\cdot))$ and a smoothing kernel $G_{j,q}^i(\cdot)$, we make use of exactly the same equations (6.6) for $k_q(\cdot,\cdot)$, and (6.7) for $G_{j,q}^i(\cdot)$. We additionally need to define the inducing kernel $T_q(\cdot)$ in Eq. (6.8), so we use the functional form in Eq. (4.4) to define: $T_q(\mathbf{x},\mathbf{x}') = W_q\mathcal{E}(\boldsymbol{\tau}|\mathbf{0},\mathbf{t}_q)$, where W_q is a weight an $\mathbf{t}_q$ is a diagonal matrix of length-scales. Similar to the case of CPM, we rely on the multiplication identity between Gaussian distribution applied in Álvarez and Lawrence (2011) and Álvarez (2011). Thus, we solve for $k_{\iota_q^i}(\mathbf{z},\mathbf{z}') := \text{Cov}\left[\iota_q^i(\mathbf{z}), \iota_q^i(\mathbf{z}')\right] = \int_{\mathcal{X}} T_q(\mathbf{z}-\mathbf{r})\int_{\mathcal{X}} T_q(\mathbf{z}'-\mathbf{r}')k_q(\mathbf{r},\mathbf{r}')\text{d}\mathbf{r}\text{d}\mathbf{r}'$ and $\text{Cov}\left[f_j(\mathbf{x}), \iota_q^i(\mathbf{z})\right] = \int_{\mathcal{X}} G_{j,q}^i(\mathbf{x}-\mathbf{r})k_{\iota_q^i}(\mathbf{r},\mathbf{z})\text{d}\mathbf{r}$, and arrive to the following covariances: $\text{Cov}\left[\iota_q^i(\mathbf{x}), \iota_q^i(\mathbf{x}')\right] = W_q^2\mathcal{E}(\boldsymbol{\tau}|\mathbf{0}, 2\mathbf{t}_q + \mathbf{L}_q)$ and $\text{Cov}\left[f_j(\mathbf{x}), \iota_q^i(\mathbf{x}')\right] = S_{j,q}^i W_q\mathcal{E}(\boldsymbol{\tau}|\mathbf{0}, \boldsymbol{\kappa}_j + 2\mathbf{t}_q + \mathbf{L}_q)$. Also, when solving for the covariance function:

$$\text{Cov}[f_j(\mathbf{x}), f_{j'}(\mathbf{x}')] = \sum_{q=1}^{Q}\sum_{i=1}^{R}\int_{\mathcal{X}} G_{j,q}^i(\mathbf{x}-\mathbf{v}) \times \int_{\mathcal{X}} G_{j',q}^i(\mathbf{x}'-\mathbf{v}')\, k_{\iota_q^i}(\mathbf{v},\mathbf{v}')\text{d}\mathbf{v}\text{d}\mathbf{v}',$$

we end up with: $\text{Cov}[f_j(\mathbf{x}), f_{j'}(\mathbf{x}')] = \sum_{q=1}^{Q} S_{j,q}^i S_{j',q}^i W_q W_q\mathcal{E}(\boldsymbol{\tau}|\mathbf{0}, \mathbf{T}_{j,j',q})$, where $\mathbf{T}_{j,j',q}$ represents a diagonal matrix of length-scales, $\mathbf{T}_{j,j',q} = \boldsymbol{\kappa}_j + \boldsymbol{\kappa}_{j'} + 2\mathbf{t}_q + \mathbf{L}_q$.

6.2.5 Making Predictions with CCGP based on VIKs

In a similar way to section 6.1.5, we compute $p(\mathbf{y}_* \mid \mathbf{y}) \approx \int p(\mathbf{y}_* \mid \mathbf{f}_*)\, q(\mathbf{f}_*)\, d\mathbf{f}_*$. Notice that the distribution $q(\mathbf{f}_*)$ now involves computations associated to $\boldsymbol{\iota}$ instead of $\mathbf{u}$.

Therefore, for a new set of inputs $\mathbf{X}_*$, we have to build $\mathbf{K}_{\mathbf{f}_*\boldsymbol{\iota}}$ and $\mathbf{K}_{\mathbf{f}_*\mathbf{f}_*}$ as per Eq. (6.12).

6.3 Zero-Inflated Poisson distribution

Since we aim to model non-negative values that represent counts and also tackle the problems associated to zero-inflation data (Zuur et al., 2009; Lukusa et al., 2017), here we rely on the ZIP distribution for targeting such issues (Roemmele, 2019). The ZIP probability distribution can be expressed as follows:

$$p(y_n|\pi_n, \rho_n) = \mathbf{1}_{y_n}\pi_n + (1-\pi_n)\frac{\exp(-\rho_n)\rho_n^{y_n}}{y_n!}, \tag{6.13}$$

where $\pi_n \in [0,1]$ is a parameter that represents a probability for the values at zero (Beckett et al., 2014), $\rho_n > 0$ is the Poisson rate parameter and $\mathbf{1}_{y_n} := \mathbf{1}(y_n)$ is an indicator function defined as follows: $\mathbf{1}(y_n) = 1$ if $y_n = 0$, or $\mathbf{1}(y_n) = 0$ if $y_n \neq 0$. Following the notation in Eq. (3.1), here the distribution's parameters are associated as $\pi_n = \psi_1(\mathbf{x}_n)$ and $\rho_n = \psi_2(\mathbf{x}_n)$, where $\psi_1(\mathbf{x}_n) = \sigma(f_1(\mathbf{x}_n))$ and $\psi_2(\mathbf{x}_n) = \exp(f_2(\mathbf{x}_n))$; $\sigma(f_1(\mathbf{x}_n)) = 1/(1+\exp(-f_1(\mathbf{x}_n)))$ is a sigmoid function. With the definitions above we can link Eq. (6.13) in the Log Likelihood function, $g_n = \log p\left(y_n|\psi_1(\mathbf{x}_n), ..., \psi_J(\mathbf{x}_n)\right)$, of equations (6.4) and (6.11) as follows:

$$g_n = \log\Big(\mathbf{1}_{y_n}\pi_n\exp(\rho_n) + (1-\pi_n)\Big) + \log\left(\frac{\exp(-\rho_n)\rho_n^{y_n}}{y_n!}\right).$$

It is worth noticing that the equation above is exactly the same for both CCGP models based on either the CPM (Eq. (6.4)) or VIK (Eq. (6.11)); They just differ by construction in the way of generating the LPFs, but not in the form of the likelihood function. In the experiments section, we will compare the performance of our proposed CCGP methods, the CPM-based and VIK-based in conjunction with either a ZI-Poisson likelihood or a Poisson likelihood.

6.4 Experiments

In this section, we make a quantitative and qualitative analysis of the predictions obtained by the three types of CCGP models based on: an LMC (presented in section 3.5 and chapter 4), our CPM proposed in Eq. (6.4) and also our VIK introduced in Eq. (6.11). As explained at section 6.3, we implement a ZI-Poisson likelihood and

compare its performance with a Poisson likelihood for modelling the citizens mobility in Guangzhou city. We run two types of experiments: the first corresponds to building a model per each day of the month; the second to building a model per each day of the week. [1]

6.4.1 Dataset of Guangzhou City

The dataset used to model the citizens' mobility in the region of Guangzhou was built from recordings of mobile phone GPS locations. In a nutshell, the users of a Guangzhou's mobile phone network share their longitude and latitude coordinates that are consequently preprocess through a counting algorithm. Such an algorithm consists on counting the citizens that coincide in a delta area of Guangzhou; i.e., the main region of Guangzhou is divided in a grid of 201×201, where each square (or delta area) of the grid contains a total number of citizens. The counting is performed every hour of the day, this during 31 days: from March 1 to 31 of 2019. The total number of data observations per day is $N = 201 \times 201 \times 24 = 969624$.

6.4.2 Model Training

Given that the models derived in Eq. (6.4) and Eq. (6.11) allow stochastic variational inference, we use a random mini-batching of 400 samples per iteration during training. We selected through cross-validation a number of latent functions $Q = 3$, the number of IID samples $R = 2$, and inducing points $M = 200$.[2] It is worth noticing that the expectations of the Log Likelihood in equations (6.4) and (6.11) cannot be computed in closed-form, so we opt for using the Gauss-Hermite quadrature approach (Saul et al., 2016; Jin and Andersson, 2020). Also, it is important to highlight that there is not need to compute the full covariances $\mathbf{K_{ff}}$ in Eq. (6.5) for the CPM-based model or in Eq. (6.12) for the VIK-based model, but only the diagonal values randomly selected as per the mini-batching at each optimisation iteration.

For the inference process, we make use of the VAN updates from Eq. (2.6) and Eq. (2.7) where: each variational parameter $\mathbf{m}$ and $\mathbf{V}$; each set of inducing points $\mathbf{Z}$; and all the kernels' hyper-parameters, $\mathbf{H} = \{\{\boldsymbol{\kappa}_j\}_{j=1}^{J}, \{\mathbf{L}_q\}_{q=1}^{Q}, \{\mathbf{t}_q\}_{q=1}^{Q}\}$, are optimised through a natural gradient scheme. Similar to sections 3.6, 4.4 and 5.4, here we build a bound

[1] The code with the proposed models is publicly available in the repository: https://github.com/juanjogg1987/CorrelatedChainedGPs_ConvolutionProcesses

[2] In the practice, we heuristically found that a suitable way to set R was to make it equal to the total number of LPFs, i.e., $R = J$.

of the form, $\tilde{\mathcal{F}} = \mathbb{E}_{q(\boldsymbol{\theta})}[-\mathcal{L}] + \mathbb{D}_{KL}(q(\boldsymbol{\theta})||p(\boldsymbol{\theta}))$, where $\mathcal{L}$ is any of the objective functions for either the CPM or VIK in Eq. (6.4) or (6.11) respectively; $q(\boldsymbol{\theta})$ represents a free parametrised exploratory distribution over the set of all parameters to optimise, i.e., $\boldsymbol{\theta} = \{\mathbf{m}, \mathbf{V}, \mathbf{Z}, \mathbf{H}\}$. The main difference with the algorithms of previous chapters relies on the inclusion of the parameters $\mathbf{m}$ and $\mathbf{V}$ as part of the exploratory distribution.

6.4.3 Quantitative Results: Models along the Month

The first experiment consists on building 31 CCGP models, one model per day during all the month of March. We use a dataset random split of 90% and 10% for training and testing, respectively. In order to measure the uncertainty quantification capability of the models, we report the NLPD error over the test set (Quiñonero-Candela et al., 2006). Figure 6.1 shows the performance of the CCGP models based on VIK, CPM and LMC, when using a Poisson distribution (top figure) and a ZI-Poisson distribution (bottom figure). Low NLPD values mean better performance.

We can notice from Figure 6.1 that the models with a Poisson likelihood presented metrics roughly within the interval $(2.2, 2.7)$, whilst the models with a ZI-Poisson likelihood obtained metrics approximately within the interval $(0.59, 1.02)$. Thereby, we can say that in general the models relying on a ZI-Poisson likelihood outperformed the ones based on a Poisson likelihood by achieving lower NLPD metrics. For the Poisson likelihood, the VIK model accomplished the lowest NLPD for 22 days of the month in comparison to the CPM and LMC; the CPM presented a better performance than VIK and LMC for six days; and the LMC only presented the lowest NLPD in the days: 13, 16 and 30. For the ZI-Poisson likelihood, the VIK reached better NLPD values for 13 days; the CPM obtained the lowest metrics in 18 times; and the LMC did not present a better performance at any day in comparison to the other CCGP models. Table 6.1

Table 6.1: *Summary of Statistics of NLPD-Test Performance along the month for the CCGP Models based on Poisson and ZI-Poisson Likelihoods using three types of GP Priors.*

	VIK		**CPM**		**LMC**	
	Avg ± Std	**Med**	**Avg ± Std**	**Med**	**Avg ± Std**	**Med**
Poisson	2.401± 0.081	**2.395**	2.448 ± 0.090	2.476	2.507 ± 0.094	2.515
ZIP	0.740 ± 0.053	**0.742**	0.736 ± 0.063	0.752	0.976 ± 0.032	0.982

shows a summary of the main statistics obtained by the models along the month. We

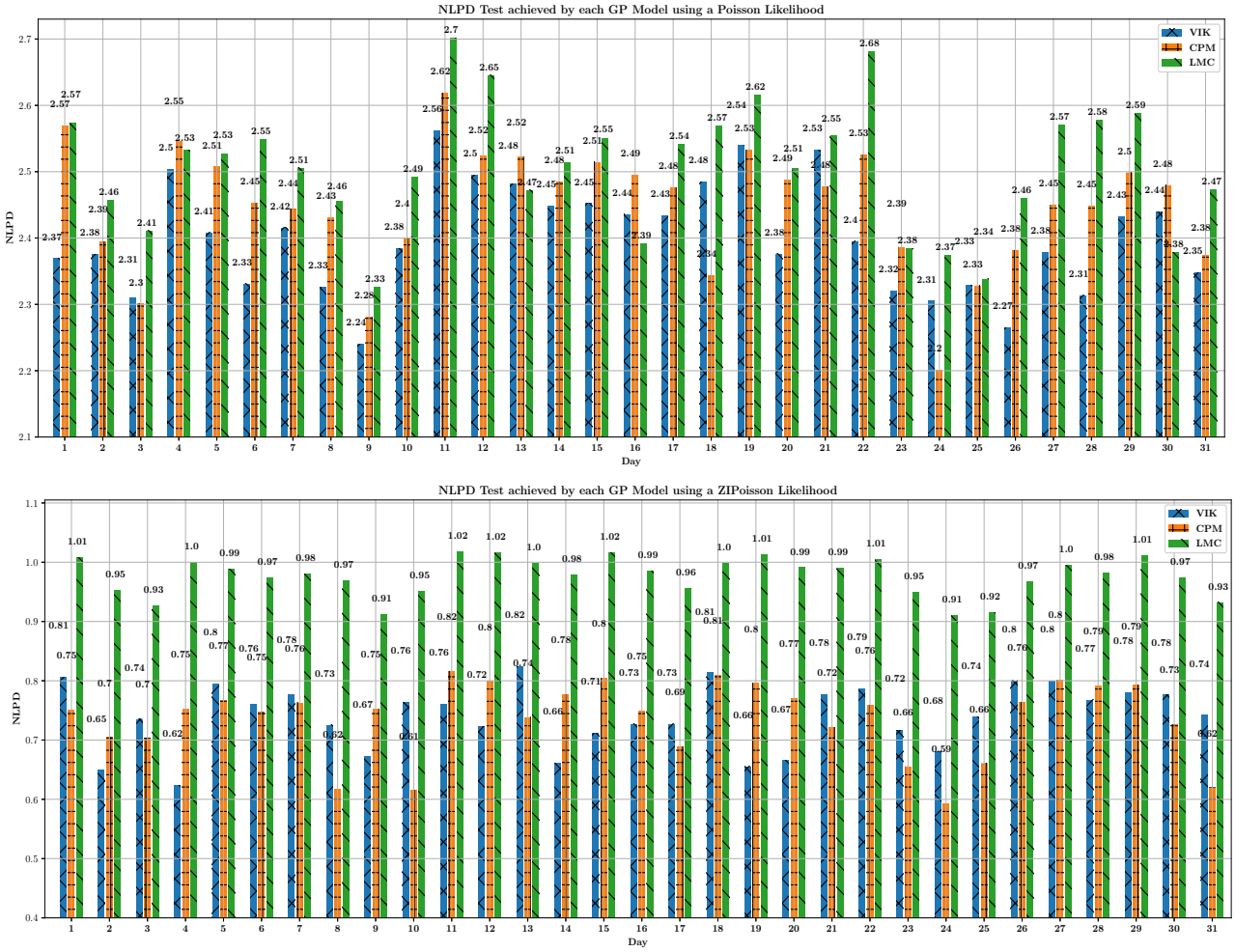


Figure 6.1: *NLPD-Test Performance along the month for the CCGP models based on VIK, CPM and LMC. Top figure: Poisson likelihood. Bottom figure: ZI-Poisson likelihood. For each day there are three bars associated to the GP priors: left bar, VIK with pattern inscription "x"; middle bar, CPM with pattern "+"; and right bar, LMC with pattern "\". Low NLPD values mean better performance.*

can see from the table that the CCGP model based on VIK reached the lowest median for both types of likelihoods; also it tended to present smaller standard deviations than the other methods. The CPM showed a slightly lower mean than the VIK when using a ZI-Poisson, but with a higher standard deviation. The LMC presented similar results to the CPM for the case of a Poisson likelihood; nevertheless, in comparison to VIK and CPM, its performance was poor when modelling with a ZI-Poisson likelihood. The results show that the use of a ZI-Poisson likelihood considerably improved the performance of the prediction capabilities of our CCGP models in the context of the zero-inflated data from Guangzhou city. Regarding the types of GP priors, the VIK and CPM showed to outperform the LMC by allowing better NLPD metrics, i.e., a better quantification of the uncertainty.

6.4.4 Quantitative Results: Models along the Week

For the second experiment, we trained seven CCGP models, one per day of the week, i.e., models for Monday, Tuesday, Wednesday, Thursday, Friday, Saturday and Sunday. In contrast to the first experiment, here we selected the observations from Monday March 4 to Sunday March 10 for training data; whilst the remaining days were used for testing: Mondays (March 11, 18, 25), Tuesday (March 12, 19, 26), Wednesday (March 13, 20, 27), Thursday (March 14, 21, 28), Friday (March 15, 22, 29), Saturday (March 16, 23, 30) and Sunday (March 17, 24, 31). For instance, we trained a model for Monday using data from March 4 and tested it over the remaining Mondays March 11, 18 and 25. Figure 6.2 shows the NLPD error obtained by the different GP models in combination with both types of likelihoods, Poisson and ZI-Poisson. From Figure 6.2 we can observe that the ranges of NLPD metrics accomplished by the CCGP models when using a ZI-Poisson likelihood were lower in comparison to the Poisson likelihood; ZIP metrics were within the range $(0.52, 0.78)$ and Poisson metrics are within $(1.72, 1.97)$. Comparing these latter results with the ones reached in the previous subsection of Models along the Month, we can regard that the CCGP models along the week present a better performance. For the Poisson likelihood, Figure 6.2 shows that the model based on VIK obtained the lowest NLPD values for all the days of the week, followed by the CPM and LMC. For the ZI-Poisson likelihood, Figure 6.2 shows that the model VIK-based reached the lowest NLPD values for Monday and Saturday in comparison to the other methods; whilst the model CPM-based attained the lowest NLPD values for Tuesday, Wednesday, Thursday, Friday and Sunday; the model LMC-based did not present a better performance than the other methods on any of the days. Though, for

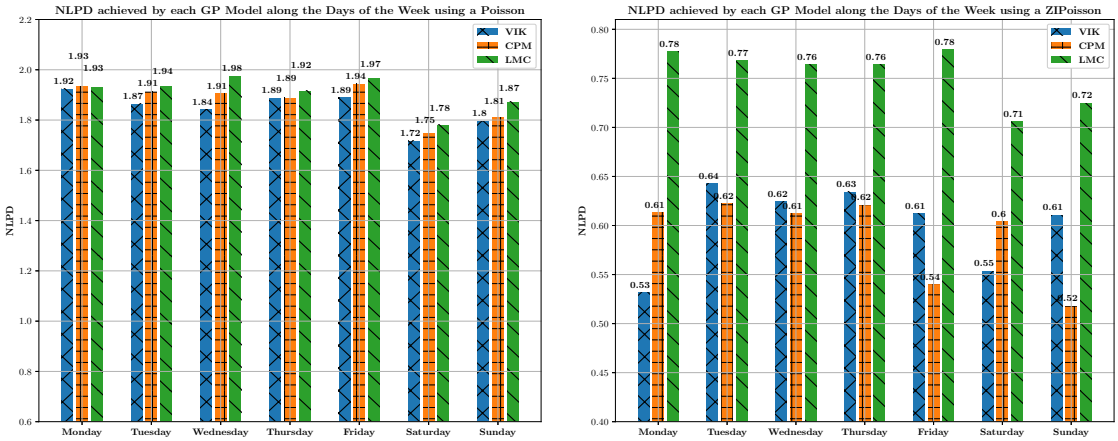


Figure 6.2: *NLPD-Test Performance along the Week for the CCGP models based on VIK, CPM and LMC. Left figure: Poisson likelihood. Right figure: ZIP likelihood. For each day there are three bars associated to the GP priors: left bar, VIK with pattern inscription "x"; middle bar, CPM with pattern "+"; and right bar, LMC with pattern "\". Low NLPD values mean better performance.*

the ZI-Poisson likelihood, the CPM presented better metrics on more days than the VIK, the summary of statistics in Table 6.2 shows that in general the VIK performs similar to the CPM for such a likelihood.

Table 6.2 allows us to see that generally the CCGP model based on VIK obtained better NLPD metrics in comparison to the other methods. The VIK performed quite similar to the CPM in the context of a ZIP likelihood; indeed, VIK and CPM presented a difference in the mean of just 0.011 for the ZIP likelihood, with equal standard deviations and equal medians. Regardless of the type of likelihood, both CPM and VIK models outperformed the LMC model.

Table 6.2: *Summary of Statistics of NLPD-Test Performance along the Week for the CCGP Models based on Poisson and ZI-Poisson Likelihoods using three types of GP Priors.*

	VIK		**CPM**		**LMC**	
	Avg ± Std	**Med**	**Avg ± Std**	**Med**	**Avg ± Std**	**Med**
Poisson	1.847 ± 0.065	**1.865**	1.878 ± 0.067	1.907	1.911 ± 0.062	1.932
ZIP	0.601 ± 0.039	**0.612**	0.590 ± 0.039	0.612	0.755 ± 0.026	0.764

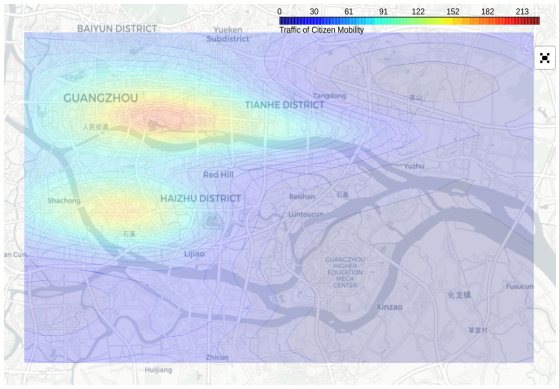
6.4.5 Qualitative Results

Since our data from Guangzhou city presents zero-inflation issues, we aim to observe the effect in the predictions when using the Poisson and ZI-Poisson likelihoods to deal with said problem. Also, in order to visualise the qualitative traits of the model that showed the highest capabilities to generalise, we selected the CCGP model based on VIK (CCGP-VIK), particularly we chose the model for Saturday, March 9 (from Figure 6.2) given that it presented a relevant NLPD performance for both Poisson and ZI-Poisson likelihoods.

Figure 6.3(b) shows the mean prediction of the CCGP-VIK with Poisson likelihood and Figure 6.3(d) the mean prediction of the CCGP-VIK with ZI-Poisson likelihood. Figures 6.3(a) and 6.3(c) are a heatmap of the real test data of the citizens mobility on Saturday, March 16 at 11:00 am; both figures are exactly the same, but displayed twice to ease the comparison to our models' predictions provided in Figures 6.3(b) and 6.3(d). Also, to ease the description of the predictions we will refer to the names that appear in the maps as key locations, for instance the names: GUANGZHOU, TIANHE DISTRICT, Red Hill, Lijiao, Shachong, Xinzao, etc.

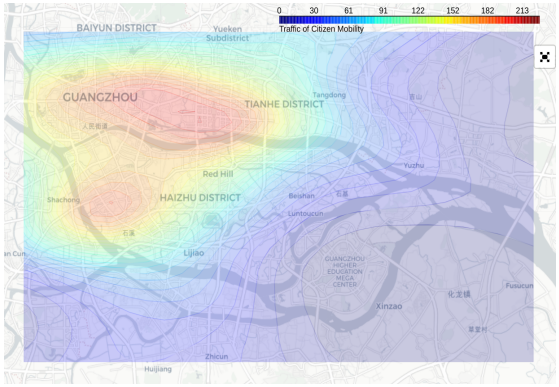
It can be seen from Figures 6.3(b) and Figure 6.3(d) that both CCGP-VIK models with Poisson and ZI-Poisson focus on the high concentrations of citizens in the city centre, that is the region between GUANGZHOU, TIANHE DISTRICT and Red Hill. The models also focus on the region with high numbers of citizens located among Shachong, HAIZHU DISTRICT and Lijiao; that is a central-west region that gathers different Metro-stations like: Jiangnanxi station, Huadiwan station, Xilang station and Jushu station. For the case of the Poisson likelihood, we can notice from Figure 6.3(b) that the predictions of high concentrations of citizens are underestimated in comparison to the test data in Figure 6.3(a); whilst for the case of the ZI-Poisson likelihood, we observe that the density of citizens looks more akin to the test data in Figure 6.3(c). With respect to the regions of the city that present many zero value observations like the north-east and south-east quadrants, we can notice that both the CCGP-VIK models with Poisson and ZI-Poisson predict very low concentrations of citizens in those regions. Although, specifically the model with Poisson likelihood concentrates on predicting massive densities of zero values in the north-east quadrant of the map that extend until Tangdong region; in contrast, the model with ZI-Poisson remains a bit conservative not presenting as huge accumulations of zero values as the Poisson distribution, and allowing to predict moderate concentrations of people around Tangdong. Likewise, for the region in the south-west quadrant below Lijiao, the model

(a) Test Data Heatmap: Saturday, March 16 at 11:00 am



(b) Prediction at 11:00 am using CCGP-VIK with Poisson

(c) Test Data Heatmap: Saturday, March 16 at 11:00 am



(d) Prediction at 11:00 am using CCGP-VIK with ZI-Poisson

Figure 6.3: *Qualitative performance of the model CCGP-VIK (trained with data from Saturday, March 9) in comparison to real test data. To the left hand side, Figures 6.3(a) and 6.3(c) are a heatmap of the real test data of the citizens mobility on Saturday, March 16 at 11:00 am; both figures are the same, but displayed twice to ease the comparison with the predictions to the right hand side. Figure 6.3(b) shows the mean prediction of the CCGP-VIK with Poisson likelihood. Figure 6.3(d) presents the mean prediction of the CCGP-VIK with ZI-Poisson likelihood. The color bar associates the number of citizens in the map area.*

with Poisson likelihood focuses on predicting very low concentrations of citizens, but the model with ZI-Poisson predicts moderate congregations of citizens that vanish from Lijiao towards Huijiang and Zhicun. Both ZIP and Poisson models neglect the concentrations of citizens in the region below Luntoucun and to the left hand side of GUANGZHOU HIGHER EDUCATION MEGA CENTER.

The main quality of the model CCGP-VIK with ZIP consists on appropriately trading-off between making predictions in the regions with high concentration of zeros without underestimating the regions with substantial presence of citizens. Conversely, the model based on a Poisson likelihood is not able to adequately forecast in the regions with major presence of citizens.

6.5 Discussion

Through the different types of experiments we noticed that the use of a ZI-Poisson distribution significantly improved the performance for modelling the citizens' mobility in Guangzhou city, in comparison to a Poisson distribution. For the case of modelling with a Poisson likelihood, the ranking of best performances usually showed the VIK at first, followed by CPM and LMC. We realised the mean NLPD metrics are very close between all the GP methods (with a difference not higher than 0.106) as shown in Table 6.1. We believe those NLPD metrics are close to each other due to the few parameters present in the Poisson distribution, which limit the capabilities of the different GP models for achieving a higher predictive performance. On the other hand, when modelling with the ZI-Poisson, the CCGP models based on VIK and CPM presented a distinguished difference with the LMC. Such a difference can be attributed to the fact of having additional hyper-parameters that allow higher modelling flexibilities than the LMC, which only depends on a set of linear combination coefficients and matrices of length-scales $\mathbf{L}_q$ related to the latent functions $u_q(\cdot)$ (Journel and Huijbregts, 1979; Álvarez et al., 2012). Such additional hyper-parameters can be identified, for instance: from the CPM, in all weights $S_{j,q,i}$ and the matrices of length-scales $\boldsymbol{\kappa}_j$ associated to the smoothing kernels, and the matrices of length-scales $\mathbf{L}_q$ for the latent functions $u_q(\cdot)$ (see section 6.1.4); and apart from the latter parameters present in the CPM, the VIKs model additionally presents weights W_q and the matrices of length-scales $\mathbf{t}_q$ associated to the inducing kernels (see section 6.2.4). Regarding the two types of experiments we carried out for modelling either along the month or along the week, the results showed a better performance in the models along the week. We associate

the high performance reached by such models with a probable high correlation between the training and testing data; the mobility patterns of citizens for instance on Monday March 4 (training data) can be very similar to the remaining Mondays 11, 18 and 25 (testing data), a fact likely to happen also for the other days.

6.6 Summary

In this chapter, we have modelled the citizens mobility in the Chinese city of Guangzhou by means of three types of CCGP models (based on an LMC, CPM or VIKs) with Poisson and ZI-Poisson likelihoods. We showed that all types of CCGP models in conjunction with a ZIP likelihood allow to overcome the issues associated to zero-inflated data, outperforming the predictive capabilities of such CCGP models when based on a Poisson likelihood. We derived a stochastic variational inference framework that grants the use of a CCGP model with CPM or VIKs in the context of a large number of data observations.

Chapter 7

Conclusions and Future Work

In this chapter we summarise the work developed throughout the thesis and provide some insights for future work.

Conclusions

In this thesis we addressed problems of intractability, scalability, and poor local optima solutions associated to non-conjugate likelihood Gaussian process models. Particularly, in this thesis we addressed the aforementioned issues in a context where the likelihood's parameters are modelled as latent parameter functions drawn from correlated Gaussian processes based on LMC, CPM and VIKs. We borrowed ideas from different optimisation mechanisms that allowed us to dealing with intractability as well as enabling scalability when needing to hand massive amounts of data observations. Also, such optimisation mechanisms allowed us to improve inference processes of the models for tackling the problems of poor local optima solutions.

- In chapter 2, we introduced different optimisation mechanisms, like VO, VI, MDA and VAN. We built this thesis upon the idea of using such optimisation mechanisms for dealing with intractability issues present in non-conjugate likelihood Gaussian process models. Such optimisation mechanisms helped us to construct scalable objective functions that granted us the use of our GP models in scenarios with large amounts of data observations; and we benefited from said mechanisms for improving the inference processes of the different GP models presented in this thesis.

- In chapter 3, we introduced the CCGP model, an approach that assumes by

construction that the multiple GP priors of a Single-Output model are correlated through a linear model of coregionalisation. Our approach bases on the so-called inducing variables framework and scales by means of stochastic variational inference. By means of various experiments carried out for different real databases, we showed that the CCGP model achieved rich predictive distributions that quantify the uncertainty better than the classical setting that builds upon independent GP priors. Also, we proposed a strategy of model training that augments a single-output GP in order to treat it as a multi-output one. We found that such strategy enhances the generalisation properties of the model accomplishing a high predictive performance. Experimental results in paper (iii) show how the CCGP model can help to improve the modelling of labellers' behaviour when dealing with datasets involving multiple annotators.

- In chapters 3, 4 and 5 we derived a fully natural gradient scheme for jointly tuning the hyper-parameters, inducing points and variational posterior parameters of: the single-output CCGP model, its multi-output version the HetMOGP with LMC, and the model extension HetMOGP with CPM. Through the experiments carried out in paper (i), we showed that such an scheme helped to improve the inference processes of the different GP models outperforming methods like SGD, Adam, ADAD, and the hybrid strategy of NG+Adam.

- In chapter 5, we provided an extension of the HetMOGP based on a Convolution Processes model, rather than on the LMC approach of the original model by Moreno-Muñoz et al. (2018). By means of experimental results, we showed in paper (i) that generally the HetMOGP with CPM attained better predictive performance than a HetMOGP based on a LMC. Also, given that a NG method had not been performed over any MOGP model before, we contributed to show its performance using two schemes: the hybrid NG+Adam and our proposed FNG. We showed that those NG methods helped to alleviate the strong conditioning problems associated to non-conjugate likelihood Gaussian process models. Particularly, the FNG achieved better local optima solutions with higher test performance rates than Adam, SGD and hybrid (NG+Adam).

- Finally in chapter 6, we modelled the citizens mobility in the Chinese city of Guangzhou, through the use of a ZIP likelihood in conjunction with GP priors, and provided comparative results to a Poisson likelihood. We derived an SVI framework that grants us to use two types of convolution process models in the

context of large datasets: 1. correlated chained GP with a convolution processes model, and 2. correlated chained GP with variational inducing kernels. Since, former works had not developed GP models based on CPM and VIKs for other type of likelihoods beyond a Gaussian, in this thesis, we derived equations that could be used for any type of likelihood. Paper (ii) shows experimental results where we found that all types of CCGP models in conjunction with a ZI-Poisson likelihood allow us to overcome the issues associated to zero-inflated data, outperforming the predictive capabilities of a CCGP model when based on a Poisson likelihood. Also, we showed that the CCGP models based on VIK or CPM generally reached a better predictive performance than the one based on an LMC.

Future work

- As a future work, it might be worth exploring the behaviour of the proposed FNG scheme over other type of GP models, for instance Deep GPs (Salimbeni et al., 2019). Likewise, it would be relevant to explore a scalable way to implement the method using a full covariance matrix $\mathbf{\Sigma}$ which can exploit full correlation between all hyper-parameters.

- The VO approach with penalisation relies on a Kullback-Leibler divergence as per Eq. (2.2), but one might research about the influence of a different family of divergences. For instance, the so-called α-divergence (Minka, 2005) presents a manifold of divergences indexed by $\alpha \in (-\infty, \infty)$, it might be worth to study how they motivate or induce exploration of the space of solutions for minimising an objective function.

- The different types of HetMOGP models introduced in this thesis could be extended by exploring the influence of assuming a full multivariate variational distribution $q(\mathbf{u})$, such an alternative might help to better quantify the uncertainty. Regarding the model training strategy, for instance, we could keep augmenting the outputs and test its performance. Indeed, it would be valuable to look into multi-output Gaussian processes and probe the output augmentation strategy in that context.

- Modelling multi-modal data is another venue for future work. One might potentially want to combine ideas from the work by Lázaro-Gredilla et al. (2012), with the HetMOGP model and the optimisation schemes proposed in this work.

Also, ideas for the model selection problem of the number Q of latent functions, like the ones based on Indian buffet processes (Guarnizo et al., 2015; Tong and Choi, 2019) can be further investigated in the particular context of MOGPs with Heterogeneous outputs.

- It might be worth to investigate the performance and possible improvements of the models described in this thesis when applying the work by Hensman et al. (2018), by combining the variational framework for sparse approximations and the spectral representation of Gaussian processes for modelling multiple heterogeneous outputs. Also, it might be worth to explore how the fully natural gradient scheme performs in such a new model structure.

- As a future work, we might study the behaviour of our GP models, based on CPM or VIKs, in conjunction with other types of likelihoods for tackling the problems described in chapter 6 of zero-inflated data. For instance, we could explore the performance of the ZINB likelihood, or the Tweedie likelihood from the exponential dispersion family. The latter can be particularly challenging given that its probability distribution needs to be evaluated using a series expansion due to not having an analytical solution.

- In the context of Multi-Output GPs, if we had a broader database that, apart from containing location information about the citizens mobility, additionally had the vehicles used for such mobility; we might implement our GP models to exploit the correlations between the types of transport vehicles. Likewise, we could explore the application of HetMOGPs with CPM for data imputation, i.e., predicting information in regions of the city where data is sensible to be lost due to failures in the mobile phone network that carries out the data collection.

Bibliography

Adam, V. (2017). Structured variational inference for coupled Gaussian processes. *arXiv preprint arXiv:1711.01131v2*.

Adam, V., Durrande, N., and John, S. (2018). Scalable GAM using sparse variational Gaussian processes. *arXiv preprint arXiv:1812.11106*.

Adam, V., Hensman, J., and Sahani, M. (2016). Scalable transformed additive signal decomposition by non-conjugate Gaussian process inference. In *26th IEEE International Workshop on Machine Learning for Signal Processing, MLSP*.

Álvarez, M. A. (2011). Convolved Gaussian process priors for multivariate regression with applications to dynamical systems. *PhD Thesis, University of Manchester*.

Álvarez, M. A. and Lawrence, N. D. (2009). Sparse convolved Gaussian processes for multi-output regression. In *Conference on Neural Information Processing Systems*, pages 57–64.

Álvarez, M. A. and Lawrence, N. D. (2011). Computationally efficient convolved multiple output Gaussian processes. *Journal of Machine Learning Research*, 12(41):1459–1500.

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

Álvarez, M. A., Rosasco, L., and Lawrence, N. D. (2012). Kernels for vector-valued functions: A review. *Foundations and Trends in Machine Learning*, 4(3):195–266.

Amari, S. (1998). Natural gradient works efficiently in learning. *Neural Computation*, 10(2):251–276.

Beckett, S., Jee, J., Ncube, T., Pompilus, S., Washington, Q., Singh, A., and Pal, N. (2014). Zero-inflated poisson (ZIP) distribution: parameter estimation and applications to model data from natural calamities. *Involve: A Journal of Mathematics*, 7(6):751–767.

Bertsekas, D. (1999). *Nonlinear Programming*. Athena Scientific.

BinTayyash, N., Georgaka, S., John, S., Ahmed, S., Boukouvalas, A., Hensman, J., and Rattray, M. (2020). Non-parametric modelling of temporal and spatial counts data from RNA-seq experiments. *bioRxiv*.

Bishop, C. M. (2006). *Pattern Recognition and Machine Learning*. Springer-Verlag New York, Inc.

Blei, D. M., Kucukelbir, A., and McAuliffe, J. D. (2017). Variational inference: A review for statisticians. *JASA*, 112(518):859–877.

Bonat, W. H., Jørgensen, B., Kokonendji, C. C., Hinde, J., and Demétrio, C. G. B. (2018). Extended Poisson–Tweedie: Properties and regression models for count data. *Stat. Modelling*, 18(1):24–49.

Boyle, P. and Frean, M. (2005). Dependent Gaussian processes. In *Conference on Neural Information Processing Systems*, pages 217–224. MIT Press.

Cao, Y. and Fleet, D. J. (2014). Generalized product of experts for automatic and principled fusion of Gaussian process predictions. *preprint arXiv:1410.7827v2*.

Chen, K., Yi, J., and Song, D. (2019). Gaussian processes model-based control of underactuated balance robots. In *2019 International Conference on Robotics and Automation (ICRA)*, pages 4458–4464.

Chong, E. and Zak, S. (2013). *An Introduction to Optimization*. Wiley Series in Discrete Mathematics and Optimization. Wiley.

Cohn, T. and Specia, L. (2013). Modelling annotator bias with multi-task Gaussian processes: An application to machine translation quality estimation. In *Association for Computational Linguistics (Volume 1: Long Papers)*, pages 32–42.

Conti, S., Gosling, J. P., Oakley, J. E., and O'Hagan, A. (2009). Gaussian process emulation of dynamic computer codes. *Biometrika*, 96(3):663–676.

Conti, S. and O'Hagan, A. (2010). Bayesian emulation of complex multi-output and dynamic computer models. *Journal of Statistical Planning and Inference*, 140(3):640 – 651.

Deisenroth, M. and Ng, J. W. (2015). Distributed Gaussian processes. In *Proceedings of the 32nd International Conference on Machine Learning*, pages 1481–1490.

Duvenaud, D. K., Nickisch, H., and Rasmussen, C. E. (2011). Additive Gaussian processes. In *Advances in Neural Information Processing Systems 24*, pages 226–234.

Guarnizo, C., Álvarez, M. A., and Orozco, A. A. (2015). Indian buffet process for model selection in latent force models. In *Iberoamerican Congress on Pattern Recognition*. Springer.

Gujarati, D. N. and Porter, D. C. (2009). *Basic econometrics*. McGraw-Hill, 5th ed edition.

Hegde, P., Heinonen, M., and Kaski, S. (2018). Variational zero-inflated Gaussian processes with sparse kernels. In *Conference on Uncertainty in Artificial Intelligence*, volume 1, pages 361–371.

Hensman, J., de G. Matthews, A. G., and Ghahramani, Z. (2015a). Scalable variational Gaussian process classification. In *International Conference on Artificial Intelligence and Statistics*.

Hensman, J., Durrande, N., and Solin, A. (2018). Variational fourier features for Gaussian processes. *Journal of Machine Learning Research*, 18(151):1–52.

Hensman, J., Fusi, N., and Lawrence, N. D. (2013). Gaussian processes for Big data. In *Proceedings of the Twenty-Ninth Conference on Uncertainty in Artificial Intelligence*, UAI'13, page 282–290, Arlington, Virginia, USA. AUAI Press.

Hensman, J., Matthews, A. G. d. G., Filippone, M., and Ghahramani, Z. (2015b). MCMC for variationally sparse Gaussian processes. In *Conference on Neural Information Processing Systems*, pages 1648–1656. MIT Press.

Higdon, D. (2002). Space and space-time modeling using process convolutions. In Anderson, C. W., Barnett, V., Chatwin, P. C., and El-Shaarawi, A. H., editors, *Quantitative Methods for Current Environmental Issues*, pages 37–56, London. Springer London.

Hoffman, M. D., Blei, D. M., Wang, C., and Paisley, J. (2013). Stochastic variational inference. *J. Mach. Learn. Res.*, 14(1):1303–1347.

Jin, S. and Andersson, B. (2020). A note on the accuracy of adaptive Gauss–Hermite quadrature. *Biometrika*, 107(3):737–744.

Jordan, M. I., Ghahramani, Z., Jaakkola, T. S., and Saul, L. K. (1999). An introduction to variational methods for graphical models. *Machine Learning*, 37(2):183–233.

Journel, A. G. and Huijbregts, C. J. (1979). Mining geostatistics. *Academic Press, London*.

Kahilakoski, O. (2011). Bayesian regression analysis of sickness absence. *Msc Thesis, Aalto University*.

Khan, M. E., Baque, P., Fleuret, F., and Fua, P. (2015). Kullback-Leibler proximal variational inference. In *Conference on Neural Information Processing Systems*, pages 3402–3410.

Khan, M. E. and Lin, W. (2017). Conjugate-computation variational inference: Converting variational inference in non-conjugate models to inferences in conjugate models. In *International Conference on Artificial Intelligence and Statistics*, pages 878–887.

Khan, M. E., Lin, W., Tangkaratt, V., Liu, Z., and Nielsen, D. (2017a). Variational adaptive-Newton method for explorative learning. *Conference on Neural Information Processing Systems, workshop on Advances in Approximate Bayesian Inference*.

Khan, M. E., Liu, Z., Tangkaratt, V., and Gal, Y. (2017b). Vprop: Variational inference using RMSprop. *Conference on Neural Information Processing Systems, workshop on Bayesian deep learning*.

Khan, M. E., Nielsen, D., Tangkaratt, V., Lin, W., Gal, Y., and Srivastava, A. (2018). Fast and scalable Bayesian deep learning by weight-perturbation in Adam. In *International Conference on Machine Learning*, pages 2616–2625.

Lázaro-Gredilla, M. and Figueiras-Vidal, A. (2009). Inter-domain Gaussian processes for sparse inference using inducing features. In *Advances in Neural Information Processing Systems*, volume 22. Curran Associates, Inc.

Lázaro-Gredilla, M., Van Vaerenbergh, S., and Lawrence, N. D. (2012). Overlapping mixtures of Gaussian processes for the data association problem. *Pattern Recogn.*, 45(4):1386–1395.

Long, J. and Freese, J. (2014). *Regression models for categorical dependent variables using Stata*. Stata Press, 3rd edition edition.

Lukusa, T. M., Lee, S.-M., and Li, C.-S. (2017). Review of zero-inflated models with missing data. *Current Research in Biostatistics*, 7(1):1–12.

Minka, T. (2005). Divergence measures and message passing. Technical Report MSR-TR-2005-173.

Moreno-Muñoz, P., Artés-Rodríguez, A., and Álvarez, M. A. (2018). Heterogeneous multi-output Gaussian process prediction. In *Conference on Neural Information Processing Systems*, pages 6712–6721.

Murphy, K. P. (2013). *Machine Learning: A Probabilistic Perspective*. MIT Press.

Nguyen, T. V. and Bonilla, E. V. (2014). Automated variational inference for Gaussian process models. In *Conference on Neural Information Processing Systems*, pages 1404–1412.

Osborne, M. A., Roberts, S. J., Rogers, A., Ramchurn, S. D., and Jennings, N. R. (2008a). Towards real-time information processing of sensor network data using computationally efficient multi-output Gaussian processes. In *Proceedings of the 7th International Conference on Information Processing in Sensor Networks*, page 109–120. IEEE Computer Society.

Osborne, M. A., Rogers, A., Ramchurn, S., Roberts, S. J., and Jennings, N. R. (2008b). Towards real-time information processing of sensor network data using computationally efficient multi-output Gaussian processes. In *International Conference on Information Processing in Sensor Networks*, pages 109–120.

Osborne, Michael A; Roberts, S. J. R. A. J. N. R. (2013). Real-time information processing of environmental sensor network data using Bayesian Gaussian processes. *ACM transactions on sensor networks*.

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

Quiñonero-Candela, J., Rasmussen, C. E., Sinz, F., Bousquet, O., and Schölkopf, B. (2006). Evaluating predictive uncertainty challenge. pages 1–27. Springer Berlin Heidelberg.

Raskutti, G. and Mukherjee, S. (2015). The information geometry of mirror descent. *IEEE Transactions on Information Theory*, 61(3):1451–1457.

Rasmussen, C. E. (2006). *Gaussian processes for machine learning*. MIT Press.

Raykar, V. C., Yu, S., Zhao, L. H., Valadez, G. H., Florin, C., Bogoni, L., and Moy, L. (2010). Learning from crowds. *Journal of Machine Learning Research*, 11(43):1297–1322.

Rodrigues, F., Henrickson, K., and Pereira, F. (2019). Multi-output Gaussian processes for crowdsourced traffic data imputation. *IEEE Transactions on Intelligent Transportation Systems*, 20(2):594 – 603.

Rodrigues, F., Pereira, F., and Ribeiro, B. (2013). Learning from multiple annotators: Distinguishing good from random labelers. *Pattern Recognition Letters*, 34(12):1428–1436.

Rodriguez-Deniz, H., Jenelius, E., and Villani, M. (2017). Urban network travel time prediction via online multi-output Gaussian process regression. In *20th IEEE International Conference on Intelligent Transportation Systems*, pages 1–6. IEEE.

Roemmele, E. S. (2019). A flexible zero-inflated Poisson regression model. *PhD Thesis, University of Kentucky*.

Rossi, S., Heinonen, M., Bonilla, E., Shen, Z., and Filippone, M. (2021). Sparse Gaussian processes revisited: Bayesian approaches to inducing-variable approximations. In Banerjee, A. and Fukumizu, K., editors, *Proceedings of The 24th International Conference on Artificial Intelligence and Statistics*, volume 130 of *Proceedings of Machine Learning Research*, pages 1837–1845. Proceedings of Machine Learning Research.

Ruiz, P., Morales-Álvarez, P., Molina, R., and Katsaggelos, A. K. (2019). Learning from crowds with variational Gaussian processes. *Pattern Recognition*, 88:298–311.

Salimbeni, H., Dutordoir, V., Hensman, J., and Deisenroth, M. (2019). Deep Gaussian processes with importance-weighted variational inference. In *International Conference on Machine Learning*, pages 5589–5598.

Salimbeni, H., Eleftheriadis, S., and Hensman, J. (2018). Natural gradients in practice: Non-conjugate variational inference in Gaussian process models. In *International Conference on Artificial Intelligence and Statistics*, pages 689–697.

Saul, A. D. (2016). Gaussian process based approaches for survival analysis. *PhD Thesis, University of Sheffield*.

Saul, A. D., Hensman, J., Vehtari, A., and Lawrence, N. D. (2016). Chained Gaussian processes. In *Proceedings of the Nineteenth International Workshop on Artificial Intelligence and Statistics*, volume 51, pages 1431–1440. Proceedings of Machine Learning Research.

Smyth, G. K. and Jørgensen, B. (2002). Fitting Tweedie's compound Poisson model to insurance claims data: Dispersion modelling. *ASTIN Bulletin*, 32(1):143–157.

Snelson, E. and Ghahramani, Z. (2006). Sparse Gaussian processes using pseudo-inputs. In *Conference on Neural Information Processing Systems*, pages 1257–1264.

Staines, J. and Barber, D. (2013). Optimization by variational bounding. In *ESANN*, pages 473–478.

Teh, Y. W., Seeger, M. W., and Jordan, M. I. (2005). Semiparametric latent factor models. In *International Conference on Artificial Intelligence and Statistics*, pages 333–340.

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

Tong, A. and Choi, J. (2019). Discovering latent covariance structures for multiple time series. volume 97, pages 6285–6294. Proceedings of Machine Learning Research.

Van der Wilk, M. (2018). Sparse Gaussian Process Approximations and Applications. *PhD Thesis, University of Cambridge*.

Walder, C., Kim, K. I., and Schölkopf, B. (2008). Sparse multiscale Gaussian process regression. In *Proceedings of the 25th International Conference on Machine Learning*, page 1112–1119. Association for Computing Machinery.

Wierstra, D., Schaul, T., Glasmachers, T., Sun, Y., Peters, J., and Schmidhuber, J. (2014). Natural evolution strategies. *J. Mach. Learn. Res.*, 15(1):949–980.

Zhao, J. and Sun, S. (2016). Variational dependent multi-output Gaussian process dynamical systems. *Journal of Machine Learning Research*, 17(1):4134–4169.

Zhu, T., Pimentel, M. A. F., Clifford, G. D., and Clifton, D. A. (2019). Unsupervised Bayesian inference to fuse biosignal sensory estimates for personalizing care. *IEEE Journal of Biomedical and Health Informatics*, 23(1):47–58.

Zuur, A., Ieno, E. N., Walker, N., Saveiliev, A. A., and Smith, G. M. (2009). *Mixed Effects Models and Extensions in Ecology with R*. Springer.

Appendices

Appendix A

Newton's Method Optimisation Example

We reproduced the same experiment used in chapters 2.1 and 2.2 for optimising:

$$\theta^{\star} = \arg\min_{\theta} g(\theta) = \arg\min_{\theta} 2\exp(-0.09\theta^2)\sin(4.5\theta). \tag{A.1}$$

without introducing any variational optimisation exploration over the variable θ. We used the Newton's method for minimising Eq. (A.1), with the same initial point $\theta = -3.0$. Figure A.1 presents the process of convergence of the Newton's method, this figure uses the same style used for Figures 2.1 and 2.2. Though here, from right to left the vertical red lines represent the occurrence of an iteration, being the furthest to the right the initial one. As it can be seen from Figure A.1, the optimisation carried out in the space of $g(\theta)$, with no exploration mechanism, converges to a poor local minima located between the interval $(-4, -3)$. From a variational optimisation perspective, we can analogously understand the Eq. (2.1), i.e., $\min_{\theta} g(\theta) \leq \tilde{\mathcal{L}} = \mathbb{E}_{q(\theta)}[g(\theta)]$ as: $g(\theta) = \tilde{\mathcal{L}} = \mathbb{E}_{q(\theta)}[g(\theta)]$, where this equality holds if the distribution is a Dirac's delta $q(\theta) = \delta(\theta - \mu)$ and $\mu = \theta$, in a nutshell, there is not exploration around $\mu = \theta$. Indeed, we can think of $q(\theta)$ as a Gaussian distribution with its variance collapsed to zero ($\sigma^2 = 0$), that is why the second sub-graph in Figure A.1 shows the black dots only moving along θ-axis (where $\theta = \mu$) while $\sigma = 0$, and the third sub-graph depicts the Dirac's delta distributions.

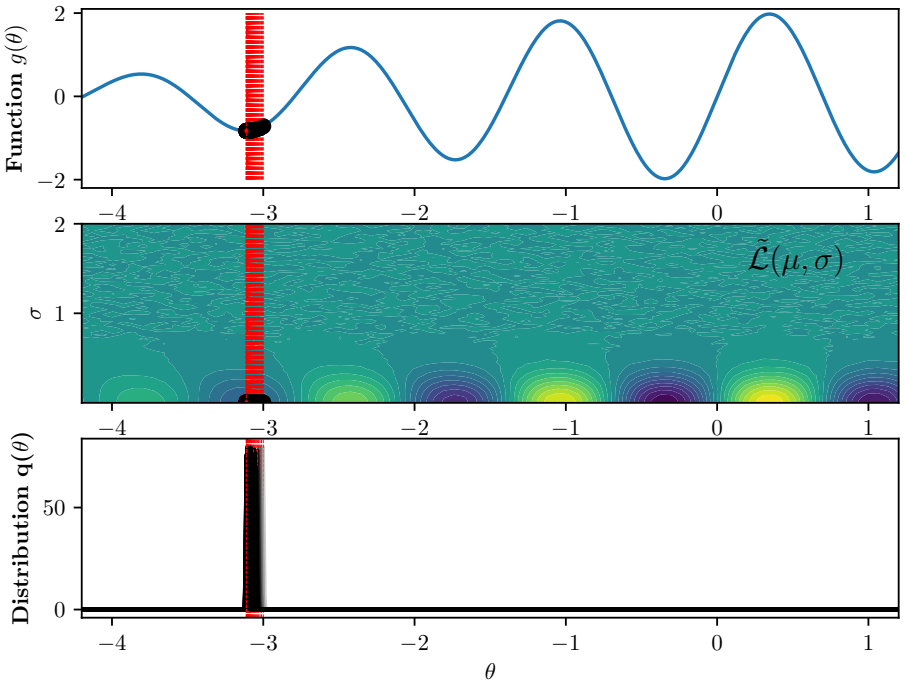


Figure A.1: *Using the Newton's method for optimising the multiple minima function* $g(\theta) = 2\exp(-0.09\theta^2)\sin(4.5\theta)$.

Appendix B

Influence of the Parameter λ During the Inference Process

With the aim to provide a criterion to select an initial value for the penalising distribution $p(\theta) = \mathcal{N}(\theta|0, \lambda^{-1})$ we have built an experiment as follows. We run the algorithm of Variational Optimisation with Penalisation over three types of objective functions that present multiple local minima. Below, we describe such functions in detail. We test $\lambda = \{0.1, 1, 10, 100\}$, and execute the algorithm 20 times for each value of λ with an initial value of $\theta_0 = -3$.

We use multi-local-minima functions of the form $g(\theta) = 2\exp(-0.09\theta^2)\sin(0.3w_1\theta)$, where the variable w_1 allows us to control the length-scale of the function. We test three scenarios: a large length-scale function with $w_1 = 7$, a moderate length-scale function with $w_1 = 15$ and a short length-scale function with $w_1 = 31$. Figures B.1, B.2 and B.3 show the landscape of each function. For $w_1 = 7$, we notice that the function presents at least three prominent local minima that are very separated one from another, at least in comparison to the functions for $w_1 = 15$ and $w_1 = 31$; for $w_1 = 15$, the function has at least six prominent local minima even closer than the case of $w_1 = 7$; and the function for $w_1 = 31$ presents multiple local minima which are very close to each other. Figures B.1, B.2 and B.3 show shaded circles in gray colour that represent the prominent local minima of each function. The y-axis of the figures to the left and to the right hand side is exactly the same, therefore each gray shaded circle that represents a local minima to the left hand side of the figures is aligned to its analogous value to the right hand side. The initial value $\theta_0 = -3$ is represented in all figures as an orange shaded circle.

On the right hand side of Figures B.1, B.2 and B.3, we can visualise the influence of

the parameter λ in the optimisation of the objective function. We notice from Figure B.1 that $\lambda = 100$ and $\lambda = 10$ tend to converge to suboptimal solutions that are not close to the local minima. Also from Figure B.2, we can see that the solutions provided when using $\lambda = 100$ are poor, being almost always the same; when using $\lambda = 10$, the algorithms reach a relevant local minima though it is not the best minimum of the function. From Figure B.3 we can notice that when using $\lambda = 100$ the algorithm tends to a region around $g(\theta) \approx -1.52$ achieving a better value of the function in comparison to the initial θ_0, though the convergence paths of the algorithm always target the same minima, thereby lacking of exploration for other possible solutions; when using $\lambda = 10$ the algorithm is prone to converge to a better local minima in comparison to the case of $\lambda = 100$. Also, for $\lambda = 10$ we can notice that the convergence paths of the algorithm not always tend to the same solution, but alternatively they go towards different local minima, showing broader explorative abilities.

On the other hand, for the case of $\lambda = 1$ and $\lambda = 0.1$, we can notice from Figure B.1 that the algorithms tend to find much better solutions in comparison to $\lambda = 10$ and $\lambda = 100$; for $\lambda = 0.1$ some solutions convergence to a local minima close to the initial value θ_0 and others to the global minimum, while for $\lambda = 1$ the solutions usually converged to the global minimum. Also, from Figure B.2 we can see that when using $\lambda = 0.1$ the algorithm converges to different types of possible solutions including the global minimum, though with a trend to converge to the solution closer to the initial θ_0; when using $\lambda = 1$ the algorithm tends to converge to either the global minimum or the prominent local minima at $g(\theta) \approx -1.5$. Figure B.3 shows that $\lambda = 0.1$ and $\lambda = 1$ reach diverse possible local minima solutions, including also the global minima among them; though the case of $\lambda = 1$ tends to arrive to better solutions than $\lambda = 0.1$.

In general, the values of $\lambda = 0.1$ and $\lambda = 1$ lead the algorithm to a variety of convergence paths, thus increasing the exploration of the space of solutions. Such an explorative behaviour is much higher than the one presented for $\lambda = 10$ and $\lambda = 100$; this either for functions that present landscapes with few notable local minima or functions with many local minima that are very close to each other. Apart of the experiments from Figures B.1, B.2 and B.3, we noticed from our manuscript's experiments that setting the penalisation parameter in the interval $0 \leq \lambda \leq 1$ led to broad exploration of the space of solution. Values very close to zero directly influence a higher explorative behaviour, though they can be very aggressive and generate possible numerical instabilities, particularly in the context of kernel methods and a high dimensionality in the input space. For instance, this type of strong behaviour can be noticed from Figure

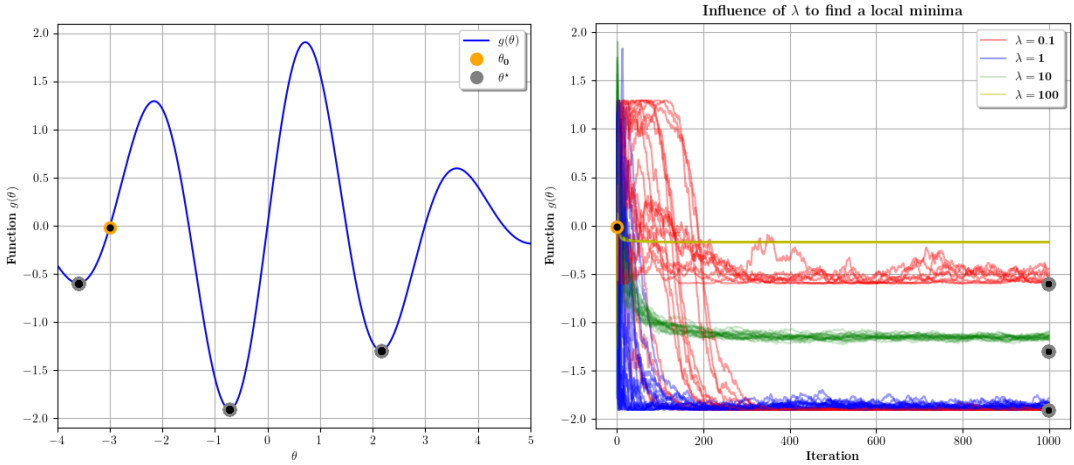


Figure B.1: *Influence of the precision parameter λ for optimising the function $g(\theta) = 2\exp(-0.09\theta^2)\sin((0.3 \times 7)\theta)$.*

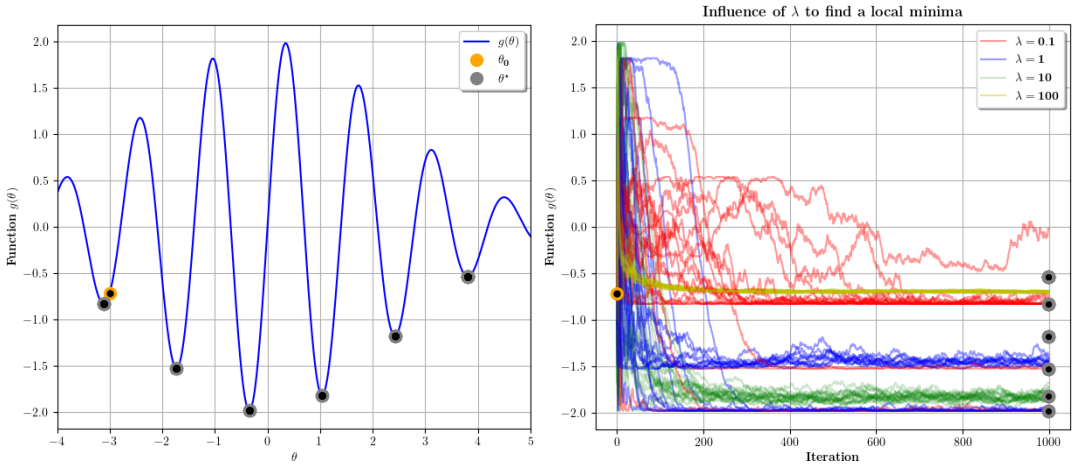


Figure B.2: *Influence of the precision parameter λ for optimising the function $g(\theta) = 2\exp(-0.09\theta^2)\sin((0.3 \times 15)\theta)$.*

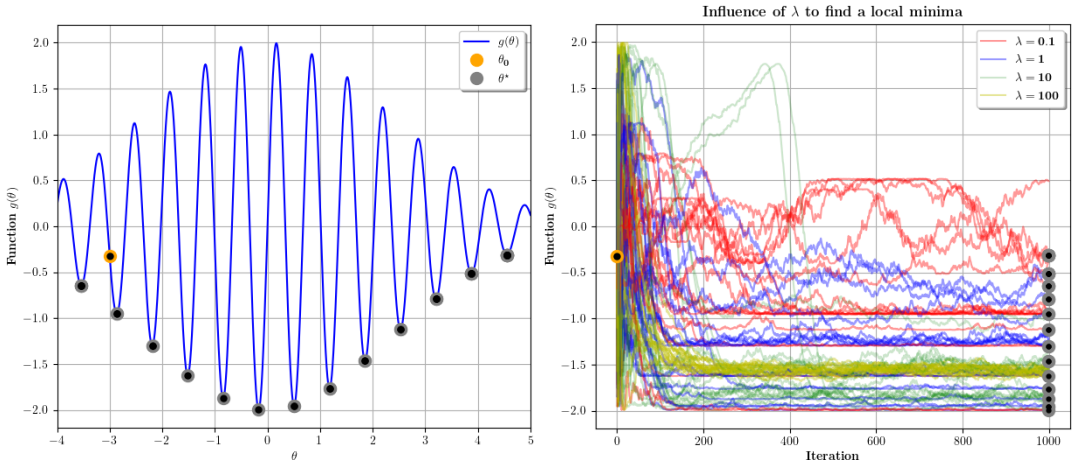


Figure B.3: *Influence of the precision parameter λ for optimising the function* $g(\theta) = 2\exp(-0.09\theta^2)\sin((0.3 \times 31)\theta)$.

B.2 and B.3 where for $\lambda = 0.1$ there are some aggressive changes in the path that caused the algorithm to arrive to poor solutions within the region $-0.5 < g(\theta) < 0.5$. Setting values close to one, or even one, can still present an explorative behaviour, but allowing a smoother convergence performance when approaching to a local minima.

Given that in practise we usually do not have any idea about the landscape of the objective functions that we are interested in optimising, then our suggestion for a practitioner is to initialise the penalisation distribution $p(\theta) = \mathcal{N}(\theta|0, \lambda^{-1})$ with $\lambda = 1.0$ since it presents a stable performance in diverse types of landscapes, or use even smaller values if more aggressive exploration is desired. Notice that the distribution $p(\theta)$ has its mean in zero ($\mathbb{E}[p(\theta)] = 0$); we do not assign any value different to zero, given that in general, when we optimise a function we do not have any idea about the locations of its local minima, then using a zero value for $p(\theta)$'s mean is a fair enough election.

Appendix C

From Mirror Descent to the Natural-Gradient

In this appendix, we provide additional details with regard to the relation between mirror descent and natural-gradient described in section 2.4. The mirror descent algorithm in the mean-parameters space of the distribution $q(\boldsymbol{\theta})$ bases on solving the following iterative sub-problems:

$$\boldsymbol{\eta}_{t+1} = \arg\min_{\boldsymbol{\eta}} \langle \boldsymbol{\eta}, \hat{\nabla}_{\boldsymbol{\eta}} \tilde{\mathcal{L}}_t \rangle + \frac{1}{\alpha_t} \mathbb{D}_{KL}(q(\boldsymbol{\theta}) || q_t(\boldsymbol{\theta})).$$

The intention of the above formulation is to exploit the parametrised distribution's structure by controlling its divergence w.r.t its older state $q_t(\boldsymbol{\theta})$. Thus, we can solve for the mirror descent algorithms setting to zero,

$$\begin{aligned} \langle \boldsymbol{\eta}, \hat{\nabla}_{\boldsymbol{\eta}} \tilde{\mathcal{L}}_t \rangle + \frac{1}{\alpha_t} \mathbb{D}_{KL}(q(\boldsymbol{\theta}) || q_t(\boldsymbol{\theta})) = 0 \\ \langle \boldsymbol{\eta}, \hat{\nabla}_{\boldsymbol{\eta}} \tilde{\mathcal{L}}_t \rangle + \frac{1}{\alpha_t} \mathbb{E}_{q(\boldsymbol{\theta})}[\log q(\boldsymbol{\theta}) - \log q_t(\boldsymbol{\theta})] = 0, \end{aligned}$$

since we can express the distribution $q(\boldsymbol{\theta})$ in the exponential-family form as follows:

$$q(\boldsymbol{\theta}) = h(\boldsymbol{\theta}) \exp\big(\langle \boldsymbol{\xi}, \phi(\boldsymbol{\theta}) \rangle - A(\boldsymbol{\xi})\big)$$

we can replace it in the above KL divergence as,

$$\begin{aligned} \langle \boldsymbol{\eta}, \hat{\nabla}_{\boldsymbol{\eta}} \tilde{\mathcal{L}}_t \rangle + \frac{1}{\alpha_t} \mathbb{E}_{q(\boldsymbol{\theta})}\big[\langle \boldsymbol{\xi}, \phi(\boldsymbol{\theta}) \rangle - A(\boldsymbol{\xi})\big] \\ - \frac{1}{\alpha_t} \mathbb{E}_{q(\boldsymbol{\theta})}\big[\langle \boldsymbol{\xi}_t, \phi(\boldsymbol{\theta}) \rangle - A(\boldsymbol{\xi}_t)\big] = 0 \end{aligned}$$

$$\langle \boldsymbol{\eta}, \hat{\nabla}_{\boldsymbol{\eta}} \tilde{\mathcal{L}}_t \rangle + \frac{1}{\alpha_t} \big[\langle \boldsymbol{\xi}, \mathbb{E}_{q(\boldsymbol{\theta})}[\phi(\boldsymbol{\theta})] \rangle - A(\boldsymbol{\xi}) \big] - \frac{1}{\alpha_t} \big[\langle \boldsymbol{\xi}_t, \mathbb{E}_{q(\boldsymbol{\theta})}[\phi(\boldsymbol{\theta})] \rangle - A(\boldsymbol{\xi}_t) \big] = 0,$$

given that $\mathbb{E}_{q(\boldsymbol{\theta})}[\phi(\boldsymbol{\theta})] = \boldsymbol{\eta}$ represent the mean-parameters, we can write again,

$$\langle \boldsymbol{\eta}, \hat{\nabla}_{\boldsymbol{\eta}} \tilde{\mathcal{L}}_t \rangle + \frac{1}{\alpha_t} \big[\langle \boldsymbol{\xi}, \boldsymbol{\eta} \rangle - A(\boldsymbol{\xi}) - \langle \boldsymbol{\xi}_t, \boldsymbol{\eta} \rangle + A(\boldsymbol{\xi}_t) \big] = 0$$

deriving w.r.t $\boldsymbol{\eta}$ we arrive to:

$$\hat{\nabla}_{\boldsymbol{\eta}} \tilde{\mathcal{L}}_t + \frac{1}{\alpha_t} \big[\boldsymbol{\xi} - \boldsymbol{\xi}_t \big] = 0.$$

Where the recursive update comes from making $\boldsymbol{\xi} := \boldsymbol{\xi}_{t+1}$:

$$\boldsymbol{\xi}_{t+1} = \boldsymbol{\xi}_t - \alpha_t \hat{\nabla}_{\boldsymbol{\eta}} \tilde{\mathcal{L}}_t$$

where $\hat{\nabla}_{\boldsymbol{\eta}} \tilde{\mathcal{L}}_t = \mathbf{F}^{-1} \hat{\nabla}_{\boldsymbol{\xi}} \tilde{\mathcal{L}}_t$ as per the work "The information geometry of mirror descent," (G. Raskutti and S. Mukherjee (2015)), where the authors provide a formal proof of such equivalence.

Appendix D

Bound Derivation for HetMOGP with Linear Model of Coregionalisation

In this appendix, we show how to derive the ELBO that appears in Eq. (4.5). We build the objective ELBO for the linear model of coregionalisation by assuming a variational distribution $q(\mathbf{f}, \mathbf{u}) = p(\mathbf{f}|\mathbf{u})q(\mathbf{u})$ as follows:

$$\begin{aligned}
\mathcal{L} &= \mathbb{E}_{q(\mathbf{f},\mathbf{u})}\Big[\log \frac{p(\mathbf{y}|\mathbf{f})p(\mathbf{f}|\mathbf{u})p(\mathbf{u})}{q(\mathbf{f},\mathbf{u})}\Big] \\
&= \mathbb{E}_{p(\mathbf{f}|\mathbf{u})q(\mathbf{u})}\Big[\log \frac{p(\mathbf{y}|\mathbf{f})\cancel{p(\mathbf{f}|\mathbf{u})}p(\mathbf{u})}{\cancel{p(\mathbf{f}|\mathbf{u})}q(\mathbf{u})}\Big] \\
&= \mathbb{E}_{p(\mathbf{f}|\mathbf{u})q(\mathbf{u})}\Big[\log p(\mathbf{y}|\mathbf{f})\Big] + \mathbb{E}_{q(\mathbf{u})}\Big[\log \frac{p(\mathbf{u})}{q(\mathbf{u})}\Big].
\end{aligned}$$

Notice that the right hand side term in the equation above does not depend on $p(\mathbf{f}|\mathbf{u})$ then only $q(\mathbf{u})$ remains. The left hand side term does not depend on $\mathbf{u}$ so we can integrate it out as follows:

$$q(\mathbf{f}_{d,j}) = \int p(\mathbf{f}_{d,j}|\mathbf{u})q(\mathbf{u})d\mathbf{u},$$

with this result, the marginal posterior over all the latent parameter functions is build as,

$$q(\mathbf{f}) = \prod_{d=1}^{D}\prod_{j=1}^{Jd} q(\mathbf{f}_{d,j}),$$

this way we can keep developing the ELBO,

$$\begin{aligned}
\mathcal{L} &= \mathbb{E}_{q(\mathbf{f})}\Big[\log p(\mathbf{y}|\mathbf{f})\Big] + \mathbb{E}_{q(\mathbf{u})}\Big[\log \prod_{q=1}^{Q} \frac{p(\mathbf{u}_q)}{q(\mathbf{u}_q)}\Big] \\
&= \mathbb{E}_{q(\mathbf{f})}\Big[\log \prod_{d=1}^{D}\prod_{n=1}^{N} p(y_{d,n}|\psi_{d,1}(\mathbf{x}_n), ..., \psi_{d,J_d}(\mathbf{x}_n))\Big] \\
&\qquad + \mathbb{E}_{q(\mathbf{u})}\Big[\log \prod_{q=1}^{Q} \frac{p(\mathbf{u}_q)}{q(\mathbf{u}_q)}\Big] \\
&= \sum_{d=1}^{D}\sum_{n=1}^{N} \mathbb{E}_{q(\mathbf{f})}\Big[\log p(y_{d,n}|\psi_{d,1}(\mathbf{x}_n), ..., \psi_{d,J_d}(\mathbf{x}_n))\Big] \\
&\qquad - \sum_{q=1}^{Q} \mathbb{D}_{\mathrm{KL}}(q(\mathbf{u}_q)||p(\mathbf{u}_q)).
\end{aligned}$$

We write again as a negative ELBO:

$$\tilde{\mathcal{L}} = \sum_{n=1}^{N}\sum_{d=1}^{D} \mathbb{E}_{q(\mathbf{f}_{d,1})\cdots q(\mathbf{f}_{d,J_d})}\left[g_{d,n}\right] + \sum_{q=1}^{Q} \mathbb{D}_{KL}\left(q(\mathbf{u}_q)\|p(\mathbf{u}_q)\right), \tag{D.1}$$

where $g_{d,n} = -\log p(y_{d,n}|\psi_{d,1}(\mathbf{x}_n), ..., \psi_{d,J_d}(\mathbf{x}_n))$ is the NLL function associated to each output.

Appendix E

Bound Derivation for HetMOGP with Convolution Processes

In this appendix, we show how to derive the ELBO that appears in Eq. (5.3). We derive the ELBO for the Heterogeneous MOGP with convolution processes, assuming a variational distribution $q(\mathbf{f}, \check{\mathbf{u}}) = p(\mathbf{f}|\check{\mathbf{u}})q(\check{\mathbf{u}})$ as follows:

$$\begin{aligned}\mathcal{L} &= \mathbb{E}_{q(\mathbf{f},\check{\mathbf{u}})}\Big[\log \frac{p(\mathbf{y}|\mathbf{f})p(\mathbf{f}|\check{\mathbf{u}})p(\check{\mathbf{u}})}{q(\mathbf{f},\check{\mathbf{u}})}\Big] \\ &= \mathbb{E}_{p(\mathbf{f}|\check{\mathbf{u}})q(\check{\mathbf{u}})}\Big[\log \frac{p(\mathbf{y}|\mathbf{f})\cancel{p(\mathbf{f}|\check{\mathbf{u}})}p(\check{\mathbf{u}})}{\cancel{p(\mathbf{f}|\check{\mathbf{u}})}q(\check{\mathbf{u}})}\Big] \\ &= \mathbb{E}_{p(\mathbf{f}|\check{\mathbf{u}})q(\check{\mathbf{u}})}\Big[\log p(\mathbf{y}|\mathbf{f})\Big] + \mathbb{E}_{q(\check{\mathbf{u}})}\Big[\log \frac{p(\check{\mathbf{u}})}{q(\check{\mathbf{u}})}\Big].\end{aligned}$$

Since the right hand side term in the equation above does not depend on $p(\mathbf{f}|\check{\mathbf{u}})$ then only $q(\check{\mathbf{u}})$ remains in the expectation. Regarding the left hand side term, $p(\mathbf{y}|\mathbf{f})$ does not depend on $\check{\mathbf{u}}$, so we can integrate out $q(\check{\mathbf{u}})$, as follows:

$$\begin{aligned}q(\mathbf{f}) &= \int p(\mathbf{f}|\check{\mathbf{u}})q(\check{\mathbf{u}})d\check{\mathbf{u}}. \\ &= \int \prod_{d=1}^{D}\prod_{j=1}^{Jd} p(\mathbf{f}_{d,j}|\check{\mathbf{u}}_{d,j})q(\check{\mathbf{u}}_{d,j})d\check{\mathbf{u}}_{d,j},\end{aligned}$$

Hence the marginal posterior over all the latent parameter functions is build as,

$$q(\mathbf{f}) = \prod_{d=1}^{D}\prod_{j=1}^{Jd} q(\mathbf{f}_{d,j}),$$

where each

$$q(\mathbf{f}_{d,j}) = \int p(\mathbf{f}_{d,j}|\check{\mathbf{u}}_{d,j})q(\check{\mathbf{u}}_{d,j})d\check{\mathbf{u}}_{d,j}.$$

This way we can keep developing the ELBO,

$$\begin{aligned}
\mathcal{L} &= \mathbb{E}_{q(\mathbf{f})}\Big[\log p(\mathbf{y}|\mathbf{f})\Big] + \mathbb{E}_{q(\check{\mathbf{u}})}\Big[\log \prod_{d=1}^{D}\prod_{j=1}^{J_d}\frac{p(\check{\mathbf{u}}_{d,j})}{q(\check{\mathbf{u}}_{d,j})}\Big] \\
&= \mathbb{E}_{q(\mathbf{f})}\Big[\log \prod_{d=1}^{D}\prod_{n=1}^{N} p(y_{d,n}|\psi_{d,1}(\mathbf{x}_n), ..., \psi_{d,J_d}(\mathbf{x}_n))\Big] \\
&\qquad + \mathbb{E}_{q(\check{\mathbf{u}})}\Big[\log \prod_{d=1}^{D}\prod_{j=1}^{J_d}\frac{p(\check{\mathbf{u}}_{d,j})}{q(\check{\mathbf{u}}_{d,j})}\Big] \\
&= \sum_{d=1}^{D}\sum_{n=1}^{N}\mathbb{E}_{q(\mathbf{f})}\Big[\log p(y_{d,n}|\psi_{d,1}(\mathbf{x}_n), ..., \psi_{d,J_d}(\mathbf{x}_n))\Big] \\
&\qquad - \sum_{d=1}^{D}\sum_{j=1}^{J_d}\mathbb{D}_{\mathrm{KL}}(q(\check{\mathbf{u}}_{d,j})||p(\check{\mathbf{u}}_{d,j})).
\end{aligned}$$

We write again as a negative ELBO:

$$\begin{aligned}
\tilde{\mathcal{L}} = &\sum_{n=1}^{N}\sum_{d=1}^{D}\mathbb{E}_{q(\mathbf{f}_{d,1})\cdots q(\mathbf{f}_{d,J_d})}\left[g_{d,n}\right] \\
&+ \sum_{d=1}^{D}\sum_{j=1}^{J_d}\mathbb{D}_{KL}\left(q(\check{\mathbf{u}}_{d,j})\|p(\check{\mathbf{u}}_{d,j})\right),
\end{aligned} \tag{E.1}$$

where $g_{d,n} = -\log p(y_{d,n}|\psi_{d,1}(\mathbf{x}_n), ..., \psi_{d,J_d}(\mathbf{x}_n))$ is the NLL function associated to each output.

Appendix F

Computing the Gradients w.r.t the Posterior' Parameters

The computation of the gradients $\hat{\nabla}_{\boldsymbol{\Sigma}}\tilde{\mathcal{F}}$ and $\hat{\nabla}_{\boldsymbol{\mu}}\tilde{\mathcal{F}}$ is directly influenced by the penalisation (or prior) distribution $p(\boldsymbol{\theta}) = \mathcal{N}(\boldsymbol{\theta}|\mathbf{0}, \lambda_1^{-1}\mathbf{I})$ with precision $\lambda_1 > 0$. Using the Gaussian identities, we can express the gradients as follows:

$$\hat{\nabla}_{\boldsymbol{\mu}}\tilde{\mathcal{F}} = \mathbb{E}_{q(\boldsymbol{\theta})}\left[\hat{\nabla}_{\boldsymbol{\theta}}\tilde{\mathcal{L}}\right] + \lambda_1\boldsymbol{\mu}$$
$$\hat{\nabla}_{\boldsymbol{\Sigma}}\tilde{\mathcal{F}} = \frac{1}{2}\mathbb{E}_{q(\boldsymbol{\theta})}\left[\hat{\nabla}^2_{\boldsymbol{\theta}\boldsymbol{\theta}}\tilde{\mathcal{L}}\right] + \frac{1}{2}\lambda_1\mathbf{I} - \frac{1}{2}\boldsymbol{\Sigma}^{-1}.$$

The other gradients $\hat{\nabla}_{\mathbf{m}_{(\cdot)}}\tilde{\mathcal{F}} = \mathbb{E}_{q(\boldsymbol{\theta})}[\hat{\nabla}_{\mathbf{m}_{(\cdot)}}\tilde{\mathcal{L}}]$ and $\hat{\nabla}_{\mathbf{V}_{(\cdot)}}\tilde{\mathcal{F}} = \mathbb{E}_{q(\boldsymbol{\theta})}[\hat{\nabla}_{\mathbf{V}_{(\cdot)}}\tilde{\mathcal{L}}]$ depend on the inner gradients $\hat{\nabla}_{\mathbf{m}}\tilde{\mathcal{L}}$ and $\hat{\nabla}_{\mathbf{V}}\tilde{\mathcal{L}}$ of the negative ELBO in Eq. (4.5) for LMC, or in Eq. (5.3) for CPM.

F.1 Particular Gradients for Linear Model of Coregionalisation

Taking the derivative of $\tilde{\mathcal{L}}$ for the LMC w.r.t each parameter $\mathbf{m}_q$ and $\mathbf{V}_q$ we arrive to,

$$\hat{\nabla}_{\mathbf{m}_q}\tilde{\mathcal{L}} = \sum_{d=1}^{D}\sum_{j=1}^{J_d}\mathbf{A}^{\top}_{\mathbf{f}_{d,j}\mathbf{u}_q}\mathbf{g}_{\mathbf{m}d,j} + \mathbf{K}^{-1}_{\mathbf{u}_q\mathbf{u}_q}\mathbf{m}_q, \tag{F.1}$$

$$\hat{\nabla}_{\mathbf{V}_q}\tilde{\mathcal{L}} = \sum_{d=1}^{D}\sum_{j=1}^{J_d}\mathbf{A}^{\top}_{\mathbf{f}_{d,j}\mathbf{u}_q}\mathrm{diag}(\mathbf{g}_{\mathbf{v}d,j})\mathbf{A}_{\mathbf{f}_{d,j}\mathbf{u}_q} - \frac{1}{2}\left[\mathbf{V}_q^{-1} - \mathbf{K}^{-1}_{\mathbf{u}_q\mathbf{u}_q}\right], \tag{F.2}$$

where $\mathbf{A}_{\mathbf{f}_{d,j}\mathbf{u}_q} = \mathbf{K}_{\mathbf{f}_{d,j}\mathbf{u}_q}\mathbf{K}_{\mathbf{u}_q\mathbf{u}_q}^{-1}$, the vector, $\mathbf{g}_{\mathbf{m}d,j} \in \mathbb{R}^{N\times 1}$, has entries computed with $\mathbb{E}_{q_{f_{d,1}(\mathbf{x}_n)},\ldots,q_{f_{d,Jd}(\mathbf{x}_n)}}[\nabla_{f_{d,j}(\mathbf{x}_n)} g_{d,n}]$, the vector $\mathbf{g}_{\mathbf{v}d,j} \in \mathbb{R}^{N\times 1}$ has entries calculated using the expectation, $\frac{1}{2}\mathbb{E}_{q_{f_{d,1}(\mathbf{x}_n)},\ldots,q_{f_{d,J_d}(\mathbf{x}_n)}}[\nabla^2_{f_{d,j}(\mathbf{x}_n)f_{d,j}(\mathbf{x}_n)} g_{d,n}]$, and $\text{diag}(\mathbf{g}_{\mathbf{v}d,j})$ is a new matrix with the elements of $\mathbf{g}_{\mathbf{v}d,j}$ on its diagonal. Notice that each distribution $q_{f_{d,j,n}}$ represents the n-th marginal of each distribution $q_{\mathbf{f}_{d,j}}$. The above equations allow us to use mini-batches at each iteration of the inference process. Then, instead of using all data observations N, we randomly sample a mini-batch $\mathbf{X}_B \in \mathbb{R}^{B\times P}$ and $\mathbf{y}_B \in \mathbb{R}^{B\times D}$ from the dataset $D = \{\mathbf{X}, \mathbf{y}\}$, here B accounts for the mini-batch size. We simply construct: the matrix $\mathbf{A}_{\mathbf{f}_{d,j}\mathbf{u}_q}$ which becomes $\in \mathbb{R}^{B\times M}$, and the vectors $\mathbf{g}_{\mathbf{m}d,j}$ and $\mathbf{g}_{\mathbf{v}d,j}$ which become $\in \mathbb{R}^{B\times 1}$. Then we scale the first term to the right hand side of Eq. (F.1) and Eq. (F.2) by a factor of N/B. We refer to $D_B = \{\mathbf{X}_B, \mathbf{y}_B\}$ as the mini-batch data collection. It is worth noticing that for the case of the CCGP model we can simply treat the number of outputs as $D = 1$.

F.2 Particular Gradients for Convolution Processes Model

Taking the derivative of $\tilde{\mathcal{L}}$ for the CPM w.r.t each parameter $\mathbf{m}_{d,j}$ and $\mathbf{V}_{d,j}$ we find that,

$$\begin{aligned}
\hat{\nabla}_{\mathbf{m}_{d,j}}\tilde{\mathcal{L}} &= \mathbf{A}^{\top}_{\mathbf{f}_{d,j}\check{\mathbf{u}}_{d,j}}\check{\mathbf{g}}_{\mathbf{m}_{d,j}} + \mathbf{K}^{-1}_{\check{\mathbf{u}}_{d,j}\check{\mathbf{u}}_{d,j}}\mathbf{m}_{d,j},\\
\hat{\nabla}_{\mathbf{V}_{d,j}}\tilde{\mathcal{L}} &= \mathbf{A}^{\top}_{\mathbf{f}_{d,j}\check{\mathbf{u}}_{d,j}}\text{diag}(\check{\mathbf{g}}_{\mathbf{v}_{d,j}})\mathbf{A}_{\mathbf{f}_{d,j}\check{\mathbf{u}}_{d,j}}\\
&\qquad - \frac{1}{2}\big[\mathbf{V}^{-1}_{d,j} - \mathbf{K}^{-1}_{\check{\mathbf{u}}_{d,j}\check{\mathbf{u}}_{d,j}}\big],
\end{aligned}$$

where $\mathbf{A}_{\mathbf{f}_{d,j}\check{\mathbf{u}}_{d,j}} = \mathbf{K}_{\mathbf{f}_{d,j}\check{\mathbf{u}}_{d,j}}\mathbf{K}^{-1}_{\check{\mathbf{u}}_{d,j}\check{\mathbf{u}}_{d,j}}$, the vector $\check{\mathbf{g}}_{\mathbf{m}_{d,j}} \in \mathbb{R}^{N\times 1}$ has entries computed with $\mathbb{E}_{q_{f_{d,1}(\mathbf{x}_n)},\ldots,q_{f_{d,Jd}(\mathbf{x}_n)}}[\nabla_{f_{d,j}(\mathbf{x}_n)} g_{d,n}]$, the vector $\check{\mathbf{g}}_{\mathbf{v}_{d,j}} \in \mathbb{R}^{N\times 1}$ has entries calculated using $\frac{1}{2}\mathbb{E}_{q_{f_{d,1}(\mathbf{x}_n)},\ldots,q_{f_{d,J_d}(\mathbf{x}_n)}}[\nabla^2_{f_{d,j}(\mathbf{x}_n)f_{d,j}(\mathbf{x}_n)} g_{d,n}]$, and $\text{diag}(\check{\mathbf{g}}_{\mathbf{v}_{d,j}})$ is a new matrix with the elements of $\check{\mathbf{g}}_{\mathbf{v}_{d,j}}$ on its diagonal. Notice that each distribution $q_{f_{d,j}(\mathbf{x}_n)}$ represents the n-th marginal of each distribution $q(\mathbf{f}_{d,j})$.

Appendix G

FNG Algorithm

Algorithm 1 shows a pseudo-code implementation of the proposed method in section (3.6.3) and (4.4.3) for CCGP and HetMOGP models based on LMC, respectively; and of the method in section (5.4.3) for HetMOGP with CPM when applying inducing variables $\check{u}_{d,j}(\cdot)$. Here, we use $\boldsymbol{\Sigma}^{(\cdot)}$ and $\boldsymbol{\mu}^{(\cdot)}$ for referring to either $\boldsymbol{\Sigma}$ and $\boldsymbol{\mu}$ in CCGP; or $\boldsymbol{\Sigma}^{D}$ and $\boldsymbol{\mu}^{D}$ in HetMOGP with LMC; or $\boldsymbol{\Sigma}^{C}$ and $\boldsymbol{\mu}^{C}$ in HetMOGP with CPM. Likewise, we use $\mathbf{V}_{(\cdot)}$ and $\mathbf{m}_{(\cdot)}$ for referring to $\mathbf{V}_q$ and $\mathbf{m}_q$ in CCGP and HetMOGP models with LMC, or to $\mathbf{V}_{d,j}$ and $\mathbf{m}_{d,j}$ in HetMOGP with CPM. In practice, we found useful to update the parameters $\boldsymbol{\mu}_{t+1}^{(\cdot)}$ using $\sqrt{\mathbf{p}_t^{(\cdot)}}$ and $\sqrt{\mathbf{p}_{t+1}^{(\cdot)}}$ instead of $\mathbf{p}_t^{(\cdot)}$ and $\mathbf{p}_{t+1}^{(\cdot)}$, for improving the method's convergence.

Algorithm 1 Fully Natural Gradient Algorithm

Input: $\alpha_t, \beta_t, \gamma_t, \nu_t, \lambda_1$

Output: $\boldsymbol{\Sigma}_{t+1}^{(\cdot)}, \boldsymbol{\mu}_{t+1}^{(\cdot)}, \mathbf{V}_{(\cdot),t+1}, \mathbf{m}_{(\cdot),t+1}$

set $t = 1$
while Not Converged **do**
  sample $\boldsymbol{\theta}_t \sim q(\boldsymbol{\theta}|\boldsymbol{\mu}_t^{(\cdot)}, \boldsymbol{\Sigma}_t^{(\cdot)})$
  randomly sample a mini-batch D_B
  $\mathbb{E}_{q(\boldsymbol{\theta})}\big[\hat{\nabla}_{\boldsymbol{\theta}}\tilde{\mathcal{L}}\big]$ and $\mathbb{E}_{q(\boldsymbol{\theta})}\big[\hat{\nabla}_{\boldsymbol{\theta}}\tilde{\mathcal{L}} \circ \hat{\nabla}_{\boldsymbol{\theta}}\tilde{\mathcal{L}}\big]$ using samples $\boldsymbol{\theta}_t$
  update $\mathbf{p}_{t+1}$ and $\boldsymbol{\mu}_{t+1}$
  compute $\hat{\nabla}_{\mathbf{m}_{(\cdot)}}\tilde{\mathcal{F}}$ and $\hat{\nabla}_{\mathbf{V}_{(\cdot)}}\tilde{\mathcal{F}}$
  update $\mathbf{V}_{(\cdot),t+1}$ and $\mathbf{m}_{(\cdot),t+1}$
  $\boldsymbol{\Sigma}_{t+1}^{(\cdot)} = \text{diag}\left((\mathbf{p}_{t+1}^{(\cdot)} + \lambda_1 \mathbf{1})^{-1}\right)$
  $t = t + 1$
end while

Appendix H

Maximum a Posteriori in the Context of Variational Inference

In context of Bayesian inference, posterior distribution $p(\boldsymbol{\theta}|\mathbf{X})$ is proportional to the likelihood $p(\mathbf{X}|\boldsymbol{\theta})$ times the prior $p(\boldsymbol{\theta})$, i.e., $p(\boldsymbol{\theta}|\mathbf{X}) \propto p(\mathbf{X}|\boldsymbol{\theta})p(\boldsymbol{\theta})$. Although, if the likelihood and prior are non-conjugate distributions, it is necessary to approximate the posterior, for instance using variational inference. In this context of variational inference, we do not have access to the true posterior, but to the approximate posterior $q(\boldsymbol{\theta})$, which it is optimised by maximising the ELBO,

$$\mathcal{L} = \mathbb{E}_{q(\boldsymbol{\theta})}\big[\log p(\mathbf{X}|\boldsymbol{\theta})\big] - \mathbb{D}_{KL}(q(\boldsymbol{\theta})||p(\boldsymbol{\theta})) \leq \log p(\mathbf{X}).$$

Notice that if we are only interested in a point estimate of the parameter $\boldsymbol{\theta}$ of the Log Likelihood function, then a feasible solution for the parameter is $\boldsymbol{\theta}^{\star} = \mathbb{E}_{q(\boldsymbol{\theta})}[\boldsymbol{\theta}] = \boldsymbol{\mu}$, where $q(\boldsymbol{\theta}) := q(\boldsymbol{\theta}|\boldsymbol{\mu}, \boldsymbol{\Sigma})$. This corresponds to the MAP solution due to the fact that,

$$\boldsymbol{\theta}_{\text{MAP}} = \arg\max_{\boldsymbol{\theta}} p(\boldsymbol{\theta}|\mathbf{X}),$$

where $p(\boldsymbol{\theta}|\mathbf{X})$ represents the true posterior. Since in the context of variational inference, we only have access to an approximate free parametrised posterior $p(\boldsymbol{\theta}|\mathbf{X}) \approx q(\boldsymbol{\theta}|\boldsymbol{\mu}, \boldsymbol{\Sigma})$, therefore the equation above implies that,

$$\boldsymbol{\theta}_{\text{MAP}} = \arg\max_{\boldsymbol{\theta}} q(\boldsymbol{\theta}|\boldsymbol{\mu}, \boldsymbol{\Sigma}),$$

and it is clear that the maximum of the distribution $q(\boldsymbol{\theta}|\boldsymbol{\mu}, \boldsymbol{\Sigma})$ is located at its mean, thereby $\boldsymbol{\theta}_{\text{MAP}} = \boldsymbol{\mu}$.

Appendix I

Rule of Thumb to Select the number Q of Latent Functions $u_q(\cdot)$

Some research have particularly focused on the topic of selection of the number Q of latent functions $u_q(\cdot)$, for instance the works: "Multi-task Gaussian Process Learning of Robot Inverse Dynamics" (Kian Ming A. et al., 2008); "Indian Buffet process for model selection in latent force models" (Guarnizo et al., 2015) and "Discovering Latent Covariance Structures for Multiple Time Series" (A. Tong and J. Choi, 2019). Kian Ming A. et al. (2008) uses a Bayesian Information Criterion (BIC) for setting Q, whereas both Guarnizo et al. (2015) and A. Tong and J. Choi (2019) use Indian buffet process prior together with Bayesian inference for selecting Q. Although, selecting such a parameter is still an open question for research.

Since the selection of such a number Q is still an open problem in the literature, and the main focus of our manuscript aims to target the problem of inference regardless of the Q value; we proposed a rule of thumb for selecting the value of Q, useful from the point of view of the practitioner. In the Heterogeneous MOGP, the total number of latent parameter functions, $f_{d,j}(\mathbf{x})$, is equal to $J = \sum_{d=1}^{D} J_d$, where J_d is the number of LPFs per likelihood. To select Q, we apply the rule of thumb described below:

1. If the number of outputs is less or equal than five, then set the number of latent functions $Q = J$, i.e., if $D <= 5$ set $Q = J$.

2. If the number of outputs is higher than five, then set the number of latent functions $Q = 3$, i.e., if $D > 5$ set $Q = 3$.

In order to compute $J = \sum_{d=1}^{D} J_d$, the look up table below indicates the number of LPFs J_d as per each specific type of likelihood used for each output:

Likelihood Type	**Latent Functions $f_{d,j}$ (J_d)**
Gaussian	1
Poisson	1
Bernoulli	1
HetGaussian	2
Beta	2
Gamma	2

Thus, using the above rule of thumb we have selected the Q value for our experiments as follows: for the human data, we have three outputs with different likelihoods, Bernoulli, Heteroscedastic-Gaussian and Beta. As per the Table above, the Bernoulli likelihood requires one LPF; the Heteroscedastic-Gaussian likelihood requires two LPFs, and the Beta likelihood also requires two LPFs. This gives a total of $Q = 5$. For the London data, we have two outputs with likelihoods: Bernoulli with one LPF and Heteroscedastic-Gaussian with two LPFs, all for a total of $Q = 3$. For the Naval data, we have two outputs with likelihoods: Beta that requires two LPFs, and Gamma also requires two LPFs. This give a total of $Q = 4$. For any other experiment with a number of outputs $D > 5$ we have selected $Q = 3$.

We opt for this rule of thumb as a way to allow the HetMOGP model to have a high flexibility for modelling the data. The higher the number Q of latent function $u_q(\cdot)$ is, the higher the flexibility will be to model possibly high frequency trendings that can be present in the data observations. Although, setting a high number Q goes in detriment to the computational complexity of the model and overloads making future predictions; this is a reason why our rule of thumb sets $Q = 3$ when having more than five outputs. Also, setting $Q = 3$ allows us to at least account for low, medium and high length-scale resolutions for modelling a dataset with a very high number of outputs.

Appendix J

Additional Information of Datasets and Experiments Setting

The datasets used in our experiments were taken from the following web-pages:

- The HUMAN is captured using EB2 *app*, visit https://www.eb2.tech/
- For information about LONDON dataset visit: https://www.gov.uk/government/collections/price-paid-data
- For information about NAVAL dataset visit http://archive.ics.uci.edu/ml/datasets
- For information about SARCOS dataset see http://www.gaussianprocess.org/gpml/data/
- See http://mocap.cs.cmu.edu/subjects.php for MOCAP dataset, subject 7 refers to MOCAP7 and subject 9 refers to MOCAP9.
- Visit https://data.gov.uk/dataset/208c0e7b-353f-4e2d-8b7a-1a7118467acc/gb-road-traffic-counts for information about TRAFFIC dataset.

J.1 Additional Analysis per Output over LONDON and NAVAL datasets

For the LONDON dataset, Figure J.1 shows that Adam converges to a richer minimum of the NELBO than SGD. Moreover, the NLPD for Adam is, on average, better than the SGD for both HetGaussian and Bernoulli outputs. Particularly, Adam presents for the Bernoulli output few "outliers" under its boxes that suggest it can find sporadically

rich local optima, but its general trend was to provide poor solutions for that specific output in contrast to the HetGaussian output. The HYB and FNG arrive to a very similar value of the NELBO, both being better than Adam and SGD. HYB and FNG methods attain akin NLPD metrics for the HetGaussian output, though our method shows smaller boxes being more confident along iterations. Both methods present large variances for the Bernoulli output, but the average and median trend of our approach is much better, being more robust to the initialisation than HYB method. The NLPD performance for the NAVAL dataset shows in Figure J.2 that the SGD method cannot make progress. We tried to set a bigger step-size, but usually increasing it derived in numerical problems due to ill-conditioning. The methods Adam and HYB show almost the same behaviour along the NELBO optimisation, in fact the NLPD boxes for the Beta and Gamma outputs look quite similar for both methods. The difference of performance can be noticed for the Beta output, where at the end, HYB method becomes more confident reducing its variance. Our FNG method ends up with a slightly upper NLPD solution in the Gamma output in comparison to Adam and HYB, but being more confident showing a smaller spread in the box-plot across iterations. For the Gamma output, FNG shows at the end some "outliers" under the NLPD boxes, accounting for sporadic convergence to strong solutions. For the Beta distribution, our method obtains a better solution with the finest NLPD in comparison to SGD, HYB and Adam.

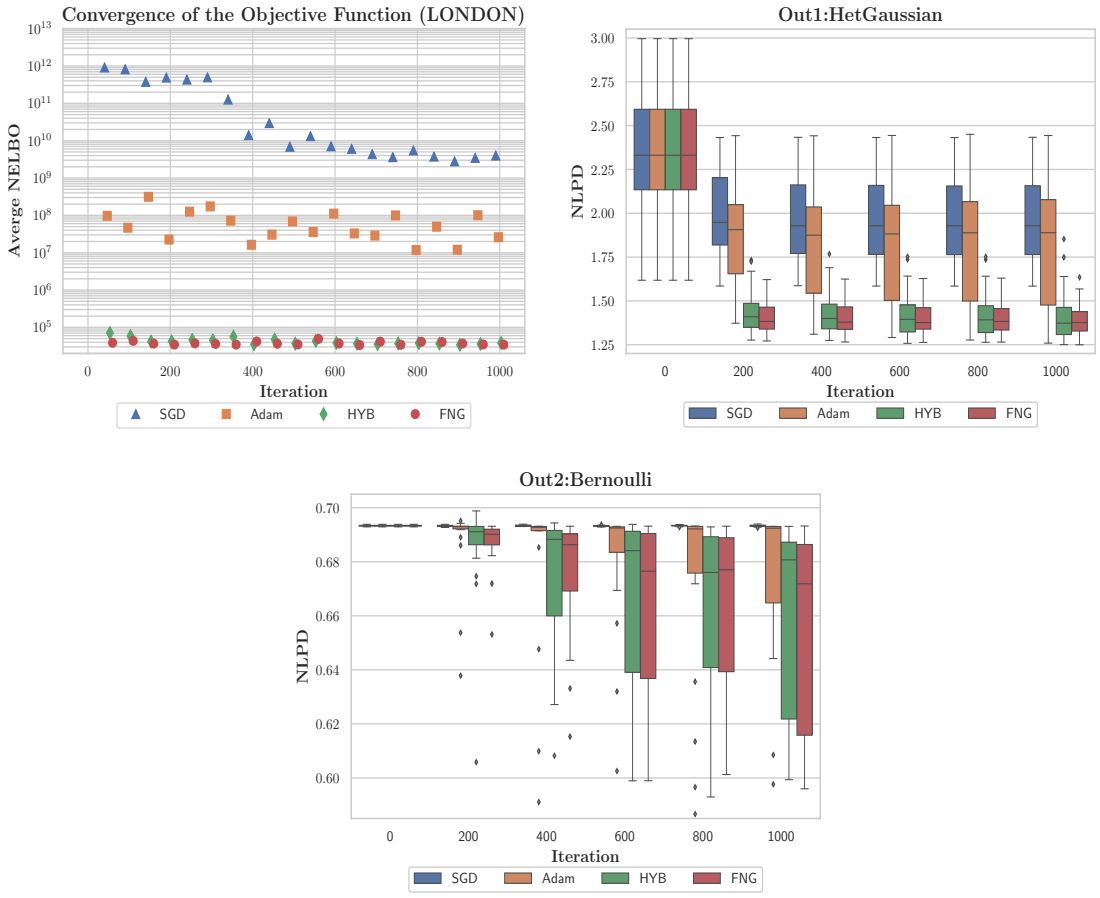


Figure J.1: *Performance of the diverse inference methods on the LONDON dataset using 20 different initialisations over HetMOGP with LMC. The left sub-figure shows the average NELBO convergence of each method. The other sub-figures show the box-plot trending of the NLPD over the test set for each output. The box-plots at each iteration follow the legend's order from left to right: SGD, Adam, HYB and FNG. The isolated diamonds that appear in the outputs' graphs represent "outliers".*

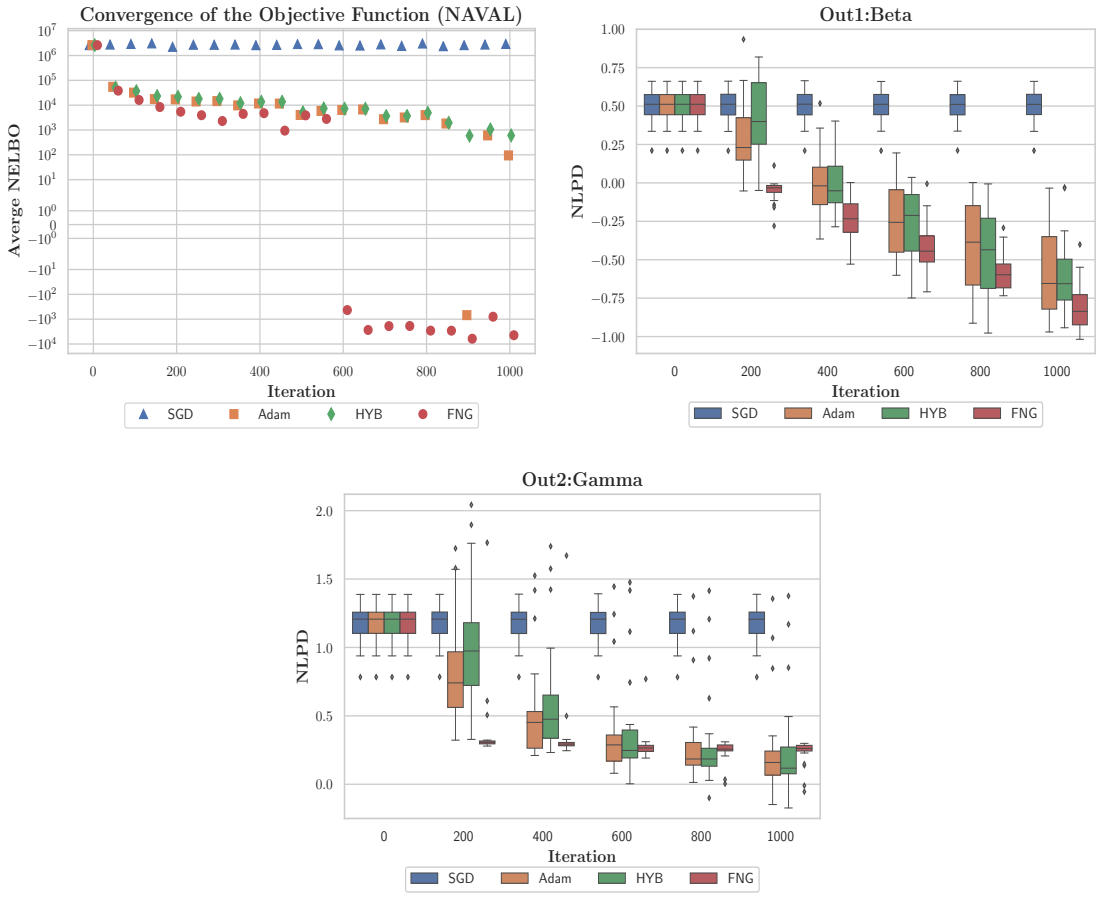


Figure J.2: *Performance of the diverse inference methods on the NAVAL dataset using 20 different initialisations over HetMOGP with LMC. The left sub-figure shows the average NELBO convergence of each method. The other sub-figures show the box-plot trending of the NLPD over the test set for each output. The box-plots at each iteration follow the legend's order from left to right: SGD, Adam, HYB and FNG. The isolated diamonds that appear in the output graphs represents "outliers".*

Appendix K

Paper i: A Fully Natural Gradient Scheme for Improving Inference of the Heterogeneous Multi-Output Gaussian Process Model

Paper accepted in the Journal IEEE Transactions on Neural Networks and Learning Systems.

A Fully Natural Gradient Scheme for Improving Inference of the Heterogeneous Multi-Output Gaussian Process Model

Juan-José Giraldo, and Mauricio A. Álvarez

***Abstract*—A recent novel extension of multi-output Gaussian processes handles heterogeneous outputs assuming that each output has its own likelihood function. It uses a vector-valued Gaussian process prior to jointly model all likelihoods' parameters as latent functions drawn from a Gaussian process with a linear model of coregionalisation covariance. By means of an inducing points framework, the model is able to obtain tractable variational bounds amenable to stochastic variational inference. Nonetheless, the strong conditioning between the variational parameters and the hyper-parameters burdens the adaptive gradient optimisation methods used in the original approach. To overcome this issue we borrow ideas from variational optimisation introducing an exploratory distribution over the hyper-parameters, allowing inference together with the posterior's variational parameters through a fully natural gradient optimisation scheme. Furthermore, in this work we introduce an extension of the heterogeneous multi-output model, where its latent functions are drawn from convolution processes. We show that our optimisation scheme can achieve better local optima solutions with higher test performance rates than adaptive gradient methods, this for both the linear model of coregionalisation and the convolution processes model. We also show how to make the convolutional model scalable by means of stochastic variational inference and how to optimise it through a fully natural gradient scheme. We compare the performance of the different methods over toy and real databases.**

***Index Terms*—Natural Gradient, Multi-Output Gaussian Process, Heterogeneous Outputs, Convolution Processes, Variational Optimisation.**

I. Introduction

A MULTI-OUTPUT Gaussian Processes (MOGP) model generalises the Gaussian Process (GP) model by exploiting correlations not only in the input space, but also in the output space [1]. Major research about MOGP models has focused on finding proper definitions of a cross-covariance function between the multiple outputs [2], [3]. Nevertheless few works have been concerned about targeting the issue that those outputs not necessarily follow the same statistical data type. To address that regard, a recent approach known as the Heterogeneous Multi-Output Gaussian Process (HetMOGP) model extents the MOGP application [4] to any arbitrary combination of D likelihood distributions over the output observations [5]. The HetMOGP jointly models all likelihoods' parameters as latent functions drawn from a Gaussian process with a linear model of coregionalisation (LMC) covariance. It can be seen as a generalisation of a Chained GP [6] for multiple correlated output functions of an heterogeneous nature. The HetMOGP's scalability bases on the schemes of variational inducing variables for single-output GPs [7]. This scheme relies on the idea of augmenting the GP prior probability space, through the inclusion of a so-called set of inducing points that change the full GP covariance by a low-rank approximation [8], [9]. Such inducing points help reducing significantly the MOGP's computational costs from $\mathcal{O}(D^3N^3)$ to $\mathcal{O}(DNM^2)$ and storage from $\mathcal{O}(D^2N^2)$ to $\mathcal{O}(DNM)$, where N, D and $M \ll N$ represent the number of data observations, outputs and inducing points, respectively [10], [4].

The adequate performance of a variational GP model depends on a proper optimisation process able to find rich local optima solutions for maximising a bound to the marginal likelihood. Variational GP models generally suffer from strong conditioning between the variational posterior distribution, the multiple hyper-parameters of the GP prior and the inducing points [11]. In particular, the HetMOGP model is built upon a linear combinations of Q latent functions, where each latent function demands a treatment based on the inducing variables framework. On this model then, such strong conditionings are enhanced even more due to the dependence of inducing points per underlying latent function, and the presence of additional linear combination coefficients. Since the model is extremely sensitive to any small change on any of those variables, stochastic gradient updates in combination with adaptive gradient methods (AGMs, e.g. Adam) tend to drive the optimisation to poor local minima.

With the purpose to overcome the optimisation problems present in variational GP models, there has recently been a growing interest in alternative optimisation schemes that adopt the natural gradient (NG) direction [12]. For instance, in [7] the authors derived a mathematical analysis that suggested we can make better progress when optimising a variational GP along the NG direction, but without providing any experimental results of its performance. The authors in [13] propose to linearise the non-conjugate terms of the model for admitting closed-form updates which are equivalent to optimising in the natural gradient direction. The work in [14] shows how to convert inference in non-conjugate models as it is done in the conjugate ones, by way of expressing the posterior distribution in the mean-parameter space. Furthermore, it shows that by means of exploiting the mirror descent algorithm (MDA) one can arrive to NG updates for tuning the variational posterior

J.J. Giraldo and M. A. Álvarez are with the Department of Computer Science, The University of Sheffield, UK, (e-mail: jjgiraldogutierrez1@sheffield.ac.uk, mauricio.alvarez@sheffield.ac.uk)

distribution. Those works coincide in improvements of training and testing performance, and also fast convergence rates. Nonetheless, they only show results in a full GP model where the kernel hyper-parameters are fixed using a grid search. On the other hand, the work in [15] does show a broad experimental analysis of the NG method for sparse GPs. The authors conclude that the NG is not prone to suffer from ill-conditioning issues in comparison to the AGMs. Also the NG has been used to ease optimisation of the variational posterior over the latent functions of a deep GP model [16]. However, in those two latter cases the NG method only applies for the latent functions' posterior parameters, while an Adam method performs a cooperative optimisation for dealing with the hyper-parameters and inducing points. The authors in [15] call this strategy a hybrid between NG and Adam, and termed it NG+Adam.

The main contributions of this paper include the following:

- We propose a fully natural gradient (FNG) scheme for jointly tuning the hyper-parameters, inducing points and variational posterior parameters of the HetMOGP model. To this end, we borrow ideas from different variational optimisation (VO) strategies like [17], [18] and [19], by introducing an exploratory distribution over the hyper-parameters and inducing points. Such VO strategies have shown to be successful exploratory-learning tools able to avoid poor local optima solutions; they have been broadly studied in the context of reinforcement and Bayesian deep learning, but not much in the context of GPs.
- We provide an extension of the HetMOGP based on a Convolution Processes (CPM) model, rather than an LMC approach as in the original model. This is a novel contribution since there are no former MOGP models with convolution processes that involve stochastic variational inference (SVI), nor a model of heterogeneous outputs that relies on convolution processes.
- We provide a FNG scheme for optimising the new model extension, the HetMOGP with CPM.
- To the best of our knowledge the NG method has not been performed over any MOGP model before. Hence, in this work we also contribute to show how a NG method used in a full scheme over the MOGP's parameters and kernel hyper-parameters alleviates the strong conditioning problems. This, by achieving better local optima solutions with higher test performance rates than Adam and stochastic gradient descent (SGD).
- We explore for the first time in a MOGP model the behaviour of the hybrid strategy NG+Adam, and provide comparative results to our proposed scheme.

II. Variational Optimisation: an Exploratory Mechanism for Optimisation

This section introduces the variational optimisation method as an exploratory mechanism for minimising an objective function [17]. It also shows how Variational Inference (VI) can be seen as a particular case of variational optimisation.

A. Variational Optimisation

The goal in optimisation is to find a proper set of parameters that minimise a possibly non-convex function $g(\boldsymbol{\theta})$ by solving, $\boldsymbol{\theta}^* = \arg\min_{\boldsymbol{\theta}} g(\boldsymbol{\theta})$, where $\boldsymbol{\theta}^*$ represents the set of parameters that minimise the function. The classical way to deal with the above optimisation problem involves deriving w.r.t $\boldsymbol{\theta}$ and solving in a closed-form, or through a gradient descent method. Usually, gradient methods tend to converge to the closest local minima from the starting point without exploring much the space of solutions [20] (see section I of Supplemental Material (SM) for a comparison between VO and Newton's method). Alternatively the variational optimisation method proposes to solve the same problem [17], but introducing exploration in the parameter space of a variational (or exploratory) distribution $q(\boldsymbol{\theta}|\boldsymbol{\xi})$ by bounding the function $g(\boldsymbol{\theta})$ as follows:

$$\tilde{\mathcal{L}}(\boldsymbol{\xi}) = \mathbb{E}_{q(\boldsymbol{\theta}|\boldsymbol{\xi})}[g(\boldsymbol{\theta})] + \mathbb{D}_{KL}\big(q(\boldsymbol{\theta}|\boldsymbol{\xi})||p(\boldsymbol{\theta})\big), \tag{1}$$

where $\mathbb{D}_{KL}(\cdot||\cdot)$ is a Kullback-Leibler (KL) divergence and $p(\boldsymbol{\theta})$ is a penalization distribution. The work of VO in [17] does not introduce the KL term in the equation above, i.e. $\tilde{\mathcal{L}}(\boldsymbol{\xi}) = \mathbb{E}_{q(\boldsymbol{\theta}|\boldsymbol{\xi})}[g(\boldsymbol{\theta})]$, this implies that during an inference process, the exploratory distribution is free to collapse to zero becoming a Dirac's delta $q(\boldsymbol{\theta}) = \delta(\boldsymbol{\theta} - \boldsymbol{\mu})$, where $\boldsymbol{\mu} = \boldsymbol{\theta}^*$ and $\boldsymbol{\mu}$ represents the $q(\boldsymbol{\theta})$'s mean [21], [22]. This collapsing effect limits the exploration of $\boldsymbol{\theta}$'s space (see section I of SM for a graphical example). In contrast, by using the KL term, we can force the exploratory distribution $q(\boldsymbol{\theta}|\boldsymbol{\xi})$ to trade-off between minimising the expectation $\mathbb{E}_{q(\boldsymbol{\theta}|\boldsymbol{\xi})}[g(\boldsymbol{\theta})]$ and not going far away from the imposed $p(\boldsymbol{\theta})$ penalization [23]. Indeed, the KL term in Eq. (1) reduces the collapsing effect of $q(\boldsymbol{\theta})$ and helps to gain additional exploration when an inference process is carried out. With the aim to better understand such behaviour, let us define an example inspired by the one in [18]; we define $g(\theta) = 2\exp(-0.09\theta^2)\sin(4.5\theta)$, a function with multiple local minima, $q(\theta) = \mathcal{N}(\theta|\mu, \sigma^2)$ represents a variational distribution over θ, with parameters mean μ and variance σ^2, and $p(\theta) = \mathcal{N}(\theta|0, \lambda^{-1})$ with $\lambda = 1.0$. We built the graph in Fig. 1 to show what happens at each iteration of the optimisation process. Figure 1 shows three perspectives of a such experiment, where we initialise the parameters $\theta = \mu = -3.0$ and $\sigma = 3.0$. We can notice from Fig. 1 that the initial value of $\theta = \mu = -3.0$ is far away from $g(\theta)$'s global minimum at $\theta \approx -0.346$. When the inference process starts, the exploratory distribution $q(\theta)$ modifies its variance and moves its mean towards a better region in the space of θ. From the third row we can also see that $q(\theta)$ initially behaves as a broad distribution (in light-gray colour) with a mean located at $\mu = -3.0$, while the iterations elapse, the distribution $q(\theta)$ modifies its shape in order to reach a better local minima solution (at $\mu \approx -0.346$). The distribution presents such behaviour in spite of being closer to other poor local minima like the ones between the intervals $(-4, -3)$ and $(-2, -1)$. Additionally, when the mean μ is close to $\theta \approx -0.346$ (the global minimum), the variance parameter reduces constantly making the distribution look narrower, which means it is increasing the certainty of the solution. This behaviour implies that in the long term $q(\theta)$'s mean will be much closer to θ^*.

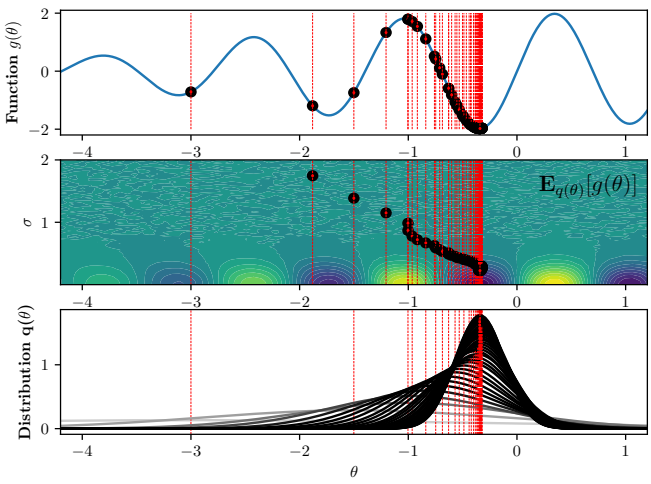


Fig. 1. First row shows what happens from the perspective of the original function $g(\theta) = 2\exp(-0.09\theta^2)\sin(4.5\theta)$, the black dots represent the position of $\theta = \mu$ at each iteration. Second row shows a contour graph of the space of solutions w.r.t σ and μ, here the black dots refer to the position of σ and μ at each iteration, and the low and high colour intensities relate to low and high values of $\mathbb{E}_{q(\theta)}[g(\theta)]$, notice that here we do not include the KL term information for easing the visualisation of the multiple local minima. Third row shows $q(\theta)$'s behaviour, for each Gaussian bell we use a colour code from light-gray to black for representing initial to final stages of the inference. All sub-graphs present vertical lines for aligning iterations, i.e., from left to right the lines represent the occurrence of an iteration. To avoid excessive overlapping, the third row only shows $q(\theta)$ every two iterations.

Therefore, a feasible minima solution for the original objective function $g(\theta)$ is $\theta = \mathbb{E}_{q(\theta)}[\theta] = \mu$, this can be seen in the first sub-graph where at each iteration $\theta = \mu$, in fact, at the end μ is fairly close to the value $\theta \approx -0.346$. Given that in the practise we usually do not have any idea about the landscape of the objective functions that we are interested in optimising, then our suggestion for a practitioner is to initialise the penalisation distribution $p(\theta) = \mathcal{N}(\theta|0, \lambda^{-1})$ with $\lambda = 1.0$; this value presents a stable performance in diverse types of landscapes, or use even a smaller value if more aggressive exploration is desired. (see section II of SM for a detailed analysis of the influence of λ during optimisation).

B. Variational Inference: VO for the Negative Log Likelihood

A common way to build a probabilistic model for a set of observations $\mathbf{X} = \{\mathbf{x}_n\}_{n=1}^N \in \mathbb{R}^{N\times P}$ is to assume that each observation is drawn independently and identically distributed (IID) from a probability distribution $p(\mathbf{X}|\boldsymbol{\theta})$, commonly known as a likelihood. Fitting the model consists on finding the parameter $\boldsymbol{\theta}$ that makes the distribution appropriately explain the data. This inference process is called maximum likelihood estimation, given that is equivalent to the optimisation problem of maximising the log likelihood function $\log p(\mathbf{X}|\boldsymbol{\theta})$, i.e., minimising the negative log likelihood (NLL) function $-\log p(\mathbf{X}|\boldsymbol{\theta})$ [24]. From a Bayesian perspective, we can introduce a prior distribution $p(\boldsymbol{\theta})$ over the parameter of interest, $p(\boldsymbol{\theta}|\mathbf{X}) \propto p(\mathbf{X}|\boldsymbol{\theta})p(\boldsymbol{\theta})$, which implies that there also exists a posterior distribution $p(\boldsymbol{\theta}|\mathbf{X})$ over such parameter, useful to render future predictions of the model. When the likelihood and prior are conjugate, the posterior distribution can be computed in closed form, but that is not always the case. Hence, if the likelihood and prior are non-conjugate, it is necessary to approximate the posterior [25]. Variational inference is a powerful framework broadly used in machine learning, that allows to estimate the posterior by minimising the KL divergence $\mathbb{D}_{KL}\big(q(\boldsymbol{\theta}|\boldsymbol{\xi})||p(\boldsymbol{\theta}|\mathbf{X})\big)$ between an approximate variational posterior $q(\boldsymbol{\theta}|\boldsymbol{\xi})$ and the true posterior $p(\boldsymbol{\theta}|\mathbf{X})$ [26]. Since we do not have access to the true posterior, minimising such KL divergence is equivalent to maximising a lower bound to the marginal likelihood. It emerges from the equality: $\log \mathbb{E}_{q(\boldsymbol{\theta}|\boldsymbol{\xi})}\left[\frac{p(\mathbf{X}|\boldsymbol{\theta})p(\boldsymbol{\theta})}{q(\boldsymbol{\theta}|\boldsymbol{\xi})}\right] = \log p(\mathbf{X})$, in which, after applying the Jensen's inequality we arrive to,

$$-\tilde{\mathcal{L}}(\boldsymbol{\xi}) = \mathbb{E}_{q(\boldsymbol{\theta}|\boldsymbol{\xi})}\left[\log \frac{p(\mathbf{X}|\boldsymbol{\theta})p(\boldsymbol{\theta})}{q(\boldsymbol{\theta}|\boldsymbol{\xi})}\right] \leq \log p(\mathbf{X}), \quad (2)$$

where $\log p(\mathbf{X})$ represents the log marginal likelihood and $-\tilde{\mathcal{L}}(\boldsymbol{\xi})$ is an evidence lower bound (ELBO) [27]. It is noteworthy that if we replace $g(\boldsymbol{\theta}) = -\log p(\mathbf{X}|\boldsymbol{\theta})$ in Eq. (1), we end up with exactly the same lower bound of Eq. (2). Therefore, VI can be seen as a particular case of VO with a KL divergence penalisation, where the objective $g(\boldsymbol{\theta})$ is nothing but the NLL. We can distinguish from two perspectives when using VO for maximum likelihood: for the Bayesian perspective we are not only interested in a point estimate for the parameter $\boldsymbol{\theta}$, but in the uncertainty codified in $q(\boldsymbol{\theta})$'s (co)variance for making future predictions; and for the non-Bayesian perspective the main goal in maximum likelihood estimation is to optimise the function $g(\boldsymbol{\theta}) = -\log p(\mathbf{X}|\boldsymbol{\theta})$. For this case, if $q(\boldsymbol{\theta}|\boldsymbol{\mu}, \boldsymbol{\Sigma})$ is a Gaussian distribution, we can make use of only the posterior's mean $\mathbb{E}_{q(\boldsymbol{\theta}|\boldsymbol{\xi})}[\boldsymbol{\theta}] = \boldsymbol{\mu}$ as a feasible solution for $\boldsymbol{\theta}^*$ without taking into account the uncertainty. This is also known as the maximum a posteriori (MAP) solution in the context of VI, due to the fact that $\boldsymbol{\theta}_{\text{MAP}} = \arg\max_{\boldsymbol{\theta}} p(\boldsymbol{\theta}|\mathbf{X}) \approx q(\boldsymbol{\theta}|\boldsymbol{\mu}, \boldsymbol{\Sigma})$, where the maximum of the distribution $q(\boldsymbol{\theta}|\boldsymbol{\mu}, \boldsymbol{\Sigma})$ is located at its mean, thereby $\boldsymbol{\theta}_{\text{MAP}} = \boldsymbol{\mu}$ (see section VIII of SM for details) [25].

III. Exploiting The Mirror Descent Algorithm

Direct update equations for the parameters of a (posterior) distribution using natural gradients involve the inversion of a Fisher information matrix, which in general it is complex to do. The purpose of this section is to show how an alternative formulation of the NG updates can be derived from the MDA. We introduce the Variational Adaptive-Newton (VAN), a method that benefits from a Gaussian posterior distribution to easily express the parameters updates in the NG direction. And we also introduce the concept of *natural-momentum* which takes advantage of the KL divergence for providing an extra memory information to the MDA.

A. Connection between Natural-Gradient and Mirror Descent

The NG allows to solve an optimisation problem like the one in Eq. (1), where the goal consists on finding an optimal distribution $q(\boldsymbol{\theta})$ that best minimises the objective bound [12]. The method takes advantage of the inverse Fisher information matrix, $\mathbf{F}^{-1}$, associated to the random variable $\boldsymbol{\theta}$, by iteratively weighting the following gradient updates,

$\boldsymbol{\xi}_{t+1} = \boldsymbol{\xi}_t - \alpha_t \mathbf{F}_t^{-1} \hat{\nabla}_{\boldsymbol{\xi}} \tilde{\mathcal{L}}_t$, where α_t is a positive step-size parameter and $\boldsymbol{\xi}_t$ represents the natural (or canonical) parameters of the distribution $q(\boldsymbol{\theta})$. Such natural parameters can be better noticed by expressing the distribution in the general form of the exponential family, $q(\boldsymbol{\theta}) = h(\boldsymbol{\theta}) \exp\big(\langle \boldsymbol{\xi}, \phi(\boldsymbol{\theta}) \rangle - A(\boldsymbol{\xi})\big)$, where $A(\boldsymbol{\xi})$ is the log-partition function, $\phi(\boldsymbol{\theta})$ is a vector of sufficient statistics and $h(\boldsymbol{\theta})$ is a scaling constant [24]. The updates for $\boldsymbol{\xi}_{t+1}$ are expensive due to involving the computation of the inverse Fisher matrix at each iteration. Since an exponential-family distribution has an associated set of mean-parameters $\boldsymbol{\eta} = \mathbb{E}[\phi(\boldsymbol{\theta})]$, then an alternative way to induce the NG updates consists on formulating a MDA in such mean-parameter space. Hence, the algorithm bases on solving the following iterative sub-problems:

$$\boldsymbol{\eta}_{t+1} = \arg\min_{\boldsymbol{\eta}} \langle \boldsymbol{\eta}, \hat{\nabla}_{\boldsymbol{\eta}} \tilde{\mathcal{L}}_t \rangle + \frac{1}{\alpha_t} \mathbb{D}_{KL}(q(\boldsymbol{\theta}) || q_t(\boldsymbol{\theta})), \quad (3)$$

where $\boldsymbol{\eta}$ is the set of $q(\boldsymbol{\theta})$'s mean-parameters, $\tilde{\mathcal{L}}$ is a VO bound of a function $g(\boldsymbol{\theta})$, $\hat{\nabla}_{\boldsymbol{\eta}} \tilde{\mathcal{L}}_t := \hat{\nabla}_{\boldsymbol{\eta}} \tilde{\mathcal{L}}(\boldsymbol{\eta}_t)$ denotes a stochastic gradient, $q_t(\boldsymbol{\theta}) := q(\boldsymbol{\theta}|\boldsymbol{\eta}_t)$ and α_t is a positive step-size parameter [14]. The intention of the above formulation is to exploit the parametrised distribution's structure by controlling its divergence w.r.t its older state $q_t(\boldsymbol{\theta})$. Replacing the distribution $q(\boldsymbol{\theta})$ in its exponential-form, in the above KL divergence, and setting Eq. (3) to zero, let us express,

$$\langle \boldsymbol{\eta}, \hat{\nabla}_{\boldsymbol{\eta}} \tilde{\mathcal{L}}_t \rangle + \frac{1}{\alpha_t} \big[\langle \boldsymbol{\xi}, \boldsymbol{\eta} \rangle - A(\boldsymbol{\xi}) - \langle \boldsymbol{\xi}_t, \boldsymbol{\eta} \rangle + A(\boldsymbol{\xi}_t) \big] = 0,$$

and by deriving w.r.t $\boldsymbol{\eta}$, we arrive to $\boldsymbol{\xi}_{t+1} = \boldsymbol{\xi}_t - \alpha_t \hat{\nabla}_{\boldsymbol{\eta}} \tilde{\mathcal{L}}_t$, where $\boldsymbol{\xi}_{t+1} := \boldsymbol{\xi}$ and $\hat{\nabla}_{\boldsymbol{\eta}} \tilde{\mathcal{L}}_t = \mathbf{F}^{-1} \hat{\nabla}_{\boldsymbol{\xi}} \tilde{\mathcal{L}}_t$ as per the work in [28], where the authors provide a formal proof of such equivalence. The formulation in Eq. (3) is advantageous since it is easier to compute derivatives w.r.t $\boldsymbol{\eta}$ than computing the inverse Fisher information matrix $\mathbf{F}^{-1}$. Therefore, the MDA for solving iterative sub-problems in the mean-parameter space is equivalent to updating the canonical parameters in the NG direction (see section III of SM for more details).

B. Variational Adaptive-Newton and Natural-Momentum

The VAN method aims to solve the problem in Eq. (3) using a Gaussian distribution $q(\boldsymbol{\theta}) := q(\boldsymbol{\theta}|\boldsymbol{\mu}, \boldsymbol{\Sigma})$ as the exploratory mechanism for optimisation [18]. This implies that if $\boldsymbol{\mu}$ and $\boldsymbol{\Sigma}$ represent the mean and covariance respectively, then $q(\boldsymbol{\theta})$'s mean-parameters are $\boldsymbol{\eta} = \{\boldsymbol{\mu}, \boldsymbol{\Sigma} + \boldsymbol{\mu}\boldsymbol{\mu}^\top\}$, and also its analogous natural-parameters are $\boldsymbol{\xi} = \{\boldsymbol{\Sigma}^{-1}\boldsymbol{\mu}, -\frac{1}{2}\boldsymbol{\Sigma}^{-1}\}$. When plugging these parametrisations and solving for the MDA in Eq. (3), we end up with the following updates: $\boldsymbol{\Sigma}_{t+1}^{-1} = \boldsymbol{\Sigma}_t^{-1} + 2\alpha_t \hat{\nabla}_{\boldsymbol{\Sigma}} \tilde{\mathcal{L}}_t$ and $\boldsymbol{\mu}_{t+1} = \boldsymbol{\mu}_t - \alpha_t \boldsymbol{\Sigma}_{t+1} \hat{\nabla}_{\boldsymbol{\mu}} \tilde{\mathcal{L}}_t$, where $\boldsymbol{\mu}_t$ and $\boldsymbol{\Sigma}_t$ are the mean and covariance parameters at the instant t respectively; the stochastic gradients are $\hat{\nabla}_{\boldsymbol{\mu}} \tilde{\mathcal{L}}_t := \hat{\nabla}_{\boldsymbol{\mu}} \tilde{\mathcal{L}}(\boldsymbol{\mu}_t, \boldsymbol{\Sigma}_t)$ and $\hat{\nabla}_{\boldsymbol{\Sigma}} \tilde{\mathcal{L}}_t := \hat{\nabla}_{\boldsymbol{\Sigma}} \tilde{\mathcal{L}}(\boldsymbol{\mu}_t, \boldsymbol{\Sigma}_t)$. These latter updates represent a NG descent algorithm for exploring the space of solutions of the variable $\boldsymbol{\theta}$ through a Gaussian distribution [14]. It is possible to keep exploiting the structure of the distribution $q(\boldsymbol{\theta})$, this by including an additional KL divergence term in the MDA of Eq. (3) as follows: $\boldsymbol{\eta}_{t+1}$

$$= \arg\min_{\boldsymbol{\eta}} \langle \boldsymbol{\eta}, \hat{\nabla}_{\boldsymbol{\eta}} \tilde{\mathcal{L}}_t \rangle + \frac{1}{\tilde{\alpha}_t} \mathrm{KL}(\boldsymbol{\theta})_t - \frac{\tilde{\gamma}_t}{\tilde{\alpha}_t} \mathrm{KL}(\boldsymbol{\theta})_{t-1}, \quad (4)$$

where $q_t(\boldsymbol{\theta}) := q(\boldsymbol{\theta}|\boldsymbol{\mu}_t, \boldsymbol{\Sigma}_t)$ represents the exploratory distributions $q(\boldsymbol{\theta})$ with the parameters obtained at time t, and $\mathrm{KL}(\cdot)_t := \mathbb{D}_{KL}(q(\cdot)||q_t(\cdot))$. Such additional KL term, called as a *natural-momentum* in [19], provides extra memory information to the MDA for potentially improving its convergence rate. This momentum can be controlled by the relation between the positive step-sizes $\tilde{\alpha}_t$ and $\tilde{\gamma}_t$. When solving for Eq. (4), we arrive to the following NG update equations:

$$\boldsymbol{\Sigma}_{t+1}^{-1} = \boldsymbol{\Sigma}_t^{-1} + 2\alpha_t \hat{\nabla}_{\boldsymbol{\Sigma}} \tilde{\mathcal{L}}_t \quad (5)$$

$$\boldsymbol{\mu}_{t+1} = \boldsymbol{\mu}_t - \alpha_t \boldsymbol{\Sigma}_{t+1} \hat{\nabla}_{\boldsymbol{\mu}} \tilde{\mathcal{L}}_t + \gamma_t \boldsymbol{\Sigma}_{t+1} \boldsymbol{\Sigma}_t^{-1} (\boldsymbol{\mu}_t - \boldsymbol{\mu}_{t-1}), \quad (6)$$

where $\alpha_t = \tilde{\alpha}_t/(1 - \tilde{\gamma}_t)$ and $\gamma_t = \tilde{\gamma}_t/(1 - \tilde{\gamma}_t)$ are positive step-size parameters [23], [19].

IV. Heterogeneous Multi-Output Gaussian Process Model

This section provides a brief summary of the state of the art in multi output GPs. It later describes the HetMOGP model. Also, how the inducing points framework allows the model to obtain tractable variational bounds amenable to SVI.

A. Multi-Output Gaussian Processes Review

A MOGP generalises the GP model by exploiting correlations not only in the input space, but also in the output space [1]. Major research about MOGPs has focused on finding proper definitions of a cross-covariance function between multiple outputs. Classical approaches that define such cross-covariance function include the LMC [2] or process convolutions [3]. The works in [1], [4] provide a review of MOGPs that use either LMC or convolution processes approaches. MOGPs have been applied in several problems including sensor networks with missing signals [29]; motion capture data for completing a sequence of missing frames [30]; and natural language processing, where annotating linguistic data is often a complex and time consuming task, and MOGPs can learn from the outputs of multiple annotators [31]. They have been also used in computer emulation, where the LMC, also termed as a Multiple-Output emulator, can be used as a substitute of a computationally expensive deterministic model [32], [33]. Likewise, MOGPs have been useful for learning the couplings between multiple time series and helping to enhance their forecasting capabilities [34]. Recent approaches have focused on building cross-covariances between outputs in the spectral domain [35]. For instance, by constructing a multi-output Convolution Spectral Mixture kernel which incorporates time and phase delays in the spectral density [36]. Other works have concentrated in tackling the issues regarding inference scalability and computation efficiency [4], for example in the context of large datasets using collaborative MOGPs [37]; introducing an scalable inference procedure with a mixture of Gaussians as a posterior approximation [38]. Other works have investigated alternative paradigms to MOGPs. For instance, the work in [39] has explored combinations of GPs with Bayesian neural networks (BNN) so as to take advantage from the GPs' non-parametric flexibility and the BNN's structural properties for modelling multiple-outputs. Another recent work relies

on a product rule to decompose the joint distribution of the outputs given the inputs into conditional distributions, i.e. decoupling the model into single-output regression tasks [40]. Most work on MOGPs including [33], [36], [40] has focused on Gaussian multivariate regression. As we have mentioned before, in this paper we focus on the HetMOGP that concerns about outputs with different statistical data types, and extends the MOGPs' application to heterogeneous outputs [5].

B. The Likelihood Function for the HetMOGP

The HetMOGP model is an extension of the Multi-Output GP that allows different kinds of likelihoods as per the statistical data type each output demands [5]. For instance, if we have two outputs problem, where one output is binary $y_1 \in \{0, 1\}$ while the other is a real value $y_2 \in \mathbb{R}$, we can assume our likelihood as the product of a Bernoulli and Gaussian distribution for each output respectively. In general the HetMOGP likelihood for D outputs can be written as:

$$p(\mathbf{y}|\mathbf{f}) = \prod_{n=1}^{N}\prod_{d=1}^{D} p(y_{d,n}|\psi_{d,1}(\mathbf{x}_n), ..., \psi_{d,J_d}(\mathbf{x}_n)), \quad (7)$$

where the vector $\mathbf{y} = [\mathbf{y}_1^\top, ..., \mathbf{y}_D^\top]^\top$ groups all the output observations and each $\psi_{d,j}(\mathbf{x}_n)$ represents the j-th parameter that belongs to the d-th likelihood. It is worth noticing that each output vector $\mathbf{y}_d$ is generated by a particular set of input observations $\mathbf{X}_d$. Though, in order to ease the explanation of the model and to be consistent with the equation above, we have assumed that all outputs $\mathbf{y}_d = [y_{d,1}, ..., y_{d,N}]^\top$ are related to the same input observations $\mathbf{X} = [\mathbf{x}_1, ..., \mathbf{x}_N]^\top \in \mathbb{R}^{N\times P}$. Each likelihoods' parameter $\psi_{d,j}(\mathbf{x}_n)$ is chained to a latent function $f_{d,j}(\cdot)$ that follows a GP prior, through a link function $\phi(\cdot)$, i.e., $\psi_{d,j}(\mathbf{x}_n) = \phi(f_{d,j}(\mathbf{x}_n))$. For instance, if we have two outputs where the first likelihood is a Heteroscedastic Gaussian, then its parameters mean and variance are respectively chained as $\psi_{1,1}(\mathbf{x}_n) = f_{1,1}(\mathbf{x}_n)$ and $\psi_{1,2}(\mathbf{x}_n) = \exp(f_{1,2}(\mathbf{x}_n))$; if the second likelihood is a Gamma, its parameters are linked as $\psi_{2,1}(\mathbf{x}_n) = \exp(f_{2,1}(\mathbf{x}_n))$ and $\psi_{2,2}(\mathbf{x}_n) = \exp(f_{2,2}(\mathbf{x}_n))$ [6]. Notice that J_d accounts for the number of latent functions necessary to parametrise the d-th likelihood, thus the total number of functions $f_{d,j}(\cdot)$ associated to the model becomes $J = \sum_{d=1}^{D} J_d$. Each $f_{d,j}(\cdot)$ is considered a latent parameter function (LPF) that comes from a LMC as follows:

$$f_{d,j}(\mathbf{x}) = \sum_{q=1}^{Q}\sum_{i=1}^{R_q} a_{d,j,q}^i u_q^i(\mathbf{x}), \quad (8)$$

where $u_q^i(\mathbf{x})$ are IID samples from GPs $u_q(\cdot) \sim \mathcal{GP}(0, k_q(\cdot,\cdot))$ and $a_{d,j,q}^i \in \mathbb{R}$ is a linear combination coefficient (LCC). In Section V, we introduce a different way to model $f_{d,j}(\mathbf{x})$ based on convolution processes. For the sake of future explanations let us assume that $R_q = 1$. In this way the number of LCCs per latent function $u_q(\cdot)$ becomes J. The coefficients per function $u_q(\cdot)$ can be grouped in a vector $\mathbf{w}_q = [a_{1,1,q}, ..., a_{1,J_1,q}, ..., a_{D,J_D,q}]^\top \in \mathbb{R}^{J\times 1}$; and we can cluster all vectors $\mathbf{w}_q$ in a specific vector of LCCs $\mathbf{w} = [\mathbf{w}_1^\top, ..., \mathbf{w}_Q^\top]^\top \in \mathbb{R}^{QJ\times 1}$.

C. The Inducing Points Method

A common approach for reducing computational complexity in GP models is to augment the GP prior with a set of *inducing variables*. For the specific case of the HetMOGP model with LMC prior, the vector of *inducing variables* $\mathbf{u} = [\mathbf{u}_1^\top, ..., \mathbf{u}_Q^\top]^\top \in \mathbb{R}^{QM\times 1}$ is built from $\mathbf{u}_q = [u_q(\mathbf{z}_q^{(1)}), ..., u_q(\mathbf{z}_q^{(M)})]^\top \in \mathbb{R}^{M\times 1}$. Notice that the vector $\mathbf{u}_q$ is constructed by additional evaluations of the functions $u_q(\cdot)$ at some unknown inducing points $\mathbf{Z}_q = [\mathbf{z}_q^{(1)}, ..., \mathbf{z}_q^{(M)}]^\top \in \mathbb{R}^{M\times P}$. The vector of all inducing variables can be expressed as $\mathbf{z} = [\text{vec}(\mathbf{Z}_1)^\top, ..., \text{vec}(\mathbf{Z}_Q)^\top]^\top \in \mathbb{R}^{QMP\times 1}$ [9], [41]. We can write the augmented GP prior as follows,

$$p(\mathbf{f}|\mathbf{u})p(\mathbf{u}) = \prod_{d=1}^{D}\prod_{j=1}^{J_d} p(\mathbf{f}_{d,j}|\mathbf{u}) \prod_{q=1}^{Q} p(\mathbf{u}_q), \quad (9)$$

where $\mathbf{f} = [\mathbf{f}_{1,1}^\top, ..., \mathbf{f}_{1,J_1}^\top, ..., \mathbf{f}_{D,J_D}^\top]^\top$ is a vector built from $\mathbf{f}_{d,j} = [f_{d,j}(\mathbf{x}_1), ..., f_{d,j}(\mathbf{x}_N)]^\top \in \mathbb{R}^{N\times 1}$. Following the conditional Gaussian properties we can express,

$$p(\mathbf{f}_{d,j}|\mathbf{u}) = \mathcal{N}(\mathbf{f}_{d,j}|\mathbf{A}_{\mathbf{f}_{d,j}\mathbf{u}}\mathbf{u}, \widetilde{\mathbf{Q}}_{\mathbf{f}_{d,j}\mathbf{f}_{d,j}}), p(\mathbf{u}) = \mathcal{N}(\mathbf{u}|\mathbf{0}, \mathbf{K}_{\mathbf{uu}}),$$

where the matrix $\mathbf{K}_{\mathbf{uu}} \in \mathbb{R}^{QM\times QM}$ is a block-diagonal with blocks $\mathbf{K}_{\mathbf{u}_q\mathbf{u}_q} \in \mathbb{R}^{M\times M}$ built from evaluations of $\text{cov}[u_q(\cdot), u_q(\cdot)] = k_q(\cdot,\cdot)$ between all pairs of inducing points $\mathbf{Z}_q$ respectively; and we have introduced the following definitions, $\mathbf{A}_{\mathbf{f}_{d,j}\mathbf{u}} = \mathbf{K}_{\mathbf{f}_{d,j}\mathbf{u}}\mathbf{K}_{\mathbf{uu}}^{-1}$, $\widetilde{\mathbf{Q}}_{\mathbf{f}_{d,j}\mathbf{f}_{d,j}} = \mathbf{K}_{\mathbf{f}_{d,j}\mathbf{f}_{d,j}} - \mathbf{Q}_{\mathbf{f}_{d,j}\mathbf{f}_{d,j}}$, $\mathbf{Q}_{\mathbf{f}_{d,j}\mathbf{f}_{d,j}} = \mathbf{K}_{\mathbf{f}_{d,j}\mathbf{u}}\mathbf{K}_{\mathbf{uu}}^{-1}\mathbf{K}_{\mathbf{u}\mathbf{f}_{d,j}}$, $\mathbf{K}_{\mathbf{f}_{d,j}\mathbf{u}} = \mathbf{K}_{\mathbf{u}\mathbf{f}_{d,j}}^\top$. Here the covariance matrix $\mathbf{K}_{\mathbf{f}_{d,j}\mathbf{f}_{d,j}} \in \mathbb{R}^{N\times N}$ is built from the evaluation of all pairs of input data $\mathbf{X}$ in the covariance function $\text{cov}[f_{d,j}(\cdot), f_{d,j}(\cdot)] = \sum_{q=1}^{Q} a_{d,j,q}^2 k_q(\cdot,\cdot)$; and the cross covariance matrix $\mathbf{K}_{\mathbf{f}_{d,j}\mathbf{u}} = [\mathbf{K}_{\mathbf{f}_{d,j}\mathbf{u}_1}, ..., \mathbf{K}_{\mathbf{f}_{d,j}\mathbf{u}_Q}] \in \mathbb{R}^{N\times QM}$ is constructed with the blocks $\mathbf{K}_{\mathbf{f}_{d,j}\mathbf{u}_q} \in \mathbb{R}^{N\times M}$, formed by the evaluations of $\text{cov}[f_{d,j}(\cdot), u_q(\cdot)] = a_{d,j,q}k_q(\cdot,\cdot)$ between inputs $\mathbf{X}$ and $\mathbf{Z}_q$. Each kernel covariance $k_q(\cdot,\cdot)$ has an Exponentiated Quadratic (EQ) form as follows:

$$\mathcal{E}(\boldsymbol{\tau}|\mathbf{0}, \mathbf{L}) = \frac{|\mathbf{L}|^{-1/2}}{(2\pi)^{P/2}} \exp\left[-\frac{1}{2}\boldsymbol{\tau}^\top\mathbf{L}^{-1}\boldsymbol{\tau}\right], \quad (10)$$

where $\boldsymbol{\tau} := \mathbf{x} - \mathbf{x}'$ and $\mathbf{L}$ is a diagonal matrix of length-scales. Thus, each $k_q(\mathbf{x}, \mathbf{x}') = \mathcal{E}(\boldsymbol{\tau}|\mathbf{0}, \mathbf{L}_q)$.

D. The Evidence Lower Bound

We follow a VI derivation similar to the one used for single output GPs [7], [6]. This approach allows the use of HetMOGP for large data. The goal is to approximate the true posterior $p(\mathbf{f}, \mathbf{u}|\mathbf{y})$ with a variational distribution $q(\mathbf{f}, \mathbf{u})$ by optimising the following negative ELBO:

$$\tilde{\mathcal{L}} = \sum_{n,d=1}^{N,D} \mathbb{E}_{q(\mathbf{f}_{d,1})\cdots q(\mathbf{f}_{d,J_d})}[g_{d,n}] + \sum_{q=1}^{Q} \mathbb{D}_{KL}(\mathbf{u}_q), \quad (11)$$

where $g_{d,n} = -\log p(y_{d,n}|\psi_{d,1}(\mathbf{x}_n), ..., \psi_{d,J_d}(\mathbf{x}_n))$ is the NLL function associated to each output, $\mathbb{D}_{KL}(\mathbf{u}_q) := \mathbb{D}_{KL}(q(\mathbf{u}_q)\|p(\mathbf{u}_q))$, and we have set a tractable posterior $q(\mathbf{f}, \mathbf{u}) = p(\mathbf{f}|\mathbf{u})q(\mathbf{u})$, where $p(\mathbf{f}|\mathbf{u})$ is already defined in Eq. (9), $q(\mathbf{u}|\mathbf{m}, \mathbf{V}) = \prod_{q=1}^{Q} q(\mathbf{u}_q)$, and each $q(\mathbf{u}_q) = \mathcal{N}(\mathbf{u}_q|\mathbf{m}_q, \mathbf{V}_q)$ is a Gaussian distribution with mean $\mathbf{m}_q$ and

covariance $\mathbf{V}_q$ [42] (see section IV of SM for details on the ELBO derivation). The above expectation associated to the NLL is computed using the marginal posteriors,

$$q(\mathbf{f}_{d,j}) := \mathcal{N}(\mathbf{f}_{d,j}|\tilde{\mathbf{m}}_{\mathbf{f}_{d,j}}, \tilde{\mathbf{V}}_{\mathbf{f}_{d,j}}), \tag{12}$$

with the following definitions, $\tilde{\mathbf{m}}_{\mathbf{f}_{d,j}} := \mathbf{A}_{\mathbf{f}_{d,j}\mathbf{u}}\mathbf{m}$, $\tilde{\mathbf{V}}_{\mathbf{f}_{d,j}} := \mathbf{K}_{\mathbf{f}_{d,j}\mathbf{f}_{d,j}} + \mathbf{A}_{\mathbf{f}_{d,j}\mathbf{u}}(\mathbf{V} - \mathbf{K}_{\mathbf{uu}})\mathbf{A}_{\mathbf{f}_{d,j}\mathbf{u}}^\top$, where mean $\mathbf{m} = [\mathbf{m}_1^\top, ..., \mathbf{m}_Q^\top]^\top \in \mathbb{R}^{QM\times 1}$ and the covariance matrix $\mathbf{V} \in \mathbb{R}^{QM\times QM}$ is a block-diagonal matrix with blocks given by $\mathbf{V}_q \in \mathbb{R}^{M\times M}$.[1] The objective function derived in Eq. (11) for the HetMOGP model with LMC requires fitting the parameters of each posterior $q(\mathbf{u}_q)$, the inducing points $\mathbf{z}$, the kernel hyper-parameters $\mathbf{l}_{\text{kern}} = [\mathbf{L}_1^\top, ..., \mathbf{L}_Q^\top]^\top$ and the coefficients $\mathbf{w}$. With the aim to fit said variables in a FNG scheme, later on we will apply the VO perspective on Eq. (11) for inducing randomness and gain exploration over $\mathbf{z}$, $\mathbf{l}_{\text{kern}}$ and $\mathbf{w}$; and by means of the MDA we will derive the inference updates for all the model's variables.

V. Heterogeneous Multi-Output GPs with Convolution Processes

The HetMOGP model with convolution processes follows the same likelihood defined in Eq. (7), though each $f_{d,j}(\mathbf{x}_n)$ is considered a LPF that comes from a convolution process as follows: $f_{d,j}(\mathbf{x}) = \sum_{q=1}^{Q}\sum_{i=1}^{R_q}\int_{\mathcal{X}} G_{d,j,q}^i(\mathbf{x}-\mathbf{r}')\,u_q^i(\mathbf{r}')\,\mathrm{d}\mathbf{r}'$, where $u_q^i(\mathbf{x})$ are IID samples from Gaussian Processes $u_q(\cdot) \sim \mathcal{GP}(0, k_q(\cdot,\cdot))$ and each $G_{d,j,q}(\cdot)$ represents a smoothing kernel. We will also use $R_q = 1$ as in the LMC for simplicity in the following derivations.

A. The Inducing Points Method

With the purpose to reduce the computational complexities involved in GPs we follow the inducing variables framework by augmenting the probability space as,

$$p(\mathbf{f}|\check{\mathbf{u}})p(\check{\mathbf{u}}) = \prod_{d=1}^{D}\prod_{j=1}^{J_d} p(\mathbf{f}_{d,j}|\check{\mathbf{u}}_{d,j})p(\check{\mathbf{u}}_{d,j}), \tag{13}$$

with $p(\check{\mathbf{u}}) = \prod_{d=1}^{D}\prod_{j=1}^{J_d} p(\check{\mathbf{u}}_{d,j})$, and $p(\mathbf{f}|\check{\mathbf{u}}) = \prod_{d=1}^{D}\prod_{j=1}^{J_d} p(\mathbf{f}_{d,j}|\check{\mathbf{u}}_{d,j})$, where the vector $\check{\mathbf{u}} = [\check{\mathbf{u}}_{1,1}^\top, ..., \check{\mathbf{u}}_{1,J_1}^\top, ..., \check{\mathbf{u}}_{D,J_D}^\top]^\top \in \mathbb{R}^{JM\times 1}$ is built from the inducing variables $\check{\mathbf{u}}_{d,j} = [f_{d,j}(\mathbf{z}_{d,j}^{(1)}), ..., f_{d,j}(\mathbf{z}_{d,j}^{(M)})]^\top \in \mathbb{R}^{M\times 1}$. As it can be seen, these inducing variables are additional evaluations of the functions $f_{d,j}(\cdot)$ at each set of inducing points $\mathbf{Z}_{d,j} = [\mathbf{z}_{d,j}^{(1)}, ..., \mathbf{z}_{d,j}^{(M)}]^\top \in \mathbb{R}^{M\times P}$, thus the set of all inducing variables is $\mathbf{z} = [\mathrm{vec}(\mathbf{Z}_{1,1})^\top, ..., \mathrm{vec}(\mathbf{Z}_{1,J_1})^\top, ..., \mathrm{vec}(\mathbf{Z}_{D,J_D})^\top]^\top \in \mathbb{R}^{JMP\times 1}$. Using the properties of Gaussian distributions, we can express $p(\mathbf{f}_{d,j}|\check{\mathbf{u}}_{d,j}) = \mathcal{N}(\mathbf{f}_{d,j}|\mathbf{A}_{\mathbf{f}_{d,j}\check{\mathbf{u}}_{d,j}}\check{\mathbf{u}}_{d,j}, \bar{\mathbf{Q}}_{\mathbf{f}_{d,j}})$, $p(\check{\mathbf{u}}_{d,j}) = \mathcal{N}(\check{\mathbf{u}}_{d,j}|\mathbf{0}, \mathbf{K}_{\check{\mathbf{u}}_{d,j}})$, with the following definitions: $\mathbf{A}_{\mathbf{f}_{d,j}\check{\mathbf{u}}_{d,j}} = \mathbf{K}_{\mathbf{f}_{d,j}\check{\mathbf{u}}_{d,j}}\mathbf{K}_{\check{\mathbf{u}}_{d,j}}^{-1}$, $\bar{\mathbf{Q}}_{\mathbf{f}_{d,j}} = \mathbf{K}_{\mathbf{f}_{d,j}\mathbf{f}_{d,j}} - \check{\mathbf{Q}}_{\mathbf{f}_{d,j}}$, $\check{\mathbf{Q}}_{\mathbf{f}_{d,j}} = \mathbf{K}_{\mathbf{f}_{d,j}\check{\mathbf{u}}_{d,j}}\mathbf{K}_{\check{\mathbf{u}}_{d,j}}^{-1}\mathbf{K}_{\check{\mathbf{u}}_{d,j}\mathbf{f}_{d,j}}$, $\mathbf{K}_{\mathbf{f}_{d,j}\check{\mathbf{u}}_{d,j}} = \mathbf{K}_{\check{\mathbf{u}}_{d,j}\mathbf{f}_{d,j}}^\top$. Here the covariance matrix $\mathbf{K}_{\mathbf{f}_{d,j}\mathbf{f}_{d,j}} \in \mathbb{R}^{N\times N}$ is built from the evaluation of all pairs of input data $\mathbf{X} \in \mathbb{R}^{N\times P}$ in the covariance function $\mathrm{cov}[f_{d,j}(\mathbf{x})\,f_{d',j'}(\mathbf{x}')] = \sum_{q=1}^{Q}\int_{\mathcal{X}} G_{d,j,q}(\mathbf{x}-\mathbf{r})\int_{\mathcal{X}} G_{d',j',q}(\mathbf{x}'-\mathbf{r}')\,k_q(\mathbf{r},\mathbf{r}')\mathrm{d}\mathbf{r}\mathrm{d}\mathbf{r}'$, the cross covariance matrix $\mathbf{K}_{\mathbf{f}_{d,j}\check{\mathbf{u}}_{d,j}} \in \mathbb{R}^{N\times M}$ is formed by evaluations of the equation above between inputs $\mathbf{X}$ and $\mathbf{Z}_{d,j}$, and the matrix $\mathbf{K}_{\check{\mathbf{u}}_{d,j}} \in \mathbb{R}^{M\times M}$ is also built from evaluations of the equation above between all pairs of inducing points $\mathbf{Z}_{d,j}$ respectively. We can compute the above covariance function analytically for certain forms of $G_{d,j,q}(\cdot)$ and $k_q(\mathbf{r},\mathbf{r}')$. In this paper, we follow the work in [4] by defining the kernels in the EQ form of Eq. (10): $k_q(\mathbf{x},\mathbf{x}') = \mathcal{E}(\boldsymbol{\tau}|\mathbf{0},\mathbf{L}_q)$ and $G_{d,j,q}(\boldsymbol{\tau}) = S_{d,j,q}\mathcal{E}(\boldsymbol{\tau}|\mathbf{0},\boldsymbol{\kappa}_{d,j})$, where $S_{d,j,q}$ is a weight associated to the LPF indexed by $f_{d,j}(\cdot)$ and to the latent function $u_q(\cdot)$, and $\boldsymbol{\kappa}_{d,j}$ is a diagonal covariance matrix particularly associated to each $f_{d,j}(\cdot)$; $\boldsymbol{\kappa}_{d,j}$ can be seen as a matrix of length-scales in its diagonal. Therefore, when solving for the $\mathrm{cov}[f_{d,j}(\mathbf{x})\,f_{d',j'}(\mathbf{x}')]$ above we end up with the closed-form,

$$k_{f_{d,j},f_{d',j'}}(\boldsymbol{\tau}) = \sum_{q=1}^{Q} S_{d,j,q}S_{d',j',q}\mathcal{E}(\boldsymbol{\tau}|\mathbf{0},\mathbf{P}_{d,j,d',j',q}), \tag{14}$$

where $\mathbf{P}_{d,j,d',j',q}$ represents a diagonal matrix of length-scales, $\mathbf{P}_{d,j,d',j',q} = \boldsymbol{\kappa}_{d,j} + \boldsymbol{\kappa}_{d',j'} + \mathbf{L}_q$.

B. The Evidence Lower Bound

We now introduce the negative ELBO for the HetMOGP that uses convolution processes. It follows as

$$\tilde{\mathcal{L}} = \sum_{n,d=1}^{N,D}\mathbb{E}_{q(\mathbf{f}_{d,1})\cdots q(\mathbf{f}_{d,J_d})}[g_{d,n}] + \sum_{d,j=1}^{D,J_d}\mathbb{D}_{KL}(\check{\mathbf{u}}_{d,j}), \tag{15}$$

where $g_{d,n} = -\log p(y_{d,n}|\psi_{d,1}(\mathbf{x}_n), ..., \psi_{d,J_d}(\mathbf{x}_n))$ is the NLL function associated to each output, $\mathbb{D}_{KL}(\check{\mathbf{u}}_{d,j}) := \mathbb{D}_{KL}(q(\check{\mathbf{u}}_{d,j})\|p(\check{\mathbf{u}}_{d,j}))$, and we have set a tractable posterior $q(\mathbf{f},\check{\mathbf{u}}) = p(\mathbf{f}|\check{\mathbf{u}})q(\check{\mathbf{u}})$, where $p(\mathbf{f}|\check{\mathbf{u}})$ is already defined in Eq. (13), $q(\check{\mathbf{u}}|\mathbf{m},\mathbf{V}) = \prod_{d=1}^{D}\prod_{j=1}^{J_d} q(\check{\mathbf{u}}_{d,j})$, and each $q(\check{\mathbf{u}}_{d,j}) = \mathcal{N}(\check{\mathbf{u}}_{d,j}|\mathbf{m}_{d,j},\mathbf{V}_{d,j})$ is a Gaussian distribution with mean $\mathbf{m}_{d,j} \in \mathbb{R}^{M\times 1}$ and covariance $\mathbf{V}_{d,j} \in \mathbb{R}^{M\times M}$ (see section V of SM for details on the ELBO derivation). The above expectation is computed w.r.t the marginals, $q(\mathbf{f}_{d,j})$

$$= \int p(\mathbf{f}_{d,j}|\check{\mathbf{u}}_{d,j})q(\check{\mathbf{u}}_{d,j})d\check{\mathbf{u}}_{d,j} = \mathcal{N}(\mathbf{f}_{d,j}|\tilde{\mathbf{m}}_{\mathbf{f}_{d,j}}, \tilde{\mathbf{V}}_{\mathbf{f}_{d,j}}), \tag{16}$$

with the following definitions, $\tilde{\mathbf{m}}_{\mathbf{f}_{d,j}} := \mathbf{A}_{\mathbf{f}_{d,j}\check{\mathbf{u}}_{d,j}}\mathbf{m}_{d,j}$, $\tilde{\mathbf{V}}_{\mathbf{f}_{d,j}} := \mathbf{K}_{\mathbf{f}_{d,j}\mathbf{f}_{d,j}} + \mathbf{A}_{\mathbf{f}_{d,j}\check{\mathbf{u}}_{d,j}}(\mathbf{V}_{d,j} - \mathbf{K}_{\check{\mathbf{u}}_{d,j}})\mathbf{A}_{\mathbf{f}_{d,j}\check{\mathbf{u}}_{d,j}}^\top$. The objective derived in Eq. (15) for the HetMOGP model with convolution processes requires fitting the parameters of each posterior $q(\check{\mathbf{u}}_{d,j})$, the inducing points $\mathbf{z}$, the kernel hyper-parameters $\mathbf{l}_{\text{kern}}$, the smoothing-kernels' length-scales $\boldsymbol{\kappa}_{\text{smooth}} = [\boldsymbol{\kappa}_{1,1}^\top, ..., \boldsymbol{\kappa}_{1,J_1}^\top, ..., \boldsymbol{\kappa}_{D,J_D}^\top]^\top \in \mathbb{R}_+^{JP\times 1}$ and the weights $\mathbf{s}_q = [S_{1,1,q}, ..., S_{1,J_1,q}, ..., S_{D,J_D,q}]^\top \in \mathbb{R}^{J\times 1}$ associated to each smoothing-kernel. In the interest of fitting those variables in a FNG scheme, in the following section we will explain how to apply the VO perspective over Eq. (15) so as to introduce stochasticity over $\mathbf{z}$, $\mathbf{l}_{\text{kern}}$, $\boldsymbol{\kappa}_{\text{smooth}}$ and $\mathbf{s}_q$; and through the MDA we will derive closed-form updates for all parameters of the model.

[1] Each marginal posterior derives from: $q(\mathbf{f}_{d,j}) = \int p(\mathbf{f}_{d,j}|\mathbf{u})q(\mathbf{u})d\mathbf{u}$.

VI. Deriving a Fully Natural Gradient Scheme

This section describes how to derive the FNG updates for optimising both the LMC and CPM schemes of the HetMOGP model. We first detail how to induce an exploratory distribution over the hyper-parameters and inducing points, then we write down the MDA for the model and derive the update equations. Later on, we get into specific details about the algorithm's implementation.

A. An Exploratory Distribution for HetMOGP with LMC

In the context of sparse GPs, the kernel hyper-parameters and inducing points of the model have usually been treated as deterministic variables. Here, we use the VO perspective as a mechanism to induce randomness over such variables, this with the aim to gain exploration for finding better solutions during the inference process. To this end we define and connect random real vectors to the variables through a link function $\phi(\cdot)$ as follows: for the inducing points $\mathbf{z} = \boldsymbol{\theta}_{\mathbf{z}}$, for the kernel hyper-parameters $\mathbf{l}_{\text{kern}} = \exp(\boldsymbol{\theta}_{\mathbf{L}})$ with $\boldsymbol{\theta}_{\mathbf{L}} = [\boldsymbol{\theta}_{\mathbf{L}_1}^\top, ..., \boldsymbol{\theta}_{\mathbf{L}_Q}^\top]^\top \in \mathbb{R}^{QP\times 1}$, and for the vector of LCC $\mathbf{w} = \boldsymbol{\theta}_{\mathbf{w}}$, that are used to generate the LPFs in Eq. (8). We have defined the real random vectors $\boldsymbol{\theta}_{\mathbf{z}} \in \mathbb{R}^{QMP\times 1}$, $\boldsymbol{\theta}_{\mathbf{L}_q} \in \mathbb{R}^{P\times 1}$ and $\boldsymbol{\theta}_{\mathbf{w}} \in \mathbb{R}^{QJ\times 1}$ to link the set of inducing points, the kernel hyper-parameters per latent function $u_q(\cdot)$, and the vector $\mathbf{w}$ of LCCs. We cluster the random vectors defining $\boldsymbol{\theta} = [\boldsymbol{\theta}_{\mathbf{z}}^\top, \boldsymbol{\theta}_{\mathbf{L}}^\top, \boldsymbol{\theta}_{\mathbf{w}}^\top]^\top \in \mathbb{R}^{(QMP+QP+QJ)\times 1}$ to refer to all the parameters in a single variable. Hence, we can specify an exploratory distribution $q(\boldsymbol{\theta}) := \mathcal{N}(\boldsymbol{\theta}|\boldsymbol{\mu}, \boldsymbol{\Sigma})$ for applying the VO approach in Eq. (1), though for our case the objective to bound is $\tilde{\mathcal{L}}$, already derived in Eq. (11) for the HetMOGP with a LMC. Therefore our VO bound is defined as follows:

$$\tilde{\mathcal{F}} = \mathbb{E}_{q(\boldsymbol{\theta})}\big[\tilde{\mathcal{L}}\big] + \mathbb{D}_{KL}(q(\boldsymbol{\theta})||p(\boldsymbol{\theta})), \tag{17}$$

where $p(\boldsymbol{\theta}) = \mathcal{N}(\boldsymbol{\theta}|\mathbf{0}, \lambda_1^{-1}\mathbf{I})$ is a Gaussian distribution with precision λ_1 that forces further exploration of $\boldsymbol{\theta}$'s space [23].

B. An Exploratory Distribution for the HetMOGP with CPM

The case of the CPM has the same kernel hyper-parameters $\mathbf{l}_{\text{kern}} = \exp(\boldsymbol{\theta}_{\mathbf{L}})$ and inducing points $\mathbf{z} = \boldsymbol{\theta}_{\mathbf{z}}$ as the LMC case, but differs from it since the smoothing kernels involve a new set of hyper-parameters, the smoothing-kernels' length-scales. The way we define and connect the new random real vectors is as follows: $\boldsymbol{\kappa}_{\text{smooth}} = \exp(\boldsymbol{\theta}_{\boldsymbol{\kappa}})$, with $\boldsymbol{\theta}_{\boldsymbol{\kappa}} = [\boldsymbol{\theta}_{\boldsymbol{\kappa}_{1,1}}^\top, ..., \boldsymbol{\theta}_{\boldsymbol{\kappa}_{1,J_1}}^\top, ..., \boldsymbol{\theta}_{\boldsymbol{\kappa}_{D,J_D}}^\top]^\top \in \mathbb{R}^{JP\times 1}$, where $\boldsymbol{\theta}_{\boldsymbol{\kappa}_{d,j}} \in \mathbb{R}^{P\times 1}$ is a real random vector associated to each smoothing kernel $G_{d,j,q}(\cdot)$ from Eq. (14). Also, instead of the combination coefficients $\mathbf{w}$ of the LMC, for the CPM we have an analogous set of weights from the smoothing-kernels in Eq. (14), $\mathbf{s} = \boldsymbol{\theta}_{\mathbf{s}}$, where $\mathbf{s} = [\mathbf{s}_1^\top, ..., \mathbf{s}_Q^\top]^\top \in \mathbb{R}^{QJ\times 1}$ is a vector that groups all the weights that belong to the smoothing kernels. Thus, the real random vectors for the CPM are: $\boldsymbol{\theta}_{\mathbf{z}} \in \mathbb{R}^{JMP\times 1}$, $\boldsymbol{\theta}_{\mathbf{L}} \in \mathbb{R}^{QP\times 1}$, $\boldsymbol{\theta}_{\boldsymbol{\kappa}} \in \mathbb{R}^{JP\times 1}$, and $\boldsymbol{\theta}_{\mathbf{s}} \in \mathbb{R}^{QJ\times 1}$. We group the random vectors by defining $\boldsymbol{\theta} = [\boldsymbol{\theta}_{\mathbf{z}}^\top, \boldsymbol{\theta}_{\mathbf{L}}^\top, \boldsymbol{\theta}_{\boldsymbol{\kappa}}^\top, \boldsymbol{\theta}_{\mathbf{s}}^\top]^\top \in \mathbb{R}^{(JMP+QP+JP+QJ)\times 1}$. Notice that, for the CPM, the dimensionality of the real random vector $\boldsymbol{\theta}$ differs from the one for LMC, this is due to the way the inducing variables are treated in subsection V-A and the additional set of smoothing-kernel's hyper-parameters. In the same way as defined for the LMC, we specify an exploratory distribution $q(\boldsymbol{\theta}) := \mathcal{N}(\boldsymbol{\theta}|\boldsymbol{\mu}, \boldsymbol{\Sigma})$ and follow the VO approach in Eq. (1). In this case the objective to bound is the one derived for the CPM, i.e., the new bound, $\tilde{\mathcal{F}}$, is exactly the same as Eq. (17), but using the corresponding $\tilde{\mathcal{L}}$ from Eq. (15).

C. Mirror Descent Algorithm for the HetMOGP with LMC

With the purpose of minimising our VO objective in Eq. (17), we use the MDA in Eq. (4) which additionally exploits the natural-momentum. In the interest of easing the derivation, we use the mean-parameters of distributions $q(\mathbf{u}_q)$ and $q(\boldsymbol{\theta})$ defining $\boldsymbol{\rho}_q = \{\mathbf{m}_q, \mathbf{m}_q\mathbf{m}_q^\top + \mathbf{V}_q\}$ and $\boldsymbol{\eta} = \{\boldsymbol{\mu}, \boldsymbol{\mu}\boldsymbol{\mu}^\top + \boldsymbol{\Sigma}\}$. In this way we can write the MDA as: $\boldsymbol{\eta}_{t+1}, \{\boldsymbol{\rho}_{q,t+1}\}_{q=1}^Q$

$$= \underset{\boldsymbol{\eta}, \{\boldsymbol{\rho}_q\}_{q=1}^Q}{\arg\min} \langle \boldsymbol{\eta}, \hat{\nabla}_{\boldsymbol{\eta}}\tilde{\mathcal{F}}_t\rangle + \frac{1}{\tilde{\alpha}_t}\text{KL}(\boldsymbol{\theta})_t - \frac{\tilde{\gamma}_t}{\tilde{\alpha}_t}\text{KL}(\boldsymbol{\theta})_{t-1} \tag{18}$$
$$+ \sum_{q=1}^{Q}\Big[\langle \boldsymbol{\rho}_q, \hat{\nabla}_{\boldsymbol{\rho}_q}\tilde{\mathcal{F}}_t\rangle + \frac{1}{\tilde{\beta}_t}\text{KL}(\mathbf{u}_q)_t - \frac{\tilde{\upsilon}_t}{\tilde{\beta}_t}\text{KL}(\mathbf{u}_q)_{t-1}\Big],$$

where $\tilde{\mathcal{F}}_t := \tilde{\mathcal{F}}(\mathbf{m}_t, \mathbf{V}_t, \boldsymbol{\mu}_t, \boldsymbol{\Sigma}_t)$ and $\tilde{\beta}_t$, $\tilde{\alpha}_t$, $\tilde{\upsilon}_t$, and $\tilde{\gamma}_t$ are positive step-size parameters.

D. Mirror Descent Algorithm for the HetMOGP with CPM

For the HetMOGP with CPM, we follow a similar procedure carried out for the LMC. We use the MDA in Eq. (4) and the mean-parameters of distributions $q(\check{\mathbf{u}}_{d,j})$ and $q(\boldsymbol{\theta})$ defining $\boldsymbol{\rho}_{d,j} = \{\mathbf{m}_{d,j}, \mathbf{m}_{d,j}\mathbf{m}_{d,j}^\top + \mathbf{V}_{d,j}\}$ and $\boldsymbol{\eta} = \{\boldsymbol{\mu}, \boldsymbol{\mu}\boldsymbol{\mu}^\top + \boldsymbol{\Sigma}\}$ for minimising Eq. (17). Then, our algorithm for the CPM can be written as: $\boldsymbol{\eta}_{t+1}, \{\boldsymbol{\rho}_{d,j,t+1}\}_{d=1,j=1}^{D,J_d}$

$$= \underset{\boldsymbol{\eta}, \{\boldsymbol{\rho}_{d,j}\}_{d=1,j=1}^{D,J_d}}{\arg\min} \langle \boldsymbol{\eta}, \hat{\nabla}_{\boldsymbol{\eta}}\tilde{\mathcal{F}}_t\rangle + \frac{1}{\tilde{\alpha}_t}\text{KL}(\boldsymbol{\theta})_t - \frac{\tilde{\gamma}_t}{\tilde{\alpha}_t}\text{KL}(\boldsymbol{\theta})_{t-1} \tag{19}$$
$$+ \sum_{d,j=1}^{D,J_d}\Big[\langle \boldsymbol{\rho}_{d,j}, \hat{\nabla}_{\boldsymbol{\rho}_{d,j}}\tilde{\mathcal{F}}_t\rangle + \frac{1}{\tilde{\beta}_t}\text{KL}(\check{\mathbf{u}}_{d,j})_t - \frac{\tilde{\upsilon}_t}{\tilde{\beta}_t}\text{KL}(\check{\mathbf{u}}_{d,j})_{t-1}\Big],$$

where we have used the same variables $\tilde{\beta}_t$, $\tilde{\alpha}_t$, $\tilde{\upsilon}_t$, and $\tilde{\gamma}_t$ for the step-size parameters as in the LMC. This for the sake of a unified derivation of the FNG updates in the next subsection.

E. Fully Natural Gradient Updates

We can solve for Eq. (18) and (19) by computing derivatives w.r.t $\boldsymbol{\eta}$ and $\boldsymbol{\rho}$, and setting to zero. This way we obtain results similar to Eq. (5) and (6), we call them FNG updates:

$$\boldsymbol{\Sigma}_{t+1}^{-1} = \boldsymbol{\Sigma}_t^{-1} + 2\alpha_t\hat{\nabla}_{\boldsymbol{\Sigma}}\tilde{\mathcal{F}}_t \tag{20}$$

$$\boldsymbol{\mu}_{t+1} = \boldsymbol{\mu}_t - \alpha_t\boldsymbol{\Sigma}_{t+1}\hat{\nabla}_{\boldsymbol{\mu}}\tilde{\mathcal{F}}_t + \gamma_t\boldsymbol{\Sigma}_{t+1}\boldsymbol{\Sigma}_t^{-1}(\boldsymbol{\mu}_t - \boldsymbol{\mu}_{t-1}) \tag{21}$$

$$\mathbf{V}_{(\cdot),t+1}^{-1} = \mathbf{V}_{(\cdot),t}^{-1} + 2\beta_t\hat{\nabla}_{\mathbf{V}_{(\cdot)}}\tilde{\mathcal{F}}_t \tag{22}$$

$$\mathbf{m}_{(\cdot),t+1} = \mathbf{m}_{(\cdot),t} - \beta_t\mathbf{V}_{(\cdot),t+1}\hat{\nabla}_{\mathbf{m}_{(\cdot)}}\tilde{\mathcal{F}}_t + \upsilon_t\mathbf{V}_{(\cdot),t+1}\mathbf{V}_{(\cdot),t}^{-1}(\mathbf{m}_{(\cdot),t} - \mathbf{m}_{(\cdot),t-1}), \tag{23}$$

where we have defined, $\mathbf{m}_{(\cdot),t}$, as a way of referring to either $\mathbf{m}_{q,t}$ or $\mathbf{m}_{d,j,t}$ depending on the case of LMC or CPM. This also applies for $\mathbf{V}_{(\cdot),t}$ without loss of generality. And $\alpha_t = \tilde{\alpha}_t/(1-\tilde{\gamma}_t)$, $\beta_t = \tilde{\beta}_t/(1-\tilde{v}_t)$, $\gamma_t = \tilde{\gamma}_t/(1-\tilde{\gamma}_t)$ and $\upsilon_t = \tilde{v}_t/(1-\tilde{v}_t)$ are positive step-size parameters (see section VI of SM for details on the gradients derivation).

F. Implementation

In order to implement the proposed method, we have to take into account that our computational complexity depends on inverting the covariance matrix $\boldsymbol{\Sigma}$ in Eq. (20). Such complexity can be expressed as $\mathcal{O}((QMP + QP + QJ)^3)$ for the LMC, or $\mathcal{O}((JMP + QP + JP + QJ)^3)$ for the CPM, where the terms with the number of inducing points and/or input dimensionality tend to dominate the complexity in both cases. Likewise, the gradient $\hat{\nabla}_{\boldsymbol{\Sigma}}\tilde{\mathcal{F}}$ involves computing the Hessian $\hat{\nabla}^2_{\boldsymbol{\theta}\boldsymbol{\theta}}\tilde{\mathcal{L}}$ which can be computationally expensive and prone to suffer from non-positive definiteness. To alleviate those complexity issues we assume $\boldsymbol{\Sigma} = \text{diag}(\boldsymbol{\sigma}^2)$, where $\boldsymbol{\sigma}$ is a vector of standard deviations, and $\text{diag}(\boldsymbol{\sigma}^2)$ represents a matrix with the elements of $\boldsymbol{\sigma}^2$ on its diagonal. Additionally, we estimate the Hessian by means of the Gauss-Newton (GN) approximation $\hat{\nabla}^2_{\boldsymbol{\theta}\boldsymbol{\theta}}\tilde{\mathcal{L}} \approx \hat{\nabla}_{\boldsymbol{\theta}}\tilde{\mathcal{L}} \circ \hat{\nabla}_{\boldsymbol{\theta}}\tilde{\mathcal{L}}$ [43], [23]. The authors in [19] term this method as the variational RMSprop with momentum. They alternatively express Eq. (20) and (21) as:

$$\mathbf{p}_{t+1} = (1-\alpha_t)\,\mathbf{p}_t + \alpha_t \mathbb{E}_{q(\boldsymbol{\theta})}\left[\hat{\nabla}_{\boldsymbol{\theta}}\tilde{\mathcal{L}} \circ \hat{\nabla}_{\boldsymbol{\theta}}\tilde{\mathcal{L}}\right] \tag{24}$$

$$\begin{aligned}\boldsymbol{\mu}_{t+1} = \boldsymbol{\mu}_t &- \alpha_t(\mathbf{p}_{t+1} + \lambda_1\mathbf{1})^{-1} \circ \hat{\nabla}_{\boldsymbol{\mu}}\tilde{\mathcal{F}} \\ &+ \gamma_t(\mathbf{p}_t + \lambda_1\mathbf{1}) \circ (\mathbf{p}_{t+1} + \lambda_1\mathbf{1})^{-1} \circ \left(\boldsymbol{\mu}_t - \boldsymbol{\mu}_{t-1}\right),\end{aligned} \tag{25}$$

where $\hat{\nabla}_{\boldsymbol{\mu}}\tilde{\mathcal{F}} = (\mathbb{E}_{q(\boldsymbol{\theta})}\left[\hat{\nabla}_{\boldsymbol{\theta}}\tilde{\mathcal{L}}\right] + \lambda_1\boldsymbol{\mu}_t)$, $\circ$ represents an element-wise product and we have made a variable change defining a vector $\mathbf{p}_t := \boldsymbol{\sigma}_t^{-2} - \lambda_1\mathbf{1}$, with $\mathbf{1}$ as a vector of ones. The GN approximation provides stronger numerical stability by preventing that $\boldsymbol{\sigma}^2$ becomes negative. Also, using $\text{diag}(\boldsymbol{\sigma}^2)$ we reduce the computational complexity from $\mathcal{O}((QMP + QP + QJ)^3)$ to $\mathcal{O}(QMP + QP + QJ)$ for the LMC, or $\mathcal{O}((JMP + QP + JP + QJ)^3)$ to $\mathcal{O}(JMP + QP + JP + QJ)$ for the CPM (see section VII of SM for a pseudo-code implementation of the algorithm).

G. Predictive Distribution

In order to make predictions with the HetMOGP model, it is necessary to compute the following distribution: $p(\mathbf{y}_*|\mathbf{y}) \approx \int p(\mathbf{y}_*|\mathbf{f}_*)q(\mathbf{f}_*)d\mathbf{f}_*$, where $q(\mathbf{f}_*) = \prod_{d=1}^{D}\prod_{j=1}^{J_d} q(\mathbf{f}_{d,j,*})$. Given that we have introduced a variational distribution $q(\boldsymbol{\theta})$ over all hyper-parameters and inducing points of the model, we could apply a fully Bayesian treatment when calculating $q(\mathbf{f}_{d,j,*})$, either for the LMC $q(\mathbf{f}_{d,j,*}) = \int p(\mathbf{f}_{d,j,*}|\mathbf{u},\boldsymbol{\theta})q(\mathbf{u})q(\boldsymbol{\theta})d\boldsymbol{\theta}d\mathbf{u},$; or the CPM $q(\mathbf{f}_{d,j,*}) = \int p(\mathbf{f}_{d,j,*}|\check{\mathbf{u}},\boldsymbol{\theta})q(\check{\mathbf{u}})q(\boldsymbol{\theta})d\boldsymbol{\theta}d\check{\mathbf{u}}$. In practice, we found that $q(\boldsymbol{\theta})$'s covariance converged to very small values, in general $\text{diag}(\boldsymbol{\sigma}^2) \leq 10^{-15}$, and almost all the uncertainty information was concentrated on $q(\mathbf{u})$'s covariance for LMC, or $q(\check{\mathbf{u}})$'s covariance for CPM. Since making predictions with the equations above becomes computationally expensive and most of the uncertainty is represented by the distribution $q(\mathbf{u})$ or $q(\check{\mathbf{u}})$, we can trade-off the computation by using the MAP solution for $q(\boldsymbol{\theta})$ and completely integrating over the remaining distribution as follows: for LMC $q(\mathbf{f}_{d,j,*}) = \int p(\mathbf{f}_{d,j,*}|\mathbf{u},\boldsymbol{\theta} = \boldsymbol{\mu})q(\mathbf{u})d\mathbf{u}$, and for CPM $q(\mathbf{f}_{d,j,*}) = \int p(\mathbf{f}_{d,j,*}|\check{\mathbf{u}},\boldsymbol{\theta} = \boldsymbol{\mu})q(\check{\mathbf{u}})d\check{\mathbf{u}}$. When solving these integrals, we arrive to exactly the same solutions in Eq. (12) if we aim to make predictions for the LMC, or Eq. (16) if the case for CPM, where we simply have to evaluate the matrix covariances $\mathbf{K}_{\mathbf{f}_{d,j,*}\mathbf{u}}$ for LMC or $\mathbf{K}_{\mathbf{f}_{d,j,*}\check{\mathbf{u}}}$ for CPM, and $\mathbf{K}_{\mathbf{f}_{d,j,*}\mathbf{f}_{d,j,*}}$, all at the new inputs $\mathbf{X}_*$.

VII. EXPERIMENTS

In this section, we explore the performance of the proposed FNG method for jointly optimising all variational parameters, hyper-parameters and inducing points. We also test the hybrid (HYB) method proposed by [15], and compare the performance against Adam and SGD methods. We run experiments on different toy and real datasets, for all datasets we use a splitting of 75% and 25% for training and testing, respectively. The experiments consist on evaluating the method's performance when starting with 20 different initialisations of $q(\boldsymbol{\theta})$'s parameters to be optimised. We report the negative evidence lower bound (NELBO) shown in Eq. (11) for LMC and Eq. (15) for CPM over the training set, and the negative log predictive density (NLPD) error for the test set; this error metric takes into account the predictions' uncertainty [44].

A. Optimising the HetMOGP with LMC on Toy Data

We are interested in looking at the performance of HetMOGP with LMC when increasing the number of outputs, which implies rising also the heterogeneity of the output data. Given that the inducing points $\mathbf{z}$ have the same input space dimensionality and strongly affect the performance of sparse MOGPs, we are also interested in assessing the behaviour when increasing the input space dimensionality. For all the toy data examples we define an input space $\mathbf{X} \in [0,1]^{N\times P}$ with $N = 2\times 10^3$ observations, we analyse a set of different dimensions $P = \{1,2,3,4,5,10\}$. We assume a number of $Q = 3$ with an EQ kernel $k_q(\cdot,\cdot)$, and the inducing points $\mathbf{Z}_q \in \mathbb{R}^{M\times P}$, with $M = 80$. We run the experiments using mini-batches of 50 samples at each iteration, and we use one sample to approximate the expectations w.r.t $q(\boldsymbol{\theta})$ in Eq. (17). Below we describe the characteristics of each toy dataset.

Toy Data 1 (T1): the first toy example consists of three outputs $D = 3$; the first output is $y_1 \in \mathbb{R}$, the second $y_2 \in [0,1]$ and the third $y_3 \in \{0,1\}$. We use a Heteroscedastic-Gaussian (HetGaussian), a Beta and Bernoulli distribution as the likelihoods for each output, respectively.

Toy Data 2 (T2): the second toy example consists of five outputs $D = 5$, where the first three are exactly the same ones as T1 with the same likelihoods and the two additional ones are $y_4 \in [0,\infty]$, and $y_5 \in [0,\infty]$. We use a Gamma and an Exponential distribution for those latter outputs, respectively.

Toy Data 3 (T3): the third toy example consists of ten outputs $D = 10$, where the data type of the first five outputs

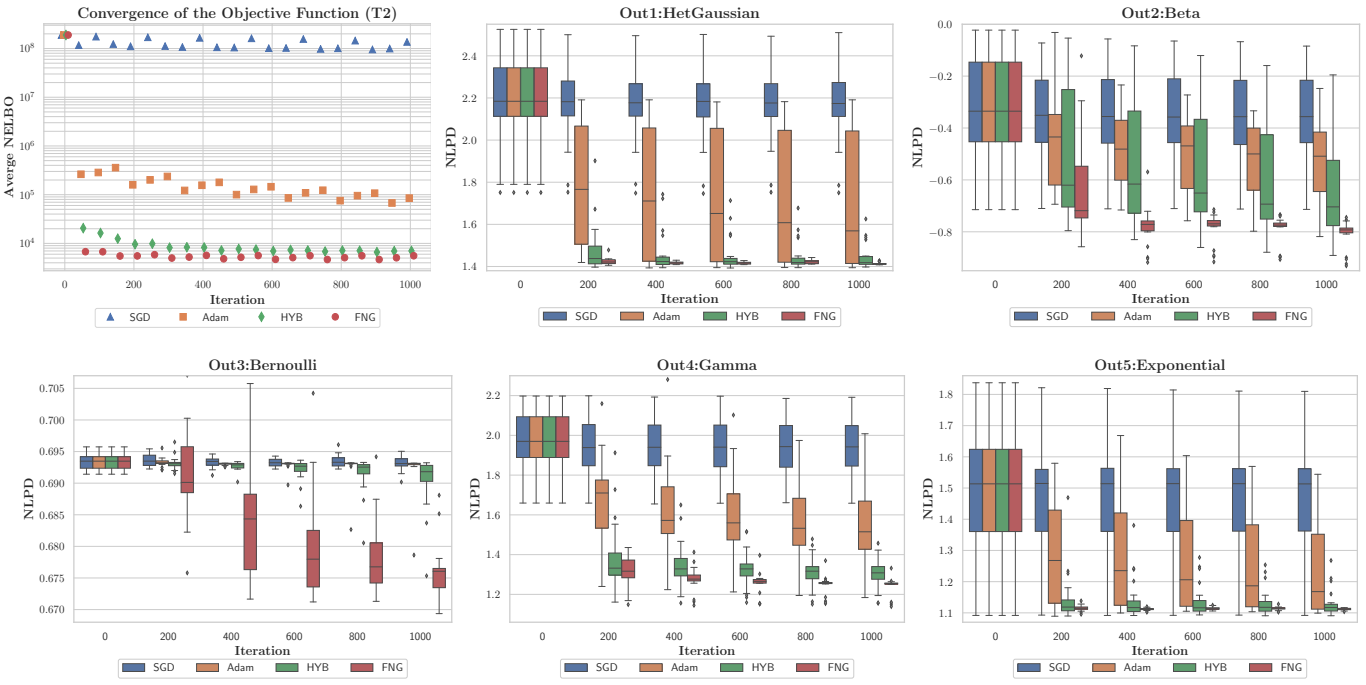


Fig. 2. Performance of the different inference methods on the T2 dataset for $P = 10$ using 20 different initialisations. The top left sub-figure shows the average NELBO convergence. The other sub-figures show the box-plot trending of the NLPD over the test set for each output. The box-plots at each iteration follow the legend's order from left to right: SGD, Adam, HYB and FNG. The isolated diamonds that appear in the outputs' graphs represent "outliers".

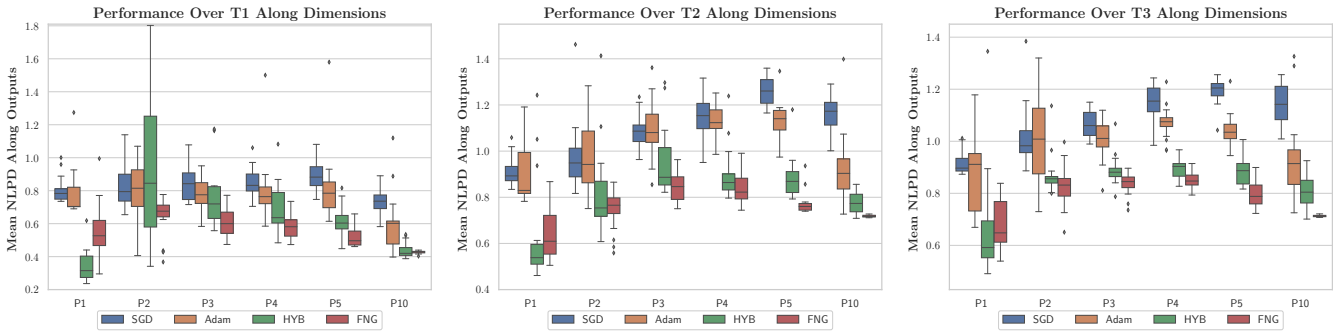


Fig. 3. Trending of the Mean NLPD along outputs for 20 different initialisations. Performance over: T1 (left), T2 (middle) and T3. Each sub-figure summarises the Mean NLPD of SGD, Adam, HYB and FNG methods along dimensions $P = \{1, 2, 3, 4, 5, 10\}$. The box-plots at each P follow the legend's order.

$\{y_d\}_{d=1}^{5}$ is exactly the same as T2. Also, the last five outputs $\{y_d\}_{d=6}^{10}$ share the same data type of the outputs in T2. We use the following ten likelihoods: HetGaussian, Beta, Bernoulli, Gamma, Exponential, Gaussian (with $\sigma_{\text{lik}} = 0.1$), Beta, Bernoulli, Gamma and Exponential. The data of the first five outputs is not the same as the last ones since the distributions of the generative model depend on the LCCs $a_{d,j,q}$ that generate the LPFs in Eq. (8). [2]

In order to visualise the convergence performance of the methods, we show results for T2 which consists of five outputs, where all of them are used in T3 and three of them in T1. We focus on the example for which $P = 10$ as the dimensionality. Fig. 2 shows the behaviour of the different algorithms over T2, where its top left sub-figure shows the average convergence of the NELBO after running 20 different initialisations. The figure shows that our FNG method tends to find a better local optima solution that minimises the NELBO followed by the HYB, Adam and SGD. The other sub-figures titled from Out1 to Out5 show the model's average NLPD achieved by each of the methods over the test set. From Fig. 2 we can notice that the SGD method does not progress much through the inference process achieving the poorest performance along the diverse outputs. The Adam method presents a big variance along the different outputs, showing its ability to explore feasible solutions, but arriving at many different poor local minima. Particularly, for the output 3, a Bernoulli likelihood, the method hardly moves from its initial NLPD value, showing in the figure a tiny variance without much improvement. This means the method lacks exploration and rapidly becomes trapped in a very poor local

[2]The code with all toy configurations is publicly available in the repository: https://github.com/juanjogg1987/Fully_Natural_Gradient_HetMOGP

minima. The HYB method in general shows smaller error bars than Adam and SGD. Indeed, it reaches low NLPD results for Gamma, HetGaussian and Exponential likelihoods, with similar behaviour to our FNG method in the two latter distributions. Although, it is difficult for HYB to achieve a proper NLPD performance on the distributions Beta and Bernoulli; though for the Beta distribution presents boxes with big variance meaning that it arrives to many different solutions, the NLPD's mean shows a trending to weak solutions. For the Bernoulli is deficient in exploring, so it also ends up in poor solutions. Our FNG method is consistent along the diverse outputs, usually tending to richer local minima solutions than the other methods. For the Beta and Gamma outputs, FNG makes a confident progress and even shows some "outliers" below its boxes which means that our method has the ability to eventually provide better solutions than the other methods. For the Bernoulli distribution, Fig. 2 shows that FNG presents big variance boxes, but with a tendency to much better solutions than the other methods. This big variance effect let us confirm that our proposed method actually takes advantage of the stochastic exploration induced over the model hyper-parameters for avoiding poor local minima solutions.

Figure 3 summarises the behaviour along the different dimensions P for each toy example. We notice from Fig. 3 that our FNG method achieves better test performance along distinct dimensions for all toy examples, followed by the HYB, Adam and SGD methods, though HYB presents better results than FNG when $P = 1$. All methods in general tend to present large variances for T1 which consists of three outputs, although this effect is reduced when the number of outputs is increased. Our FNG in general presents the smallest variance showing its ability to find better local minima even with many outputs. When increasing the dimensionality, the methods tend to degrade their performance, but the less sensitive to such behaviour are the HYB and FNG methods, where the latter, in general achieves the lowest mean NLPD along outputs for the different toy examples. Apart from the heterogeneous toy examples shown in this paper, we also ran experiments for dimensions higher than $P = 10$, although we noticed that all methods behaved similar except for the SGD which demands a very small step-size parameter that makes it progress slowly. We believe that the toy examples become difficult to control in such dimensions and the data observations become broadly scattered. We also explored experiments increasing the mini-batch size at each iteration, we noticed the gradient's stochasticity is reduced helping to increase the convergence rates of all methods, but the ones using NG perform better. When reducing the mini-batch size, our FNG method usually performs better than the others probably due to the fact that it additionally exploits the probability distribution $q(\boldsymbol{\theta})$, imposed over the hyper-parameters and inducing points.

B. Settings for Real Datasets Experiments

In this subsection we describe the different real datasets used for our experiments (See section X of SM for information about the web-pages where we took the datasets from).

HUMAN Dataset: the human behaviour dataset (HUMAN, $N_1, N_2 = 5 \times 10^3, N_3 = 21 \times 10^3, P = 1$, N_d associates the number of observations per output) contains information for monitoring psychiatric patients with a smartphone *app*. It consists of three outputs; the first monitors use/non-use of *WhatsApp*, $y_1 \in \{0, 1\}$, the second represents distance from the patient's home location, $y_2 \in \mathbb{R}$, and the third accounts for the number of smartphone active *apps*, we rescale it to $y_3 \in [0, 1]$. We use a Bernoulli, HetGaussian and a Beta distribution as the likelihoods for each output, respectively. We assume $Q = 5$ latent functions.

LONDON Dataset: the London dataset (LONDON, $N = 20 \times 10^3, P = 2$) is a register of properties sold in London in 2017; it consists of two outputs; the first represents house prices with $y_1 \in \mathbb{R}$ and the second accounts for the type of house. We use two types (flat/non-flat) with $y_2 \in \{0, 1\}$. We use a HetGaussian and Bernoulli distribution as the likelihood for each output respectively. We assume $Q = 3$ latent functions.

NAVAL Dataset: the naval dataset (NAVAL, $N = 11 \times 10^3, P = 15$) contains information of condition based maintenance of naval propulsion plants. it consists of two outputs: plant's compressor decay state coefficient and turbine decay state coefficient. We re-scaled both as $y_1, y_2 \in [0, 1]$, and used a Beta and Gamma distribution as the likelihood for each output respectively. We assume $Q = 4$ functions.

SARCOS Dataset: a seven degrees-of-freedom SARCOS anthropomorphic robot arm data, where the task is to map from a 21-dimensional input space (7 joint positions, 7 joint velocities, 7 joint accelerations) to the corresponding 7 joint output torques (SARCOS, $N = 44.5 \times 10^3, P = 21, D = 7$). We use a HetGaussian distribution as the likelihood for each output and assume $Q = 3$ functions.

MOCAP Dataset: a motion capture data for a walking subject (MOCAP7, $N = 744, P = 1, D = 40$). We use a HetGaussian distribution as the likelihood for each output and assume $Q = 3$ functions.

For the first three datasets, the number of inducing points per latent function is $M = 80$ and for each function $u_q(\cdot)$ we use an EQ kernel like Eq. (10). We run the experiments using mini-batches of 50 samples at each iteration, and we use one sample to approximate the expectations with regard to $q(\boldsymbol{\theta})$ in Eq. (17). For SARCOS we use mini-batches of 200 due to its large number of observations, and given that MOCAP7 is not a large dataset we use mini-batches of 5 with $M = 20$. To select Q, we applied a rule of thumb as follows: **I.** If $D <= 5$ set $Q = J$. We opted for this rule of thumb as a way to allow the HetMOGP model to have a high flexibility for modelling the data in presence of few outputs. **II.** if $D > 5$ set $Q = 3$. We chose this option for not overloading the computational complexity in presence of many outputs, though by setting $Q = 3$ we still can at least model low, medium and high length-scale resolutions from a dataset (see section IX of SM for details about the setting Q and the number J_d associated to each likelihood distribution).

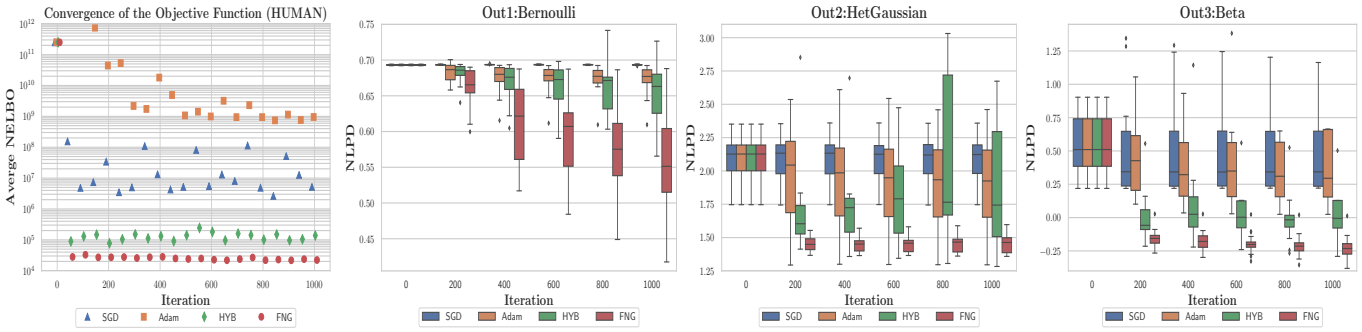


Fig. 4. Performance of the diverse inference methods on the HUMAN dataset using 20 different initialisations. The left sub-figure shows the average NELBO convergence of each method. The other sub-figures show the box-plot trending of the NLPD over the test set for each output. The box-plots at each iteration follow the legend's order from left to right: SGD, Adam, HYB and FNG. The isolated diamonds that appear in the outputs' graphs represent "outliers".

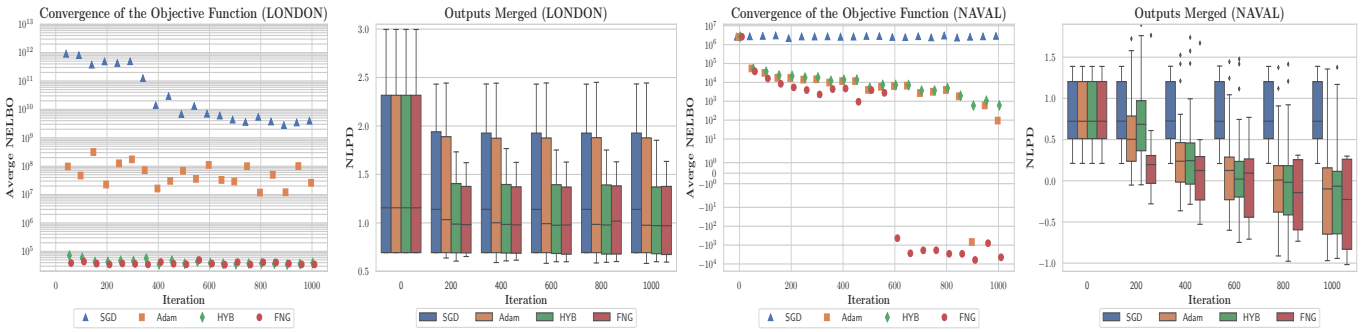


Fig. 5. Performance of the diverse inference methods on the LONDON and NAVAL datasets using 20 different initialisations. Sub-figures left and middle-left correspond to LONDON; middle-right and right refer to NAVAL. For each dataset we show the average NELBO convergence of each method and the box-plot trending of the NLPD over the test set across all output. The box-plots at each iteration follow the legend's order from left to right: SGD, Adam, HYB and FNG. The isolated diamonds that appear in the outputs' graphs represent "outliers".

C. Optimising the HetMOGP with LMC on Real Data

For this sub-section we explore our method's behaviour over the HetMOGP with LMC on HUMAN, LONDON, NAVAL, SARCOS and MOCAP7.

Figures 4 and 5 show the NELBO convergence over the training set, together with the average NLPD performance over the test set for HUMAN, LONDON and NAVAL data, respectively. We provide a merged NLPD along outputs for LONDON and NAVAL (see section VIII of SM for an analysis of each specific output). With regard to the convergence rate of the NELBO for HUMAN and LONDON datasets all methods converge similarly. Nonetheless, for the NAVAL dataset, our FNG approach presents a faster converge, followed by HYB and Adam; SGD remains without much progress along the iterations. For the HUMAN dataset, the SGD arrives at a better minimum than Adam, but the Adam's averaged NLPD is higher across outputs. HYB reaches consistent solutions being better than Adam and SGD, not only in the training process but also in testing along the HetGaussian and Beta outputs. Though, the Bernoulli output limits the overall performance of the method since there is not much improvement along the iterations. Our FNG method also shows a steady performance along outputs, commonly arriving to solutions with lower NLPD than the other methods. Our method presents the biggest variance for the Bernoulli output, implying strong exploration of the solutions' space for such likelihood, allowing it to reach the lowest average NLPD.

For the LONDON dataset, Adam converges to a richer minimum of the NELBO than SGD. Moreover, the NLPD for Adam is, on average, better than the SGD. The HYB and FNG arrive to a very similar value of the NELBO, both being better than Adam and SGD. HYB and FNG methods attain akin NLPD metrics, but the average and median trend of our approach is slightly better, being more robust to the initialisation than HYB method. The NLPD performance for the NAVAL dataset shows in Fig. 5 that the SGD method cannot make progress. We tried to set a bigger step-size, but usually increasing it derived in numerical problems due to ill-conditioning of the covariance matrices. The methods Adam and HYB show similar NLPD boxes, but at the end, Adam attains a slightly lower median with bigger variance than HYB. Regarding the NLPD, our FNG method ends up with a larger variance than SGD, Adam and HYB, but obtaining a much better mean and median trending than the others. Also, our FNG shows that the upper bar of the NLPD box is very close to the interquartile range, while the other methods present larger upper bars, this means that our FNG method concentrates in regions that provide better predictive performance than the other methods.

Fig. 6 shows the performance achieved by the different opti-

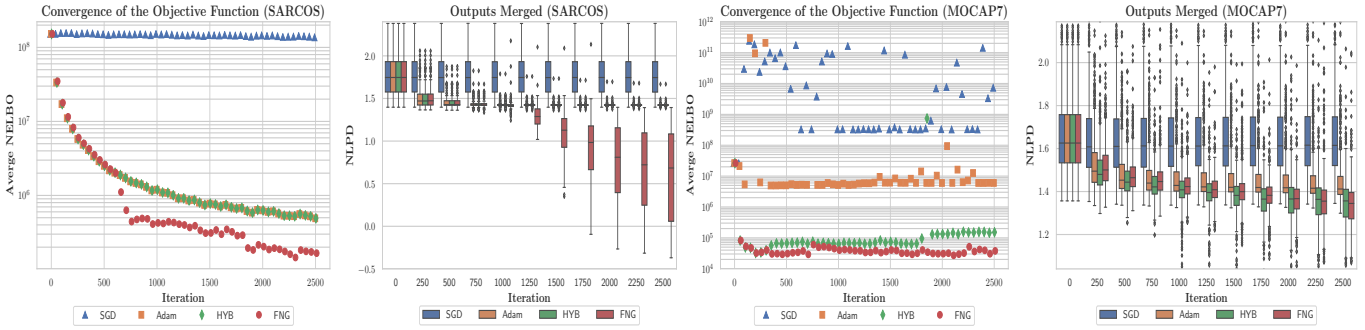


Fig. 6. Performance of the diverse inference methods on the SARCOS and MOCAP7 datasets using 20 different initialisations for HetMOGP with LMC. Sub-figures left and middle-left correspond to SARCOS; middle-right and right refer to MOCAP7. For each dataset we show the average NELBO convergence of each method and the box-plot trending of the NLPD over the test set across all output. The box-plots at each iteration follow the legend's order from left to right: SGD, Adam, HYB and FNG. The isolated diamonds that appear in the outputs' graphs represent "outliers".

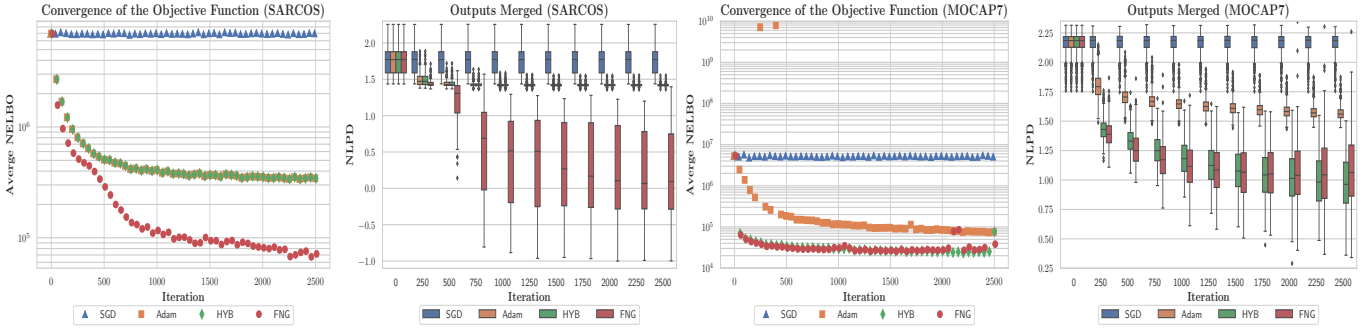


Fig. 7. Performance of the diverse inference methods on the SARCOS and MOCAP7 datasets using 20 different initialisations for HetMOGP with CPM. Sub-figures left and middle-left correspond to SARCOS; middle-right and right refer to MOCAP7. For each dataset we show the average NELBO convergence of each method and the box-plot trending of the NLPD over the test set across all output. The box-plots at each iteration follow the legend's order from left to right: SGD, Adam, HYB and FNG. The isolated diamonds that appear in the outputs' graphs represent "outliers".

misation methods for SARCOS and MOCAP7 datasets. Since these datasets present a high number of outputs we stacked the NLPD metric along all outputs. We can notice from the SARCOS experiment, in the first two sub-figures to the left, that SGD cannot improve much during the inference process both for NELBO and NLPD. Adam and HYB converge to the same local minima achieving the same average NELBO and NLPD trend, in contrast to our FNG method which attains the lowest values showing a better performance. Particularly in the SARCOS experiment, figures show how our method changes suddenly, around iteration 600, probably escaping from the same local minima to which Adam and HYB converged. For the MOCAP7's experiment, the two sub-figures to the right show that SGD slightly improves its performance in the inference process, while Adam reaches a much better minimum for the average NELBO. Although, these former methods do not perform better than HYB and FNG. The HYB and FNG behave similar before 500 iteration, but in the long term our FNG presents the lowest average NELBO. Likewise, the NLPD shows that HYB presents a slightly better trend than FNG at the early stages of the inference, but at the end, our FNG finds a better NLPD metric.

D. Optimising the HetMOGP with CPM on Real Data

In this subsection we show the performance of our FNG over the convolved MOGP for the model with heterogeneous likelihoods. We use the datasets SARCOS and MOCAP7 with a number of outputs of $D = 7$ and $D = 40$, respectively. Fig. 7 shows the performance of the different optimisation methods for fitting the HetMOGP with CPM over such datasets. Similarly to Fig. 6, we put together the NLPD metric across all outputs. The SARCOS' experiment shows that SGD does not improve much during the optimisation process. Adam and HYB seem to converge to a similar minimum value since the average NELBO and NLPD look very much alike. Otherwise, our FNG method shows to perform much better than the other methods achieving the lowest average NELBO. Also the NLPD trend exhibits a more robust performance over the test set. For MOCAP7, HYB and FNG behave similarly during the optimisation process showing almost the same average NELBO trend. Though, the former method presents a better behaviour when converging at the end. Our FNG method shows a better NLPD performance during the optimisation, but at the end HYB reaches a lower NLPD metric. Adam method accomplishes a poor minima in comparison to HYB and FNG, though a better one than SGD. We can notice from Fig. 6 and 7, both experiments over SARCOS and MOCAP7,

TABLE I
NLPD PERFORMANCE OF THE HETEROGENEOUS SCHEMES.

	LMC		CPM	
Dataset	**Median**	**Mean ± Std**	**Median**	**Mean ± Std**
LONDON	1.025	1.012 ± 0.331	0.986	0.983 ± 0.396
NAVAL	-0.310	-0.318 ± 0.475	-0.429	-0.454 ± 0.527
HUMAN	0.596	0.646 ± 0.764	0.330	0.529 ± 0.807
SARCOS	0.684	0.618 ± 0.581	0.096	0.169 ± 0.605
MOCAP9	0.752	0.774 ± 0.297	1.101	1.172 ± 0.386
MOCAP7	1.344	1.344 ± 0.170	1.078	1.141 ± 0.833
TRAFFIC	72.762	69.947 ± 25.466	68.214	74.866 ± 35.775

that the FNG presents similar convergence patterns in both the LMC and CPM, reaching better solutions than SGD, Adam and HYB. The next sub-section compares the performance between these two MOGP prior schemes.

E. Comparing MOGP priors for heterogeneous likelihoods

In this subsection we compare the MOGP models for heterogeneous likelihoods: the one based on the LMC and the one based on convolution processes. Table I presents the different NLPD metrics over a test set when using our proposed FNG scheme. Here, we make use of the real datasets from subsection VII-B, and we have additionally included two datasets for these experiments: TRAFFIC and MOCAP9 (see SM in section X for details about these additional datasets). The Table shows that the CPM in general outperforms the LMC for the different real datasets used in our experiments. The NLPD performance, for almost all datasets, shows a considerable improvement when using the convolutional approach, only for MOCAP9 the CPM did not present an improvement over the LMC. The NLPD metric for most of the datasets presents a median very close to the mean, unlike the HUMAN dataset which its mean differs much to the median, though having the median a better trend. Also, we can observe from the Table that generally the standard deviation is higher for the CPM. This is probably due to the additional hyper-parameter set, i.e., the length-scales associated to each smoothing kernel which introduce a larger parameters' space to be explored.

VIII. DISCUSSION AND CONCLUSION

In practice we noticed that some likelihoods (e.g. HetGaussian, Gamma) tend to strongly influence the value of the objective function (NELBO), so the optimisers HYB, Adam and SGD are prone to find solutions that focus on such kind of likelihoods, while neglecting the others with less influence, for instance a Bernoulli or Beta as shown in Fig. 2. On the other hand, our proposed scheme presents a more consistent performance achieving richer solutions across the different types of outputs' distributions. When increasing the outputs' size our FNG presented a consistent performance for TOY and real datasets like SARCOS and MOCAP7. We realised that HYB method presents a relevant performance for low input dimensionalities, but when the input dimensionality increases its performance degrades as shown for the TOY experiments when $P > 1$ and for the SARCOS experiment with $P = 21$. So, our method is the least sensitive to reduce its performance when increasing the input dimensionality, followed by the HYB and Adam methods. When using the SGD method we had to set a very small step-size parameter, because using large step-sizes makes the model to easily become ill-conditioned. Also, we observed that our FNG is a suitable scheme for training another type of MOGP model like the CPM. Indeed, our experiments show that the CPM can also be trained under a SVI attaining better performance than a HetMOGP based on a LMC. The new HetMOGP model based on convolution processes differs from the original one based on a LMC in the way the inducing variables are introduced. For the LMC the inducing variables are additional evaluations of the functions $u_q(\cdot)$, while for the CPM the inducing variables are additional evaluations of the functions $f_{d,j}(\cdot)$. We implemented the version of CPM using the same style of inducing variables as the LMC though, in practice, we realised that the assumption commonly used in the literature for the posterior, i.e., $q(\mathbf{f}, \mathbf{u}) = p(\mathbf{f}|\mathbf{u})q(\mathbf{u})$ is not sufficiently flexible to fit the LPFs and limits the SVI implementation. Therefore, we opted for the inducing variables procedure which does support the assumption $q(\mathbf{f}, \check{\mathbf{u}}) = p(\mathbf{f}|\check{\mathbf{u}})q(\check{\mathbf{u}})$.

The VO bound in Eq. (17) can be seen as a fully Bayesian treatment of the HetMOGP, where the model's parameters and hyper-parameters follow a prior distribution, where the positive constraint variables follow a Log-Normal distribution and the non-constraint ones follow a Gaussian distribution. Our VO bound benefits from the assumption of a Gaussian exploratory (or posterior) distribution for deriving in a closed-form our FNG optimisation scheme. This scheme helps to find solutions that directly improve the predictive capabilities of the HetMOGP model. For instance, since the inducing points' size is directly influenced by the input dimensionality, we believe that applying exploration over them helps to improve the model performance for high input dimensionalities as shown in the experiments.

In this paper, we have shown how a fully natural gradient scheme improves optimisation of a heterogeneous MOGP model by generally reaching better local optima solutions with higher test performance rates than HYB, Adam and SGD methods. We have shown that our FNG scheme provides rich local optima solutions, even when increasing the dimensionality of the input and/or output space. Furthermore, we have provided a novel extension of a stochastic scalable Heterogeneous MOGP model based on convolution processes. Our FNG method may also be an alternative tool for improving optimisation over a single output GP model. As a future work, it might be worth exploring the behaviour of the proposed scheme over other type of GP models, for instance Deep GPs [16]. Likewise, it would be relevant to explore a scalable way to implement the method using a full covariance matrix $\boldsymbol{\Sigma}$ which can exploit full correlation between all hyper-parameters. Modelling multi-modal data is another venue for future work. One might potentially want to combine ideas from the work in [45], with the HetMOGP model and the optimisation schemes proposed in this work. Also, ideas for the model selection problem of the number Q of latent functions, like the ones based on Indian buffet processes [46], [47] can be further investigated in the particular context of MOGPs with Heterogeneous outputs.

ACKNOWLEDGMENT

The authors would like to thank Emtiyaz Khan for the feedback related to mathematical aspects of the method and to the authors of [5] for lending their datasets HUMAN and LONDON. The authors would also like to thank Innovate UK for funding under the project 104316. JJG is being funded by a scholarship from the Dept. of Comp. Science, University of Sheffield. MAA has been financed by the EPSRC Research Projects EP/R034303/1 and EP/T00343X/1.

REFERENCES

[1] M. A. Álvarez, L. Rosasco, and N. D. Lawrence, "Kernels for vector-valued functions: A review," *Found. Trends Mach. Learn.*, vol. 4, no. 3, pp. 195–266, Mar. 2012.
[2] A. G. Journel and C. J. Huijbregts, *Mining Geostatistics*. Academic Press, London, 1978.
[3] D. Higdon, "Space and space-time modeling using process convolutions." Springer London, 2002, pp. 37–56.
[4] M. A. Álvarez and N. D. Lawrence, "Computationally efficient convolved multiple output Gaussian processes," *JMLR*, vol. 12, p. 1459–1500, 2011.
[5] P. Moreno-Muñoz, A. Artés-Rodríguez, and M. A. Álvarez, "Heterogeneous multi-output Gaussian process prediction," in *NeurIPS*, 2018, pp. 6712–6721.
[6] A. Saul, J. Hensman, A. Vehtai, and N. D. Lawrence, "Chained Gaussian processes," *AISTATS*, 2016.
[7] J. Hensman, N. Fusi, and N. D. Lawrence, "Gaussian processes for big data," in *UAI*, 2013.
[8] J. Quiñonero-Candela and C. E. Rasmussen, "A unifying view of sparse approximate Gaussian process regression," *JMLR*, vol. 6, pp. 1939–1959, 2005.
[9] E. Snelson and Z. Ghahramani, "Sparse Gaussian processes using pseudo-inputs," in *NIPS*, 2006, pp. 1257–1264.
[10] C. E. Rasmussen, *Gaussian processes for machine learning*. MIT Press, 2006.
[11] M. van der Wilk, "Sparse Gaussian Process Approximations and Applications," *PhD Thesis, University of Cambridge*, 2018.
[12] S. Amari, "Natural gradient works efficiently in learning," *Neural Computation*, vol. 10, no. 2, pp. 251–276, Feb. 1998.
[13] M. E. Khan, P. Baque, F. Fleuret, and P. Fua, "Kullback-Leibler proximal variational inference," in *NIPS*, 2015, pp. 3402–3410.
[14] M. E. Khan and W. Lin, "Conjugate-computation variational inference: Converting variational inference in non-conjugate models to inferences in conjugate models," in *AISTATS*, 2017.
[15] H. Salimbeni, S. Eleftheriadis, and J. Hensman, "Natural gradients in practice: Non-conjugate variational inference in Gaussian process models," in *AISTATS*, 2018, pp. 689–697.
[16] H. Salimbeni, V. Dutordoir, J. Hensman, and M. Deisenroth, "Deep Gaussian processes with importance-weighted variational inference," in *ICML*, 2019, pp. 5589–5598.
[17] J. Staines and D. Barber, "Optimization by variational bounding," in *ESANN*, 2013.
[18] M. E. Khan, W. Lin, V. Tangkaratt, Z. Liu, and D. Nielsen, "Variational adaptive-Newton method for explorative learning," *CoRR*, vol. abs/1711.05560, 2017.
[19] M. E. Khan, D. Nielsen, V. Tangkaratt, W. Lin, Y. Gal, and A. Srivastava, "Fast and scalable Bayesian deep learning by weight-perturbation in adam," in *ICML*, 2018, pp. 2616–2625.
[20] E. Chong and S. Zak, *An Introduction to Optimization*, ser. Wiley Series in Discrete Mathematics and Optimization. Wiley, 2013.
[21] D. Wierstra, T. Schaul, T. Glasmachers, Y. Sun, J. Peters, and J. Schmidhuber, "Natural evolution strategies," *J. Mach. Learn. Res.*, vol. 15, no. 1, pp. 949–980, Jan. 2014.
[22] J. Hensman, A. G. d. G. Matthews, M. Filippone, and Z. Ghahramani, "MCMC for variationally sparse Gaussian processes," in *NIPS*. MIT Press, 2015, pp. 1648–1656.
[23] M. E. Khan, Z. Liu, V. Tangkaratt, and Y. Gal, "Vprop: Variational inference using RMSprop," *CoRR*, vol. abs/1712.01038, 2017.
[24] K. P. Murphy, *Machine Learning: A Probabilistic Perspective*. MIT Press, 2013.
[25] C. M. Bishop, *Pattern Recognition and Machine Learning*. Springer-Verlag New York, Inc., 2006.
[26] D. M. Blei, A. Kucukelbir, and J. D. McAuliffe, "Variational inference: A review for statisticians," *JASA*, vol. 112, no. 518, pp. 859–877, 2017.
[27] M. I. Jordan, Z. Ghahramani, T. S. Jaakkola, and L. K. Saul, "An introduction to variational methods for graphical models," *Machine Learning*, vol. 37, no. 2, pp. 183–233, Nov 1999.
[28] G. Raskutti and S. Mukherjee, "The information geometry of mirror descent," *IEEE Trans. Information Theory*, vol. 61, no. 3, pp. 1451–1457, 2015.
[29] M. A. Osborne, A. Rogers, S. Ramchurn, S. J. Roberts, and N. R. Jennings, "Towards real-time information processing of sensor network data using computationally efficient multi-output Gaussian processes," in *IPSN*, 2008, pp. 109–120.
[30] J. Zhao and S. Sun, "Variational dependent multi-output Gaussian process dynamical systems," *JMLR*, vol. 17, no. 1, p. 4134–4169, 2016.
[31] T. Cohn and L. Specia, "Modelling annotator bias with multi-task Gaussian processes: An application to machine translation quality estimation," in *ACL (Volume 1: Long Papers)*, 2013, pp. 32–42.
[32] S. Conti, J. P. Gosling, J. E. Oakley, and A. O'Hagan, "Gaussian process emulation of dynamic computer codes," *Biometrika*, vol. 96, no. 3, pp. 663–676, 2009.
[33] S. Conti and A. O'Hagan, "Bayesian emulation of complex multi-output and dynamic computer models," *Journal of Statistical Planning and Inference*, vol. 140, no. 3, pp. 640 – 651, 2010.
[34] P. Boyle and M. Frean, "Dependent Gaussian processes," in *NIPS*. MIT Press, 2005, pp. 217–224.
[35] G. Parra and F. Tobar, "Spectral mixture kernels for multi-output Gaussian processes," in *NIPS*, 2017, pp. 6681–6690.
[36] K. Chen, T. van Laarhoven, P. Groot, J. Chen, and E. Marchiori, "Multioutput convolution spectral mixture for Gaussian processes," *IEEE Trans. NNLS*, pp. 1–12, 2019.
[37] T. V. Nguyen and E. V. Bonilla, "Collaborative multi-output gaussian processes," in *UAI*, 2014, p. 643–652.
[38] A. Dezfouli and E. V. Bonilla, "Scalable inference for Gaussian process models with black-box likelihoods," in *NIPS*, 2015, pp. 1414–1422.
[39] A. G. Wilson, D. A. Knowles, and Z. Ghahramani, "Gaussian process regression networks," in *ICML*, 2012, p. 1139–1146.
[40] J. Requeima, W. Tebbutt, W. Bruinsma, and R. E. Turner, "The Gaussian process autoregressive regression model (GPAR)," vol. 89. PMLR, 2019, pp. 1860–1869.
[41] M. K. Titsias, "Variational learning of inducing variables in sparse Gaussian processes," in *AISTATS*, 2009.
[42] J. Hensman, A. G. de G. Matthews, and Z. Ghahramani, "Scalable variational Gaussian process classification." in *AISTATS*, 2015.
[43] D. Bertsekas, *Nonlinear Programming*. Athena Scientific, 1999.
[44] J. Quiñonero-Candela, C. E. Rasmussen, F. Sinz, O. Bousquet, and B. Schölkopf, "Evaluating predictive uncertainty challenge." Springer Berlin Heidelberg, 2006, pp. 1–27.
[45] M. Lázaro-Gredilla, S. Van Vaerenbergh, and N. D. Lawrence, "Overlapping mixtures of Gaussian processes for the data association problem," *Pattern Recogn.*, vol. 45, no. 4, p. 1386–1395, 2012.
[46] C. Guarnizo, M. A. Álvarez, and A. A. Orozco, "Indian buffet process for model selection in latent force models," in *CIARP*. Springer, 2015.
[47] A. Tong and J. Choi, "Discovering latent covariance structures for multiple time series," vol. 97. PMLR, 2019, pp. 6285–6294.

Juan-José Giraldo received a degree in Electronics Engineering (B. Eng.) with Honours, from Universidad del Quindío, Colombia in 2009, a master degree in Electrical Engineering (M. Eng.) from Universidad Tecnológica de Pereira, Colombia in 2015. Currently, Mr. Giraldo is a Ph.D student in Comp. Science at the University of Sheffield, UK.

Mauricio A. Álvarez received a degree in Electronics Engineering (B. Eng.) with Honours, from Universidad Nacional de Colombia in 2004, a master degree in Electrical Engineering (M. Eng.) from Universidad Tecnológica de Pereira, Colombia in 2006, and a Ph.D. degree in Comp. Science from The University of Manchester, UK, in 2011. After finishing his Ph.D., Dr. Álvarez López joined the Dept. of Electrical Engineering at Universidad Tecnológica de Pereira, Colombia, where he was appointed as an Associate Prof. until Dec 2016. From Jan, 2017, Dr. Álvarez López joined The University of Sheffield, UK. He is currently a Senior Lecturer in ML at the Dept. of Comp. Science.

Appendix L

Paper ii: Correlated Chained Gaussian Processes for Datasets with Multiple Annotators

Paper under review in the Journal IEEE Transactions on Neural Networks and Learning Systems.

Correlated Chained Gaussian Processes for Datasets with Multiple Annotators

J. Gil-González, J. Giraldo, A. Álvarez-Meza, A. Orozco-Gutiérrez, and M. A. Álvarez

***Abstract*—The labeling process within a supervised learning task is usually carried out by an expert, which provides the ground truth (gold standard) for each sample. However, in many real-world applications, we typically have access to annotations provided by crowds holding different and unknown expertise levels. Learning from crowds intends to configure machine learning paradigms in the presence of multi-labelers, residing on two key assumptions: the labeler's performance does not depend on the input space, and independence among the annotators is imposed. Here, we propose the correlated chained Gaussian processes from multiple annotators–(CCGPMA) approach, which models each annotator's performance as a function of the input space and exploits the correlations among experts. Experimental results associated with classification and regression tasks show that our CCGPMA performs better modeling of the labelers' behaviour, indicating that it consistently outperforms other state-of-the-art learning from crowds approaches.**

***Index Terms*—Multiple annotators, Correlated Chained Gaussian Processes, Variational inference, Semi-parametric latent factor model.**

I. Introduction

SUPERVISED learning requires that a domain expert labels the instances to built the gold standard (ground truth) (1). Yet, experts are scarce, or their time is expensive, not mentioning that the labeling task is tedious and time-consuming (2). As an alternative, the labeling is distributed through multiple heterogeneous annotators, who annotate part of the whole dataset by providing their version of the hidden ground truth (3). Recently, crowdsourcing platforms, i.e., Amazon Mechanical Turk– (AMT)[1], have been introduced to capture labels from multiple sources on large datasets efficiently. The attractiveness of these platforms lies in that, at a low cost, it is possible to obtain suitable quality labels. Indeed, in some cases, such a labeling process can compete with those provided by experts (4). However, in such multi-labeler scenario, each instance is matched with multiple annotations provided by different sources with unknown and diverse expertise, being difficult to apply traditional supervised learning algorithms (5). In this sense, *learning from crowds* has been introduced as a general framework from two main perspectives: to fit the labels from multiple annotators or to adapt the supervised learning algorithms (6).

J. Gil-González and A. Orozco are with the Universidad Tecnológica de Pereira, Colombia, 660003, e-mail:{jugil,aaog}@utp.edu.co

J. Giraldo and M. A.Álvarez are with the University of Sheffield, UK. email: {jjgiraldogutierrez1,mauricio.alvarez}@sheffield.ac.uk

A. Álvarez is with the Universidad Nacional de Colombia sede Manizales, 170001, Colombia. email: amalvarezme@unal.edu.co

[1]https://www.mturk.com/

The first approach is known in the literature as "label aggregation" or "truth inference", comprising the computation of a single hard label per sample as an estimation of the ground truth. The hard labels are then used to feed a standard supervised learning algorithm (7). The straightforward method is the so-called majority voting–(MV), and it has been used in different multi-labeler problems due to its simplicity (8). Still, MV assumes homogeneity in annotators' reliability, which is hardly feasible in real applications, e.g., experts vs. spammers. Furthermore, the consensus is profoundly impacted by incorrect labels and outliers (3). Conversely, more elaborated models have been considered to improve the estimation of the correct tag through the well-known Expectation-Maximization–(EM) framework and by facing the imbalanced labeling issue (9; 8).

The second approach jointly trains the supervised learning algorithm and models the annotators' behavior. It has been shown that such strategies lead to better performance compared to the ones belonging to label aggregation. Thus, the features used to train the learning algorithm provide valuable information to puzzle out the ground truth (10). The most representative work in this area is exposed in (11), which offers an EM-based framework to learn the parameters of a logistic regression classifier and model the annotators' behavior by computing their sensitivities and specificities. In fact, such a technique has inspired several models in the context of multi-labeler scenarios, including binary classification (12; 10), multi-class discrimination (7; 13), regression (14; 15), and sequence labeling (16). Furthermore, some works have addressed the multi-labeler problem using deep learning approaches typically including an extra layer that codes the annotators' information (17; 18; 19).

Two main issues are still unsolved in the context of learning from crowds (20): we need to code the relationships between the input features and the labelers' performance while revealing relevant annotators' interdependencies. In general, the annotators' behavior is parametrized through a homogeneous constraint across the input samples. The latter assumption is not correct since an expert makes decisions based not only on his/her expertise but also on the features observed from raw data (11). Besides, it is widespread to consider independence in the annotators' labels, aiming to reduce the complexity of the model (21), or based on the fact that it is plausible to guarantee that each labeler performs the annotation process individually (22). However, this assumption is not true since there may exist correlations among the annotators (23). For example, if the sources are humans, the independence assumption is hardly feasible because knowledge is a social construction; then, people's decisions will be correlated because they share information or belong to a particular school of

thought (24; 25). Now, if we consider that the sources are algorithms, where some of them gather the same math principles, there likely exists a correlation in their labels (26).

In this work, we propose a probabilistic model, named the correlated chained Gaussian Processes for multiple annotators (CCGPMA), to jointly build a prediction algorithm applicable to classification and regression tasks. CCGPMA is based on the chained GPs model–(CGP) (27), which is a Multi-GPs framework where the parameters of an arbitrary likelihood function are modeled with multiple independent GPs (one GP prior per parameter). Unlike CGP, we consider that multiple correlated GPs model the likelihood's parameters. For doing so, we take as a basis the ideas from a Multi-output GP–(MOGP) regression (28), where each output is coded as a weighted sum of shared latent functions via a semi-parametric latent factor model–(SLFM) (29). In contrast to the MOGP, we do not have multiple outputs but multiple functions chained to the given likelihood parameters. From the multiple annotators' point of view, the likelihood parameters are related to the labelers' behavior; thereby, CCGPMA models the labelers' behavior as a function of the input features while also taking into account annotators' interdependencies. Moreover, our proposal is based on the so-called inducing variables framework (30), in combination with stochastic variational inference (31). To the best of our knowledge, this is the first attempt to build a probabilistic approach to model the labelers' behavior as a function of the input features while also considering annotators' interdependencies. Achieved results, using both simulated and real-world data, show how our method can deal with both regression and classification problems from multi-labelers data.

The remainder is organized as follows. Section 2 exposes the related work and the main contributions of the proposal. Section 3 describes the methods. Sections 4 and 5 present the experiments and discuss the results. Finally, Section 6 outlines the conclusions and future work.

II. Related work and main contributions

Most of the learning from crowds-based methods aim to model the annotators' behavior based on the accuracy (32), the confusion matrix (13), the error variance (11), and the bias (15). Concerning this, the expert parameters are modeled as fixed points (12), or as random variables, where it is considered that such parameters are homogeneous across the input data (7).

The first attempt to analyze the relationship between the annotators' parameters and the input features is the work in (23). The authors propose an approach for binary classification with multiple labelers, where the input data is represented by a defined cluster using a Gaussian Mixture Model–(GMM). The approach assumes that the annotators exhibit a particular performance measured in terms of sensitivity and specificity for each group. However, the model does not consider the information from multiple experts as an input for the GMM, yielding variations in the labelers' parameters. Similarly, in (33), the authors propose a binary classification algorithm that employs two probability models to code the annotators' performance as a function of the input space, namely a Bernoulli and a Gaussian distribution. The parameters of these distributions are computed via Logistic regression. Nonetheless, a linear dependence between the labeler expertise and the input space is assumed, which may not be appropriate because of the data structure's nonlinearities. For example, if we consider online annotators assessing some documents, they may have different labeling accuracy. Such differences may rely on whether they are more familiar with some specific topics related to studied documents (34). Authors in (35) offer a GP-based regression with multiple annotators. An additional GP models the annotators' parameters as a nonlinear function of the input space. Yet, the inference is carried out based on maximum a posteriori (MAP), without including the uncertainty of the posterior distribution.

On the other hand, it has been shown that the relaxation of the annotators' independence restriction can improve the ground truth estimation (23; 20). To the best of our knowledge, only two works address such an issue. First, the authors in (26) describe an approach to deal with regression problems, where the labelers' behavior is modeled using a multivariate Gaussian distribution. Thus, the annotators' interdependencies are coded in the covariance matrix. Further, in (36), the authors propose a binary classification method based on a weighted combination of classifiers. In turn, the weights are estimated by using a kernel alignment-based algorithm considering dependencies among the labelers.

Here, we propose a GPs-based framework to face classification and regression settings with multiple annotators. Our proposal follows the line of the works in (12; 14; 10; 7; 37) in the sense that we are modeling the unknown ground truth trough a GP prior. However, while such approaches code the annotators' parameters as fixed points (12; 14); or as random variables (10; 7; 37); we model them as random processes to take into account dependencies between the input space and the labelers' behavior. Besides, our CCGPMA shares some similarities with the works in (33; 35), because we aim to model the dependencies between the input features and the labelers' performance. Our method is also similar to the works in (26; 36), because they assume dependencies in the annotators' labels. In contrast, CCGPMA is the only one that includes both assumptions to code the annotators' behavior. Of note, we highlight that our proposal codes inconsistent annotations, being robust against outliers. Namely, CCGPMA can estimate the annotators' performance for every region in the input space; meanwhile, state-of-the-art techniques assess it based on a conventional averaging (15; 7; 10). Table I summarizes the key insights of our CCGPMA and state-of-the-art approaches.

III. Methods

A. Chained Gaussian processes

Let us consider an input-output dataset $\mathcal{D}=\{\boldsymbol{X}\in\mathcal{X},\boldsymbol{y}\in\mathcal{Y}\}$, where $\boldsymbol{X}=\left\{\boldsymbol{x}_n\in\mathcal{X}\subseteq\mathbb{R}^P\right\}_{n=1}^N$ and $\boldsymbol{y}=\{y_n\in\mathcal{Y}\}_{n=1}^N$. In turn, let a GP be a collection of random variables $f(\boldsymbol{x})$ indexed by the input samples $\boldsymbol{x}\in\mathcal{X}$ holding a joint multivariate Gaussian distribution (39). A GP is defined by its mean $m(\boldsymbol{x})=\mathbb{E}[f(\boldsymbol{x})]$ (we consider $m(\boldsymbol{x})=0$) and covariance function $\kappa_f(\boldsymbol{x},\boldsymbol{x'})=\mathbb{E}[(f(\boldsymbol{x})-m(\boldsymbol{x}))(f(\boldsymbol{x'})-m(\boldsymbol{x'}))]$, where $\kappa_f:\mathcal{X}\times\mathcal{X}\to\mathbb{R}$ is a given kernel function and $\boldsymbol{x'}\in\mathcal{X}$, yielding:

TABLE I
SURVEY OF RELEVANT SUPERVISED LEARNING MODELS DEVOTED TO MULTIPLE ANNOTATORS.

Source	Data type	Type of model	Modeling the annotator's expertise	Expertise as a function of the input space	Modeling the annotators' inter-dependencies
Raykar et al., 2010 (11)	Regression-Binary-Categorical	Probabilistic	✓	✗	✗
Zhang and Obradovic, 2011 (23)	Binary	Probabilistic	✓	✓	✗
Xiao et al., 2013 (35)	Regression	Probabilistic	✓	✓	✗
Yan et al., 2014 (33)	Binary	Probabilistic	✓	✓	✗
Wang and Bi, 2016 (34)	Binary	Deterministic	✓	✓	✗
Rodrigues et al., 2017 (15)	Regression-Binary-Categorical	Probabilistic	✓	✗	✗
Gil-Gonzalez et al., 2018 (36)	Binary	Deterministic	✓	✗	✓
Hua et al., 2018 (38)	Binary-Categorical	Deterministic	✓	✗	✗
Ruiz et al., 2019 (10)	Binary	Probabilistic	✓	✗	✗
Morales- Álvarez et al., 2019 (7)	Binary	Probabilistic	✓	✗	✗
Zhu et al., 2019 (26)	Regression	Probabilistic	✓	✗	✓
Proposal-(CCGPMA)	Regression-Binary-Categorical	Probabilistic	✓	✓	✓

$$f(\boldsymbol{x}) \sim \mathcal{GP}(0, \kappa_f(\boldsymbol{x}, \boldsymbol{x'})). \tag{1}$$

If we consider the finite set of inputs in $\boldsymbol{X}$, then $\boldsymbol{f} = [f(\boldsymbol{x}_1), \ldots, f(\boldsymbol{x}_N)]^\top \in \mathbb{R}^N$ is drawn for a multivariate Gaussian distribution $\boldsymbol{f} \sim \mathcal{N}(\boldsymbol{f}|\boldsymbol{0}, \boldsymbol{K_{ff}})$, where $\boldsymbol{K_{ff}} \in \mathbb{R}^{N\times N}$ is the covariance matrix formed by the evaluation of $\kappa_f(\cdot,\cdot)$ over the input set $\boldsymbol{X}$.

Accordingly, using GPs for modeling the input-output data collection $\mathcal{D}$ consists of constructing a joint distribution between a given likelihood function and one or multiple GP-based priors. To code each likelihood parameter as a random process, we employ the so-called chained GP–(CGP) that attaches such parameters to multiple independent GP priors, as follows (27):

$$p(\boldsymbol{y}, \hat{\boldsymbol{f}}|\boldsymbol{X}) = \prod_{n=1}^{N} p(y_n|\theta_1(\boldsymbol{x}_n), \ldots, \theta_J(\boldsymbol{x}_n)) \times \cdots \\ \cdots \times \prod_{j=1}^{J} \mathcal{N}(\boldsymbol{f}_j|\boldsymbol{0}, \boldsymbol{K}_{\boldsymbol{f}_j\boldsymbol{f}_j}), \tag{2}$$

where each $\{\theta_j(\boldsymbol{x}) \in \mathcal{M}_j\}_{j=1}^J$ represents the likelihood's parameters, being $J \in \mathbb{N}$ the number of parameters to represent the likelihood. Besides, each $\theta_j(\boldsymbol{x})$ holds a non-linear mapping from a GP prior, e.g., $\theta_j(\boldsymbol{x}) = h_j(f_j(\boldsymbol{x}))$, where $h_j : \mathbb{R} \to \mathcal{M}_j$ is a deterministic function that maps each latent function–(LF) $f_j(\boldsymbol{x})$, to the appropriate domain $\mathcal{M}_j$. Moreover, $\boldsymbol{f}_j = [f_j(\boldsymbol{x}_1), \ldots, f_j(\boldsymbol{x}_N)]^\top \in \mathbb{R}^N$ is a LF vector that follows a GP prior, and $\hat{\boldsymbol{f}} = [\boldsymbol{f}_1, \ldots, \boldsymbol{f}_J]^\top \in \mathbb{R}^{NJ}$. $\boldsymbol{K}_{\boldsymbol{f}_j\boldsymbol{f}_j} \in \mathbb{R}^{N\times N}$ is the covariance matrix belonging to the j-th GP prior, which is computed based on the kernel function $\kappa_j : \mathcal{X} \times \mathcal{X} \to \mathbb{R}$. The non-parametric formulation of a GP introduces computational loads through the inference process. For instance, considering that the dataset $\mathcal{D}$ configures a regression problem, a GP modeling involves a computational complexity of $\mathcal{O}(N^3)$ to invert the matrix $\boldsymbol{K}_{\boldsymbol{f}_j\boldsymbol{f}_j}$ (39). A common approach to reduce such computational complexity is to augment the GP prior with a set of $M << N$ inducing variables (40), $\boldsymbol{u}_j = [f_j(\boldsymbol{z}_1^j), \ldots, f_j(\boldsymbol{z}_M^j)]^\top \in \mathbb{R}^M$ through additional evaluations of $f_j(\cdot)$ at unknown locations $\boldsymbol{Z}_j = [\boldsymbol{z}_1^j, \ldots, \boldsymbol{z}_M^j] \in \mathbb{R}^{M\times P}$, which decreases the GP's computational complexity to $\mathcal{O}(NM^2)$. Further, the following augmented GP prior arises:

$$p(\boldsymbol{f}_j, \boldsymbol{u}_j) = \mathcal{N}\left(\begin{bmatrix}\boldsymbol{f}_j \\ \boldsymbol{u}_j\end{bmatrix} \middle| \boldsymbol{0}, \begin{bmatrix}\boldsymbol{K}_{\boldsymbol{f}_j\boldsymbol{f}_j} & \boldsymbol{K}_{\boldsymbol{f}_j\boldsymbol{u}_j} \\ \boldsymbol{K}_{\boldsymbol{u}_j\boldsymbol{f}_j} & \boldsymbol{K}_{\boldsymbol{u}_j\boldsymbol{u}_j}\end{bmatrix}\right), \tag{3}$$

where $\boldsymbol{K}_{\boldsymbol{f}_j\boldsymbol{u}_j} \in \mathbb{R}^{N\times M}$ is the cross-covariance matrix formed by the evaluation of the kernel function $\kappa_j(\cdot,\cdot)$ between $\boldsymbol{X}$ and $\boldsymbol{Z}_j$. Likewise, $\boldsymbol{K}_{\boldsymbol{u}_j\boldsymbol{u}_j} \in \mathbb{R}^{M\times M}$ is the inducing points-based covariance matrix. Then, the distribution of $\boldsymbol{f}_j$ conditioned to the inducing points $\boldsymbol{u}_j$ can be written as:

$$p(\boldsymbol{f}_j|\boldsymbol{u}_j) = \mathcal{N}\left(\boldsymbol{f}_j|\boldsymbol{K}_{\boldsymbol{f}_j\boldsymbol{u}_j}\boldsymbol{K}_{\boldsymbol{u}_j\boldsymbol{u}_j}^{-1}\boldsymbol{u}_j, \boldsymbol{K}_{\boldsymbol{f}_j\boldsymbol{f}_j} - \cdots \right. \\ \left. \cdots - \boldsymbol{K}_{\boldsymbol{f}_j\boldsymbol{u}_j}\boldsymbol{K}_{\boldsymbol{u}_j\boldsymbol{u}_j}^{-1}\boldsymbol{K}_{\boldsymbol{u}_j\boldsymbol{f}_j}\right), \tag{4}$$

$$p(\boldsymbol{u}_j) = \mathcal{N}\left(\boldsymbol{u}_j|\boldsymbol{0}, \boldsymbol{K}_{\boldsymbol{u}_j\boldsymbol{u}_j}\right). \tag{5}$$

In most cases Eqs. (4) and (5) are non-conjugate to the likelihood, finding the posterior distribution $p(\boldsymbol{f}, \mathbf{u}|\mathbf{y})$ is not tractable analytically; therefore, we resort to a deterministic approximation of the posterior distribution using variational inference. Hence, the actual posterior can be approximated by a parametrized variational distribution $p(\hat{\boldsymbol{f}}, \boldsymbol{u}|\boldsymbol{y}) \approx q(\hat{\boldsymbol{f}}, \boldsymbol{u})$, as:

$$q(\boldsymbol{f}, \boldsymbol{u}) = p(\boldsymbol{f}|\boldsymbol{u})q(\boldsymbol{u}) = \prod_{j=1}^{J} p(\boldsymbol{f}_j|\boldsymbol{u}_j)q(\boldsymbol{u}_j), \tag{6}$$

where $\boldsymbol{u} = \left[\boldsymbol{u}_1^\top, \ldots, \boldsymbol{u}_J^\top\right]^\top \in \mathbb{R}^{MJ}$; moreover, $p(\boldsymbol{f}_j|\boldsymbol{u}_j)$ is defined in Eq. (4), and $q(\boldsymbol{u})$ is the posterior approximation over the inducing variables:

$$q(\boldsymbol{u}) = \prod_{j=1}^{J} q(\boldsymbol{u}_j) = \prod_{j=1}^{J} \mathcal{N}\left(\boldsymbol{u}_j|\boldsymbol{m}_j, \boldsymbol{V}_j\right). \tag{7}$$

The approximation for the posterior distribution comprises the estimation of the following variational parameters: the mean vectors $\boldsymbol{m}_j \in \mathbb{R}^M$ and the covariance matrices $\boldsymbol{V}_j \in \mathbb{R}^{M\times M}$. Such an assessment is carried out by maximizing an evidence lower bound–(ELBO). Thereby, assuming that the instances $\boldsymbol{x}_n$ are independently sampled, the ELBO can be derived as:

$$\mathcal{L} = \sum_{n=1}^{N} \mathbb{E}_{q(\boldsymbol{f}_1),\ldots,q(\boldsymbol{f}_J)} \left[\log p(y_n|\theta_{1,n},\ldots,\theta_{J,n}\right] - \cdots \\ \cdots - \sum_{j=1}^{J} \mathbb{D}_{KL}(q(\boldsymbol{u}_j)||p(\boldsymbol{u}_j)), \tag{8}$$

where $\mathbb{D}_{KL}(\cdot||\cdot)$ is the Kullback-Leibler divergence and $q(\boldsymbol{f}_j)$ is defined as follows:

$$q(\boldsymbol{f}_j) = \int p(\boldsymbol{f}_j|\boldsymbol{u}_j) q(\boldsymbol{u}_j) d\boldsymbol{u}_j. \tag{9}$$

B. Correlated chained Gaussian processes

From Section III-A, we note that the CGP model assumes independence between priors, thereby lacking a correlation structure between GPs. As mentioned before, we consider that the annotators are correlated. We will enable this aspect of the model by assuming dependencies among the latent parameters of the chained GP. In particular, we introduce the correlated chained GPs–(CCGP) to model correlations between the GP latent functions, which are supposed to be generated from a semi-parametric latent factor model–(SLFM) (29):

$$f_j(\boldsymbol{x}_n) = \sum_{q=1}^{Q} w_{j,q} \mu_q(\boldsymbol{x}_n), \tag{10}$$

where $f_j : \mathcal{X} \to \mathbb{R}$ is an LF, $\mu_q(\cdot) \sim \mathcal{GP}(0, k_q(\cdot,\cdot))$ with $k_q : \mathcal{X}\times\mathcal{X} \to \mathbb{R}$ being a kernel function, and $w_{j,q} \in \mathbb{R}$ is a combination coefficient ($Q \in \mathbb{N}$). Here, each LF is chained to the likelihood's parameters to extend the joint distribution in Eq. (2) as follows:

$$p(\boldsymbol{y}, \hat{\boldsymbol{f}}, \boldsymbol{u}|\boldsymbol{X}) = p(\boldsymbol{y}|\boldsymbol{\theta}) \prod_{j=1}^{J} p(\boldsymbol{f}_j|\boldsymbol{u}) p(\boldsymbol{u}), \tag{11}$$

where $\boldsymbol{\theta} = [\boldsymbol{\theta}_1,\ldots,\boldsymbol{\theta}_J]^\top \in \mathbb{R}^{NJ}$ holds the model's parameters and $\boldsymbol{\theta}_j = [\theta_j(\boldsymbol{x}_1),\ldots,\theta_j(\boldsymbol{x}_N)]^\top \in \mathbb{R}^N$ relates the j-th parameter with the input space. Our CCGP employs the inducing variables-based method for sparse approximations of GPs (40). For each $\mu_q(\cdot)$, we introduce a set of $M \leq N$ "pseudo variables" $\boldsymbol{u}_q = [\mu_q(\boldsymbol{z}_1^q),\ldots,\mu_q(\boldsymbol{z}_M^q)]^\top \in \mathbb{R}^M$ through evaluations of $\mu_q(\cdot)$ at unknown locations $\boldsymbol{Z}_q = [\boldsymbol{z}_1^q,\ldots,\boldsymbol{z}_M^q] \in \mathbb{R}^{M\times P}$. Note that $\boldsymbol{u} = \left[\boldsymbol{u}_1^\top,\ldots,\boldsymbol{u}_Q^\top\right]^\top \in \mathbb{R}^{QM}$, yielding:

$$p(\boldsymbol{f}_j|\boldsymbol{u}) = \mathcal{N}\left(\boldsymbol{f}_j|\boldsymbol{K}_{\boldsymbol{f}_j\boldsymbol{u}}\boldsymbol{K}_{\boldsymbol{u}\boldsymbol{u}}^{-1}\boldsymbol{u}, \boldsymbol{K}_{\boldsymbol{f}_j\boldsymbol{f}_j} - \cdots \right. \\ \left. \cdots - \boldsymbol{K}_{\boldsymbol{f}_j\boldsymbol{u}}\boldsymbol{K}_{\boldsymbol{u}\boldsymbol{u}}^{-1}\boldsymbol{K}_{\boldsymbol{u}\boldsymbol{f}_j}\right), \tag{12}$$

$$p(\boldsymbol{u}) = \mathcal{N}(\boldsymbol{u}|\boldsymbol{0}, \boldsymbol{K}_{\boldsymbol{u}\boldsymbol{u}}) = \prod_{q=1}^{Q} \mathcal{N}(\boldsymbol{u}_q|\boldsymbol{0}, \boldsymbol{K}_{\boldsymbol{u}_q\boldsymbol{u}_q}), \tag{13}$$

where $\boldsymbol{K}_{\boldsymbol{u}\boldsymbol{u}} \in \mathbb{R}^{QM\times QM}$ is a block-diagonal matrix with blocks $\boldsymbol{K}_{\boldsymbol{u}_q\boldsymbol{u}_q} \in \mathbb{R}^{M\times M}$, based on the kernel function $\kappa_q(\cdot,\cdot)$. The covariance matrix $\boldsymbol{K}_{\boldsymbol{f}_j\boldsymbol{f}_j} \in \mathbb{R}^{N\times N}$ holds elements $\sum_{q=1}^{Q} w_{j,q} w_{j,q} \kappa_q(\boldsymbol{x}_n, \boldsymbol{x}_{n'})$, with $\boldsymbol{x}_n, \boldsymbol{x}_{n'} \in \boldsymbol{X}$. Likewise, $\boldsymbol{K}_{\boldsymbol{f}_j\boldsymbol{u}} = [\boldsymbol{K}_{\boldsymbol{f}_j\boldsymbol{u}_1},\ldots,\boldsymbol{K}_{\boldsymbol{f}_j\boldsymbol{u}_Q}] \in \mathbb{R}^{N\times QM}$, where $\boldsymbol{K}_{\boldsymbol{f}_j\boldsymbol{u}_q} \in \mathbb{R}^{N\times M}$ gathers elements $w_{j,q}\kappa_q(\boldsymbol{x}_n, \boldsymbol{z}_m^q)$, $m \in \{1,\ldots,M\}$. Alike CGP, in most cases, the CCGP posterior distribution $p(\hat{\boldsymbol{f}}, \boldsymbol{u}|\boldsymbol{y})$ has not an analytical solution, so the actual posterior can be approximated by a parametrized variational distribution $p(\hat{\boldsymbol{f}}, \boldsymbol{u}|\boldsymbol{y}) \approx q(\hat{\boldsymbol{f}}, \boldsymbol{u})$, as:

$$q(\hat{\boldsymbol{f}}, \boldsymbol{u}) = p(\hat{\boldsymbol{f}}|\boldsymbol{u}) q(\boldsymbol{u}) = \prod_{j=1}^{J} p(\boldsymbol{f}_j|\boldsymbol{u}) \prod_{q=1}^{Q} q(\boldsymbol{u}_q), \tag{14}$$

where $p(\boldsymbol{f}_j|\boldsymbol{u})$ is given by Eq. (12), $q(\boldsymbol{u}_q) = \mathcal{N}(\boldsymbol{u}_q|\boldsymbol{m}_q, \boldsymbol{V}_q)$, and $q(\boldsymbol{u}) = \mathcal{N}(\boldsymbol{u}|\boldsymbol{m}, \boldsymbol{V})$. Also, $\boldsymbol{m}_q \in \mathbb{R}^M$, and $\boldsymbol{V}_q \in \mathbb{R}^{M\times M}$ are respectively the mean and covariance of variational distribution $q(\boldsymbol{u}_q)$; similarly, $\boldsymbol{m} = [\boldsymbol{m}_1^\top,\ldots,\boldsymbol{m}_Q^\top]^\top \in \mathbb{R}^{QM}$, and $\boldsymbol{V} \in \mathbb{R}^{QM\times QM}$ is a block-diagonal matrix with blocks given by the covariance matrices $\boldsymbol{V}_q$. We remark that the variational approximation given by Eq. (14) is not uncommon, and it has been used in several GPs models, including (27; 41). The approximation for the posterior distribution comprises the computation of the following variational parameters: the mean vectors $\{\boldsymbol{m}_q\}_{q=1}^Q$ and the covariance matrices $\{\boldsymbol{V}_q\}_{q=1}^Q$. Such an estimation is carried out by maximizing an evidence lower bound–(ELBO), which is given as:

$$\mathcal{L} = \sum_{n=1}^{N} \mathbb{E}_{q(\boldsymbol{f}_1),\ldots,q(\boldsymbol{f}_J)} \left[\log p(y_n|\theta_{1,n},\ldots,\theta_{J,n}\right] - \cdots \\ \cdots - \sum_{q=1}^{Q} \mathbb{D}_{KL}(q(\boldsymbol{u}_q)||p(\boldsymbol{u}_q)), \tag{15}$$

where $\theta_{j,n} = \theta_j(\boldsymbol{x}_n)$, with $j \in \{1,\ldots,J\}$, and $\mathbb{D}_{KL}(\cdot||\cdot)$ is the Kullback-Leibler divergence and $q(\boldsymbol{f}_j)$ is defined as follows:

$$q(\boldsymbol{f}_j) = \mathcal{N}(\boldsymbol{f}_j|\boldsymbol{K}_{\boldsymbol{f}_j\boldsymbol{u}}\boldsymbol{K}_{\boldsymbol{u}\boldsymbol{u}}^{-1}\boldsymbol{m}, \boldsymbol{K}_{\boldsymbol{f}_j\boldsymbol{f}_j} + \cdots \\ \cdots + \boldsymbol{K}_{\boldsymbol{f}_j\boldsymbol{u}}\boldsymbol{K}_{\boldsymbol{u}\boldsymbol{u}}^{-1}(\boldsymbol{V} - \boldsymbol{K}_{\boldsymbol{u}\boldsymbol{u}})\boldsymbol{K}_{\boldsymbol{u}\boldsymbol{u}}^{-1}\boldsymbol{K}_{\boldsymbol{u}\boldsymbol{f}_j}). \tag{16}$$

Yet, in presence of non-Gaussian likelihoods, the computation of the variational expectations–(VEs) in Eq. (15) cannot be solved analytically (27; 41). Hence, aiming to model different data types, i.e., classification and regression tasks, we need to find a generic alternative to solve the integrals related to these expectations. In that sense, we use the Gaussian-Hermite quadratures approach as in (40; 27). We remark such ELBO is used to infer the model's hyperparameters such as the inducing points, the kernel hyperparameters, and the combination factors $w_{j,q}$ Eq. (10). It is worth mentioning that the CCGPs objective functions exhibit an ELBO that allows Stochastic Variational Inference–(SVI) (42). Hence, the optimization is solved through a *mini-batch*-based approach from noisy estimates of the global objective gradient, which allows dealing with large scale datasets (40; 27; 41). Finally, we notice that the computational complexity for our CCGP is similar to the model in (41). Accordingly, it is dominated by the inversion of $\boldsymbol{K}_{\boldsymbol{u}\boldsymbol{u}}$ with $\mathcal{O}(QM^3)$ and products like $\boldsymbol{K}_{\hat{\boldsymbol{f}}\boldsymbol{u}}$ with $\mathcal{O}(JNQM^2)$.

C. Correlated chained GP for multiple annotators-CCGPMA

Let us consider that a predefined panel of $R \in \mathbb{N}$ annotators (with different and unknown levels of expertise) label a given dataset of N instances. It is common to find that the each annotator r only labels $|N_r| \leq N$ samples, being $|N_r|$ the cardinality of the set $N_r \subseteq \{1, \ldots, N\}$ that contains the indexes of samples labeled by the r-th annotator. Besides, we define the set $R_n \subseteq \{1, \ldots, R\}$ holding the indexes of annotators that labeled the n-th instance. The input-output set is coupled within a multiple annotators scenario as $\mathcal{D} = \{\boldsymbol{X}, \boldsymbol{Y} = \{y_n^r\}_{n \in N, r \in R_n}\}$, where $y_n^r \in \mathcal{Y}$ is the output given by labeler r to the sample n; accordingly, our main aims are: *i)* to code each labeler's performance as a function of the input space and taking into account inter-annotator dependencies, and *ii)* to predict the true output $y_* \in \mathcal{Y}$ of a new instance $\boldsymbol{x}_* \in \mathbb{R}^P$. We highlight that to achieve such objectives, no extra information about the annotators' behaviour is provided (e.g., extra labels or information about her/his experience).

1) Classification: To model categorical data from multiple annotators with K classes ($\mathcal{Y} = \{1, \ldots, K\}$) using our CCGPMA, we use the framework proposed in (32), which introduces a binary variable $\lambda_n^r \in \{0, 1\}$ representing the r-th labeler's reliability as a function of each sample $\boldsymbol{x}_n$. If $\lambda_n^r = 1$, the r-th annotator is supposed to provide the actual label, yielding to a categorical distribution. Conversely, $\lambda_n^r = 0$ indicates that the r-th annotator gives an incorrect output, which is modeled by a uniform distribution. Therefore, the likelihood function is given as:

$$p(\boldsymbol{Y}|\boldsymbol{\theta}) = \prod_{n=1}^{N} \prod_{r \in R_n} \left(\prod_{k=1}^{K} \zeta_{k,n}^{\delta(y_n^r, k)} \right)^{\lambda_n^r} \left(\frac{1}{K} \right)^{(1-\lambda_n^r)}, \quad (17)$$

where $\delta(y_n^r, k) = 1$, if $y_n^r = k$, otherwise $\delta(y_n^r, k) = 0$. Besides, $\zeta_{k,n} = p(y_n^r = k | \lambda_n^r = 1)$ is an estimation of the unknown ground truth. Accordingly, $J = K + R$ LFs are required within our CCGPMA approach, aiming to model the likelihood parameters $\boldsymbol{\theta}$. In particular, K LFs are used to model $\zeta_{k,n}$ based on a softmax function ι as:

$$\zeta_{k,n} = \iota(f_k(\boldsymbol{x}_n)) = \frac{\exp(f_k(\boldsymbol{x}_n))}{\sum_{j=1}^{K} \exp(f_j(\boldsymbol{x}_n))}. \quad (18)$$

Besides, R LFs are utilized to compute each λ_n^r from a step function; therefore, $\lambda_n^r = 1$ if $f_{l_r}(\boldsymbol{x}_n) \geq 0$, otherwise, $\lambda_n^r = 0$ ($r \in \{1, \ldots R\}$). $l_r = K + r \in \{K+1, \ldots J\}$ indexes the r-th annotator' LF. Of note, we approximate the step function through the well-known sigmoid function ς to avoid discontinuities and favor the CCGPMA implementation. Alike to CCGP, we use variational inference to approximate the posterior distribution of our CCGPMA. In consequence, the actual posterior $p(\hat{\boldsymbol{f}}, \boldsymbol{u}|\boldsymbol{Y})$ is approximated following Eq. (14). Besides, we can derive a CCGPMA ELBO, yielding:

$$\mathcal{L} = \sum_{n=1}^{N} \sum_{r \in R_n} \mathbb{E}_{q(\boldsymbol{f}_1), \ldots, q(\boldsymbol{f}_J)} \left[\log p(y_n^r | \theta_{1,n}, \ldots, \theta_{J,n}) \right] - \cdots$$
$$\cdots - \sum_{q=1}^{Q} \mathbb{D}_{KL}(q(\boldsymbol{u}_q) || p(\boldsymbol{u}_q)), \quad (19)$$

where for the classification case, we have

$$p(y_n^r | \theta_{1,n}, \ldots, \theta_{J,n}) = \left(\prod_{k=1}^{K} \zeta_{k,n}^{\delta(y_n^r, k)} \right)^{\lambda_n^r} \left(\frac{1}{K} \right)^{(1-\lambda_n^r)}. \quad (20)$$

Finally, given a new sample $\boldsymbol{x}_*$, we are interested in the mean and variance for predictive distributions related to the ground truth $\zeta_{k,*} = p(y_* = k | \boldsymbol{x}_*, \hat{\boldsymbol{f}}, \boldsymbol{u})$, and the labelers' reliabilities λ_*^r. Accordingly, for $\zeta_{k,*}$ we obtain

$$\mathbb{E}[\zeta_{k*}] \approx \int \iota(f_k(\boldsymbol{x}_*)) q(\boldsymbol{f}_*) d\boldsymbol{f}_*, \quad (21)$$

where $q(\boldsymbol{f}_*) = \int p(\boldsymbol{f}_* | \boldsymbol{u}) q(\boldsymbol{u}) d\boldsymbol{u}$. Similarly, for the predictive variance of $\zeta_{k,*}$, we use the expression $\mathrm{Var}[\zeta_{k,*}] = \mathbb{E}[\zeta_{k,*}^2] - \mathbb{E}[\zeta_{k,*}]^2$; hence, we need to compute $\mathbb{E}[\zeta_{k,*}^2]$ as

$$\mathbb{E}[\zeta_{k*}^2] \approx \int \iota(f_k(\boldsymbol{x}_*))^2 q(\boldsymbol{f}_*) d\boldsymbol{f}_*. \quad (22)$$

On the other hand, regarding the predictive mean and variance for λ_*^r, we have

$$\mathbb{E}[\lambda_*^r] = \int \varsigma(f_{l_r,*}) q(\boldsymbol{f}_*) d\boldsymbol{f}_*. \quad (23)$$

For the variance of λ_*^r, we use the expression $\mathrm{Var}[\lambda_*^r] = \mathbb{E}[(\lambda_*^r)^2] - \mathbb{E}[\lambda_*^r]^2$; hence, we need to compute

$$\mathbb{E}[(\lambda_*^r)^2] = \int \varsigma(f_{l_r,*})^2 q(\boldsymbol{f}_*) d\boldsymbol{f}_*. \quad (24)$$

In this case, integrals in Eqs. (21) to (24) have not closed solution; hence, we approximate them using the Gaussian-Hermite quadrature.

2) Regression: For real-valued outputs, e.g., $\mathcal{Y} \subset \mathbb{R}$, we follow the multi-annotator model used in (11; 14; 35; 15), where each output y_n^r is considered to be a corrupted version of the hidden ground truth y_n. Then:

$$p(\boldsymbol{Y}|\boldsymbol{\theta}) = \prod_{n=1}^{N} \prod_{r \in R_n} \mathcal{N}(y_n^r | y_n, v_n^r), \quad (25)$$

where $v_n^r \in \mathbb{R}^+$ is the r-th annotator error-variance for the instance n. In turn, to model this likelihood function with CCGPMA, it is necessary to chain each likelihood's paramater to a latent function f_j. Thus, we require $J = R + 1$ LFs; one to model the hidden ground truth, such that $y_n = f_1(\boldsymbol{x}_n)$, and R LFs to model each error-variance $v_n^r = \exp(f_{l_r}(\boldsymbol{x}_n))$, with $r \in \{1, \ldots R\}$, and $l_r = r + 1 \in \{2, \ldots J\}$. Note that we use an exponential function to map from f_{l_r} to v_n^r, aiming

to guarantee $v_n^r > 0$ ($f_{l_r} \in \mathbb{R}$). Similar to the classification problem, the ELBO in regression settings is given by Eq. (19), where $p(y_n^r|\theta_{1,n},\ldots,\theta_{J,n}) = \mathcal{N}(y_n^r|y_n, v_n^r)$.

Now, given a new sample $\boldsymbol{x}_*$, we are interested in the mean and variances for predictive distributions concerning the ground truth y_*, and the labelers' error-variances v_*^r. First, for y_* we have that since $\mathbf{y} = \boldsymbol{f}_1$, the posterior distribution for y_* corresponds to $q(f_{1*})$, yielding:

$$\mathbb{E}[y_*] = \mu_{1,*} \tag{26}$$

$$\mathrm{Var}[y_*] = s_{1,*}, \tag{27}$$

where $\mu_{1,*}$, and $s_{1,*}$ are respectively the mean and variance of $q(f_{1*})$. Then, for v_*^r, we note that due to $\mathbf{v}_r = \exp(\boldsymbol{f}_{l_r})$, the posterior distribution for v_*^r follows a log-normal distribution with parameters $\mu_{l_r,*}$ and $s_{l_r,*}$, which respectively correspond to the mean and variance of $q(f_{l_r,*})$. In this sense, the mean and variance of v_*^r are given as:

$$\mathbb{E}[v_*^r] = \exp\left(\mu_{l_r,*} + \frac{s_{l_r,*}}{2}\right). \tag{28}$$

$$\mathrm{Var}[v_*^r] = \exp\left(2\mu_{l_r,*} + s_{l_r,*}\right)\left(\exp(s_{l_r,*}) - 1\right). \tag{29}$$

IV. Experimental Set-up

In this section, we describe the experiments' configurations to validate our CCGPMA concerning multiple annotators classification and regression tasks.

A. Classification

1) Datasets and simulated/provided annotations: We test our approach using three types of datasets: *fully synthetic data*, *semi-synthetic data*, and *fully real datasets*.

First, we generate *fully synthetic data* as one-dimensional ($P = 1$) multi-class classification problem ($K = 3$). The input feature matrix $\boldsymbol{X}$ is built by randomly sampling $N = 100$ points from an uniform distribution within the interval $[0, 1]$. The true label for the n-th sample is generated by taking the $\arg\max_i\{t_{n,i} : i \in \{1, 2, 3\}\}$, where $t_{n,1} = \sin(2\pi x_n)$, $t_{n,2} = -\sin(2\pi x_n)$, and $t_{n,3} = -\sin(2\pi(x_n + 0.25)) + 0.5$. Besides, the test instances are obtained by extracting 200 equally spaced samples from the interval $[0, 1]$.

Second, to control the label generation, we build *semi-synthetic data* from seven datasets of the UCI repository[2] focused on binary and multi class-classification: Wisconsin Breast Cancer Database–(breast), BUPA liver disorders (bupa), Johns Hopkins University Ionosphere database (ionosphere), Pima Indians Diabetes Database–(pima), Tic-Tac-Toe Endgame database–(tic-tac-toe), Occupancy Detection Data Set–(Occupancy), Skin Segmentation Data Set–(Skin), Wine Data set–(Wine), and Image Segmentation Data Set (Segmentation). Also, we test the publicly available bearing data collected by the Case Western Reserve University–(Western). The aim is to build a system to diagnose an electric motor's status based on two accelerometers. The feature extraction was performed as in (43).

TABLE II
Tested datasets.

	Name	Number of features	Number of instances	Number of classes
fully synthetic	synthetic	1	100	3
semi-synthetic	Breast	9	683	2
	Bupa	6	345	2
	Ionosphere	34	351	2
	Pima	8	768	2
	Tic-tac-toe	9	958	2
	Occupancy	7	20560	2
	Skin	4	245057	2
	Western	7	3413	4
	Wine	13	178	3
	Segmentation	18	2310	7
fully real	Voice	13	218	2
	Music	124	1000	10

Third, we evaluate our proposal on two *fully real datasets*, where both the input features and the annotations are captured from real-world problems. Namely, we use a bio-signal database, where the goal is to build a system to evaluate the presence/absence of voice pathologies. In particular, a subset ($N = 218$) of the Massachusetts Eye and Ear Infirmary Disordered Voice Database from the Kay Elemetrics company is utilized, which comprises voice records from healthy and different voice issues. Each signal is parametrized by the Mel-frequency cepstral coefficients (MFCC) to obtain an input space with $P = 13$. A set of physicians assess the voice quality by following the GRBAS protocol that comprises the evaluation of five qualitative scales: Grade of dysphonia–(G), Roughness–(R), Breathiness–(B), Asthenia–(A), and Strain–(S). For each perceptual scale, the specialist assigns a tag ranging from 0 (healthy voice) to 3 (severe disease) (44). Accordingly, we face five multi-class classification problems (one per scale). We follow the procedure in (36) to rewrite five binary classification tasks preserving the available ground truth (13). Further, we use the music genre data[3], holding a collection of songs records labeled from one to ten depending on their music genre: classical, country, disco, hip-hop, jazz, rock, blues, reggae, pop, and metal. From this set, 700 samples were published randomly in the AMT platform to obtain labels from multiples sources (2946 annotations from 44 workers). Yet, we only consider the annotators who labeled at least 20% of the instances; thus, we use the information from $R = 7$ labelers. The feature extraction is performed by following the work by authors in (32), to obtain an input space with $P = 124$. Table II summarizes the tested datasets for the classification case.

Note that the *fully synthetic* and the *semi-synthetic* datasets do not hold real annotations. Therefore, it is necessary to simulate those labels as corrupted versions of the hidden ground truth. Here, the simulations are performed by assuming: i) dependencies among annotators, and ii) the labelers' performance is modeled as a function of the input features. In turn, an SLFM-based approach (termed SLFM-C) is used to build the labels, as follows:

– Define Q deterministic functions $\hat{\mu}_q : \mathcal{X} \to \mathbb{R}$, and their combination parameters $\hat{w}_{l_r,q} \in \mathbb{R}$, $\forall r \in R, n \in N$.

[2] http://archive.ics.uci.edu/ml

[3] http://fprodrigues.com/publications/learning-from-multiple-annotators-distinguishing-good-from-random-labelers/

TABLE III
A BRIEF OVERVIEW OF THE STATE-OF-THE-ART METHODS TESTED.

Algorithm	Description
GPC-GOLD	A GPC using the real labels (upper bound).
GPC-MV	A GPC using the MV of the labels as the ground truth.
MA-LFC-C (11)	A LRC with constant parameters across the input space.
MA-DGRL (32)	A multi-labeler approach that considers as latent variables the annotator performance.
MA-GPC (12)	A multi-labeler GPC, which is as an extension of MA-LFC.
MA-GPCV (7)	An extension of MA-GPC that includes variational inference and priors over the labelers' parameters.
MA-DL (18)	A Crowd Layer for DL, where the annotators' parameters are constant across the input space.
KAAR (36)	A kernel-based approach that employs a convex combination of classifiers and codes labelers dependencies.
CGPMA-C	A particular case of our CCGPMA for classification, where $Q = J$, and we fix $w_{j,q} = 1$, if $j = q$, otherwise $w_{j,q} = 0$.

– Compute $\hat{f}_{l_r,n} = \sum_{q=1}^{Q} \hat{w}_{l_r,q}\hat{\mu}_q(\hat{x}_n)$, where $\hat{x}_n \in \mathbb{R}$ is the n-th component of $\hat{\boldsymbol{x}} \in \mathbb{R}^N$, being $\hat{\boldsymbol{x}}$ the $1-$D representation of the input features in $\boldsymbol{X}$ by using the well-known t-distributed Stochastic Neighbor Embedding approach (45).
– Calculate $\hat{\lambda}_n^r = \varsigma(\hat{f}_{l_r,n})$, where $\varsigma(\cdot) \in [0,1]$ is the sigmoid function.
– Finally, find the r-th label as $y_n^r = \begin{cases} y_n, & \text{if } \lambda_n^r \geq 0.5 \\ \tilde{y}_n, & \text{if } \lambda_n^r < 0.5 \end{cases}$, where $\tilde{y}_n$ is a flipped version of the actual label y_n.

2) Method comparison and performance metrics: The classification performance is assessed as the Area Under the Curve–(AUC). Further, the AUC is extended for multi-class settings, as discussed by authors in (46). We use a cross-validation scheme with 15 repetitions where 70% of the samples are utilized for training and the remaining 30% for testing (except for the music dataset training and testing sets are clearly defined). Table III displays the employed methods of the state-of-the-art for comparison purposes. The abreviations are fixed as: Gaussian Processes classifier (GPC), logistic regression classifier (LRC), majority voting (MV), multiple annotators (MA), Modelling annotators expertise (MAE), Learning from crowds (LFC), Distinguishing good from random labelers (DGRL), kernel alignment-based annotator relevance analysis (KAAR).

B. Regression

1) Datasets and simulated/provided annotations: We test our approach using three types of datasets: fully synthetic data, semi-synthetic data, and fully real datasets. First, We generate *fully synthetic data* as an one-dimensional regression problem, where the ground truth for the n-th sample corresponds to $y_n = \sin(2\pi \boldsymbol{x}_n)\sin(6\pi \boldsymbol{x}_n)$, where the input matrix $\boldsymbol{X}$ is formed by randomly sampling 100 points within the range $[0, 1]$ from an uniform distribution. The test instances are obtained by extracting equally spaced samples from the interval $[0, 1]$. Second, to control the label generation (10), we build *semi-synthetic data* from six datasets related to regression tasks from the well-known UCI repository. We selected the following datasets: Auto MPG Data Set–(Auto), Bike Sharing Dataset Data Set–(Bike), Concrete Compressive Strength Data Set–(Concrete), The Boston Housing Dataset–(Housing),[4] Yacht Hydrodynamics Data Set–(Yacht), and Relative location of CT slices on axial axis Data Set–(CT). Third, we evaluate our proposal on one *fully real dataset*. In particular, we use the Music dataset introduced in Section IV-A1. Notice that the music dataset configures a 10-class classification problem; however, in this experiment, we are using our CCGPMA with a likelihood function designed for real-valued labels Eq. (25). Such practice is not uncommon in machine learning, and it is usually known as "Least-square classification" (39). Table IV summarizes the tested datasets for the regression case.

[4]See https://www.cs.toronto.edu/∼delve/data/boston/bostonDetail.html for housing

TABLE IV
DATASETS FOR REGRESSION.

	Name	Number of features	Number of instances
fully synthetic	synthetic	1	100
semi-synthetic	Auto	8	398
	Bike	13	17389
	Concrete	9	1030
	Housing	13	506
	Yacht	6	308
	CT	384	53500
fully real	Music	124	1000

TABLE V
A BRIEF OVERVIEW OF STATE-OF-THE-ART METHODS TESTED FOR REGRESSION TASKS. GPR: GAUSSIAN PROCESSES REGRESSION, LR: LOGISTIC REGRESSION, AV: AVERAGE, MA: MULTIPLE ANNOTATORS, DL: DEEP LEARNING, LFCR: LEARNING FROM CROWDS FOR REGRESSION.

Algorithm	Description
GPR-GOLD	A GPR using the real labels (upper bound).
GPR-Av	A GPR using the average of the labels as the ground truth.
MA-LFCR (11)	A LR model for MA where the labelers' parameters are supposed to be constant across the input space.
MA-GPR (12)	A multi-labeler GPR, which is as an extension of MA-LFCR.
MA-DL (18)	A Crowd Layer for DL, where the annotators' parameters are constant across the input space.
CGPMA-R	A particular case of our CCGPMA for regression, where $Q = J$, and $w_{j,q} = 1$ if $j = q$, otherwise $w_{j,q} = 0$.

As we pointed out previously, *fully synthetic* and *semi-synthetic* datasets do not hold real annotations. Thus, it is necessary to generate these labels synthetically as a version of the gold standard corrupted by Gaussian noise, i.e., $y_n^r = y_n + \epsilon_n^r$, where $\epsilon_n^r \sim \mathcal{N}(0, v_n^r)$, being v_n^r the r-th annotator error-variance for the sample n. Note that we are interested in modeling such an error-variance for the r-th annotator as a function of the input features, which is correlated with the other labelers' variances. In turn, an SLFM-based approach (termed SLFM-R) is used to build the labels, as follows:

- Define Q functions $\hat{\mu}_q : \mathcal{X} \to \mathbb{R}$, and the combination parameters $\hat{w}_{l_r,q} \in \mathbb{R}$, $\forall r, q$.
- Compute $\hat{f}_{l_r,n} = \sum_{q=1}^{Q} \hat{w}_{l_r,q}\hat{\mu}_q(\hat{x}_n)$, where $\hat{x}_n$ is the n-th component of $\hat{\boldsymbol{x}} \in \mathbb{R}$, which is an $1-$D representation of input features $\boldsymbol{X}$ by using the t-distributed Stochastic Neighbor Embedding approach (45).
- Finally, determine $\hat{v}_n^r = \exp(\hat{f}_{l_r,n})$.

2) Method comparison and performance metrics: The quality assessment is carried out by estimating the regression performance as the coefficient of determination–(R^2). A cross-validation scheme is employed with 15 repetitions where 70% of the samples are utilized for training and the remaining

30% for testing (except for *fully synthetic dataset*, since it clearly defines the training and testing sets). Table V displays the employed methods of the state-of-the-art for comparison purposes. From Table V, we highlight that for the model MA-DL, the authors provided three different annotators' codification: MA-DL-B, where the bias for the annotators is measured; MA-DL-S, where the labelers' scale is computed; and measured; MA-DL-B+S, which is a version with both (18).

C. CCGPMA training

Overall, the Radial basis function–(RBF) kernel is preferred in both classification and regression tasks because of its universal approximating ability and mathematical tractability. Hence, for all GP-based approaches, the kernel functions are fixed as:

$$\kappa(\boldsymbol{x}_n, \boldsymbol{x}_{n'}) = \phi_1 \exp\left(\frac{-\|\boldsymbol{x}_n - \boldsymbol{x}_{n'}\|_2^2}{2\phi_2^2}\right), \tag{30}$$

where $\|\cdot\|^2$ stands for the L2 norm, $n, n' \in \{1, 2, \ldots, N\}$, and $\phi_1, \phi_2 \in \mathbb{R}^+$ are the kernel hyper-parameters. For concrete testing, we fix $\phi_1 = 1$, while ϕ_2 is estimated by optimizing the corresponding ELBO (as exposed in Eq. (19)). Moreover, for CGPMA, since each LF $f_j(\cdot)$ is linked to $u_q(\cdot)$, we fix $Q = R \cdot K$, and $Q = R+1$ for classification and regression respectively. On the other hand, for CCGPMA, each $f_j(\cdot)$ is built as a convex combination of $\mu_q(\cdot)$ (see Eq. (10)); therefore, there is no restriction concerning Q. However, to make a fair comparison with CGPMA, we also fix $Q = R + K$ (classification), and $Q = R + 1$ (regression) in CCGPMA. For the *fully synthetic datasets*, we use $M = 10$ inducing points per latent function, and for the remaining experiments, we test with $M = 40$, and $M = 80$. For all the experiments, we use the ADADELTA included in the climin library with a mini-batch size of 100 samples to perform SVI. However, for small datasets ($N \leqslant 500$), we employ mini-batches with a size equal to the number of samples in the training set. Finally, for all experiments related to our CCGPMA, the variational parameters' initialization is carried out as follows: the variational mean is set $\boldsymbol{m}_q = \mathbf{0}, \forall q \in \{1, \ldots, Q\}$, where $\mathbf{0} \in \mathbb{R}^M$ is an all-zeros vector; the variational covariances $\boldsymbol{V}_q = \boldsymbol{I}, \forall q \in \{1, \ldots, Q\}$ are fixed as the identity matrix $\boldsymbol{I} \in \mathbb{R}^{M \times M}$. The CCGPMA's Python code is publicly available.[5]

V. Results and Discussion

A. Classification

1) Fully synthetic data results.: We first perform a controlled experiment to test the CCGPMA capability when dealing with binary and multi-class classification. We use the *fully synthetic* dataset described in Section IV-A1. Besides, five labelers ($R = 5$) are simulated with different levels of expertise. To simulate the error-variances, we define $Q = 3$ $\hat{\mu}_q(\cdot)$ functions, yielding:

$$\hat{\mu}_1(x) = 4.5\cos(2\pi x + 1.5\pi) - 3\sin(4.3\pi x + 0.3\pi), \tag{31}$$

$$\hat{\mu}_2(x) = 4.5\cos(1.5\pi x + 0.5\pi) + 5\sin(3\pi x + 1.5\pi), \tag{32}$$

$$\hat{\mu}_3(x) = 1, \tag{33}$$

where $x \in [0, 1]$. Besides, the combination weights are gathered within the following combination matrix $\hat{\boldsymbol{W}} \in \mathbb{R}^{Q \times R}$:

$$\hat{\boldsymbol{W}} = \begin{bmatrix} 0.4 & 0.7 & -0.5 & 0.0 & -0.7 \\ 0.4 & -1.0 & -0.1 & -0.8 & 1.0 \\ 3.1 & -1.8 & -0.6 & -1.2 & 1.0 \end{bmatrix}, \tag{34}$$

holding elements $\hat{w}_{l_r,q}$. For visual inspection purposes, Fig. 1 shows the predictive label's probability–(PLP), $p(y_* = k|\boldsymbol{x}_*)$, and the AUC for all studied approaches regarding the *fully synthetic* data. Notice that for methods MA-GPC, MA-GPCV, and KAAR, we use the *one-vs-all* scheme to face this experiment (such methods were defined only for binary classification settings). Accordingly, for those models, the PLP corresponds to scores rather than probabilities. Besides, regarding the PLP of our CGPMA and CCGPMA, we provide the mean and variance for the predictive distribution $\zeta_{k,*} = p(y_* = k|\boldsymbol{x}_*, \hat{\boldsymbol{f}}, \boldsymbol{u})$, which are computed based on Eqs. (21) and (22). As seen in Fig. 1, KAAR, MA-GPC, and MA-GPCV presents a different shape than the ground truth; moreover, KAAR and MA-GPCV exhibit the worst AUC, even worse than the intuitive lower bound GPC-MV. We explain such conduct in the sense that these approaches are designed to deal with binary labels (36; 12; 10). To face such a problem, we use the *one-vs-all* scheme; still, it can lead to ambiguously classified regions (47). We note an akin predictive AUC concerning MA-DL methods and the linear approaches MA-LFC-C and MA-DGRL. Nonetheless, the linear techniques exhibit a PLP less similar to the Ground truth, which is due to MA-LFC-C and MA-DGRL only can deal with linearly separable data. Further, we analyze the results of our CGPMA-C and its particular enhancement CCGPMA-C. We remark that our methods' predictive AUC is pretty close to deep learning and linear models. Unlike them, our CGPMA-C and CCGPMA-C show the most accurate PLP compared with the absolute gold standard. CCGPMA-C behaves quite similarly to GPC-GOLD, which is the theoretical upper bound. Finally, from the GPC-MV, we do not identify notable differences with the rest of the approaches (excluding KAAR and MA-GPCV).

From the above, we recognize that analyzing both the predictive AUC and the PLP, our CCGPMA-C exhibits the best performance obtaining similar results compared with the intuitive upper bound (GPC-GOLD). Accordingly, CCGPMA-C proffers a more suitable representation of the labelers' behavior than its competitors. Indeed, CCGPMA-C codes both the annotators' dependencies and the relationship between the input features and the annotators' performance. To empirically support the above statement, Fig. 2 shows the estimated per-annotator reliability, where we only take into account models that include such types of parameters (MA-DGRL, CGPMA-C, and CCGPMA-C). As seen, MA-DGRL (see column 2 in Fig. 2) does not offer a proper representation of the annotators' behavior. CGPMA-C and CCGPMA-C (columns 3 and 4 in Fig. 2) outperforms MA-DGRL, which is a direct repercussion

[5]https://github.com/juliangilg/CCGPMA

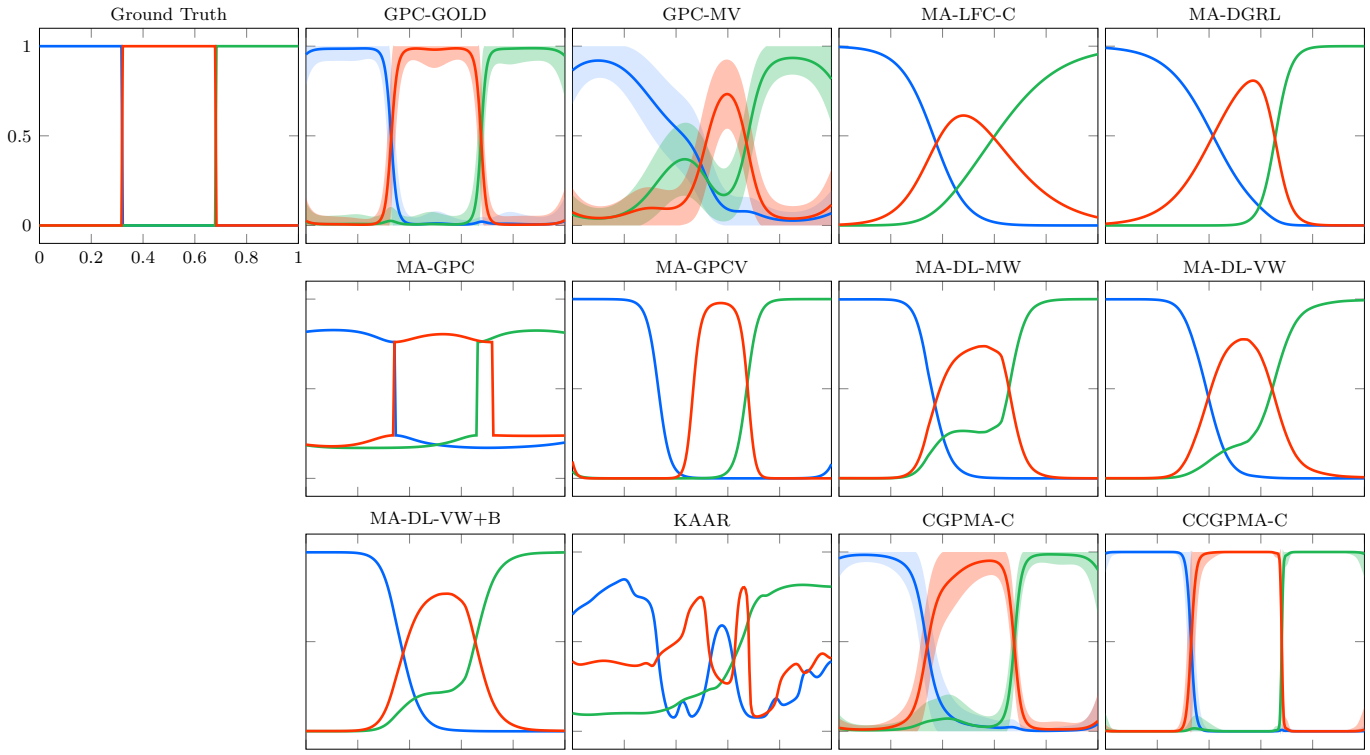


Fig. 1. Fully synthetic dataset results. The PLP is shown, comparing the prediction of our CCGPMA-C($AUC = 1$) and CCGPMA-C($AUC = 0.9999$) against: the theoretical upper bound GPC-GOLD($AUC = 1.0$), the lower bound GPC-MV($AUC = 0.9809$), and the state-of-the-art approaches MA-LFC-C($AUC = 0.9993$), MA-DGRL($AUC = 0.9999$), MA-GPC($AUC = 0.9977$), MA-GPCV($AUC = 0.9515$), MA-DL-MW($AUC = 0.9989$), MA-DL-VW($AUC = 0.9972$), MA-DL-VW+B($AUC = 0.9994$), KAAR(0.9099). Note that the shaded region in GPC-MV, CGPMA-C, and CCGPMA-C indicates the area enclosed by the mean $\pm$ two standard deviations. There is no shaded region for approaches lacking prediction uncertainty.

TABLE VI
AUC(%) CLASSIFICATION RESULTS FOR THE SEMI SYNTHETIC DATASETS. BOLD: THE HIGHEST AUC EXCLUDING THE UPPER BOUND (GPC-GOLD).

Method	Breast	Bupa	Ionosphere	Pima	TicTacToe	Occupancy	Skin	Western	Wine	Segmentation	Average
GPC-GOLD($M = 40$)	99.07 ± 0.45	69.75 ± 4.66	94.90 ± 2.35	83.78 ± 3.02	84.29 ± 3.34	99.56 ± 0.06	99.97 ± 0.01	91.85 ± 0.61	99.87 ± 0.15	95.96 ± 1.96	91.90
GPC-GOLD($M = 80$)	99.03 ± 0.46	69.97 ± 4.83	95.13 ± 2.25	83.74 ± 2.97	84.91 ± 3.23	99.56 ± 0.06	99.97 ± 0.01	92.50 ± 0.57	99.88 ± 0.16	97.81 ± 0.41	92.25
GPC-MV($M = 40$)	98.97 ± 0.45	53.66 ± 5.16	75.66 ± 5.72	53.99 ± 7.60	66.20 ± 3.57	75.85 ± 19.16	84.58 ± 0.90	86.58 ± 3.31	81.79 ± 2.12	95.62 ± 2.28	77.29
GPC-MV($M = 80$)	98.92 ± 0.48	56.98 ± 5.29	77.79 ± 5.50	53.02 ± 6.74	67.44 ± 3.57	63.12 ± 19.68	84.20 ± 0.80	84.46 ± 0.89	83.23 ± 4.87	97.49 ± 0.47	76.66
MA-LFC-C	87.89 ± 5.10	45.93 ± 14.44	73.58 ± 9.01	81.19 ± 3.13	60.04 ± 2.61	89.42 ± 0.79	$\mathbf{94.40 \pm 0.08}$	84.00 ± 2.11	96.92 ± 3.57	98.92 ± 0.31	81.23
MA-DGRL	97.57 ± 1.89	57.24 ± 3.36	64.53 ± 7.21	81.38 ± 2.90	61.29 ± 2.30	49.71 ± 1.05	93.79 ± 1.07	81.43 ± 1.50	97.95 ± 2.21	98.97 ± 0.38	78.39
MA-GPC	98.11 ± 1.16	54.46 ± 5.78	66.31 ± 14.74	53.25 ± 17.80	60.79 ± 9.95	92.57 ± 7.96	80.89 ± 0.60	86.71 ± 1.14	94.17 ± 2.62	97.34 ± 0.35	78.46
MA-GPCV	82.70 ± 5.47	55.67 ± 6.83	62.38 ± 8.71	62.17 ± 5.90	61.04 ± 10.03	60.22 ± 2.66	76.29 ± 3.74	84.51 ± 1.47	97.35 ± 1.72	99.24 ± 0.27	74.16
MA-DL-MW	94.70 ± 1.73	52.37 ± 5.68	75.35 ± 5.43	61.78 ± 2.67	68.27 ± 2.96	64.09 ± 2.26	86.36 ± 0.57	90.92 ± 0.56	97.28 ± 1.09	99.50 ± 0.17	79.06
MA-DL-VW	95.26 ± 2.45	53.27 ± 6.18	69.87 ± 4.97	60.63 ± 3.36	67.71 ± 2.67	68.40 ± 3.45	86.56 ± 0.68	91.73 ± 0.67	98.07 ± 1.52	$\mathbf{99.72 \pm 0.11}$	79.12
MA-DL-VW+B	94.65 ± 2.42	52.81 ± 6.31	71.96 ± 4.53	61.23 ± 3.78	67.80 ± 3.42	67.82 ± 3.86	86.68 ± 0.67	91.64 ± 0.85	98.17 ± 1.55	$\mathbf{99.72 \pm 0.09}$	79.25
KAAR	80.58 ± 2.74	59.20 ± 6.63	70.46 ± 7.39	58.02 ± 4.06	63.81 ± 5.45	69.16 ± 2.06	51.58 ± 4.74	85.88 ± 1.20	99.43 ± 1.05	92.17 ± 1.90	73.03
CGPMA-C($M = 40$)	99.20 ± 0.38	57.13 ± 4.68	83.56 ± 10.02	82.01 ± 3.14	70.56 ± 3.04	82.20 ± 2.73	92.62 ± 1.20	91.78 ± 0.66	99.82 ± 0.18	96.79 ± 0.65	85.56
CGPMA-C($M = 80$)	99.14 ± 0.38	56.96 ± 4.74	86.15 ± 6.96	$\mathbf{82.04 \pm 3.18}$	70.48 ± 3.12	$\mathbf{99.08 \pm 0.26}$	90.46 ± 1.64	91.85 ± 0.57	$\mathbf{99.84 \pm 0.12}$	94.06 ± 0.61	87.01
CCGPMA-C($M = 40$)	$\mathbf{99.38 \pm 0.27}$	$\mathbf{60.22 \pm 5.06}$	87.84 ± 6.72	78.10 ± 6.22	$\mathbf{74.95 \pm 5.39}$	91.98 ± 2.00	85.70 ± 2.66	93.09 ± 0.51	99.44 ± 0.33	97.67 ± 0.53	86.84
CCGPMA-C($M = 80$)	99.33 ± 0.30	59.19 ± 5.65	$\mathbf{90.55 \pm 6.29}$	80.45 ± 5.10	73.12 ± 3.23	97.75 ± 2.00	89.42 ± 2.20	$\mathbf{93.15 \pm 0.50}$	99.43 ± 0.33	97.58 ± 0.43	**88.00**

of modeling the labelers' parameters as functions of the input features. We observe that CCGPMA-C exhibits the best performance in terms of accuracy; such an outcome is due to this method improves the quality of the annotators' model by considering correlations among their decisions (26; 36)).

2) Semi-synthetic data results.: It is worth mentioning that the Semi-synthetic experiments are a common practice in the *learning from crowds* area (10; 36; 7), where the input features comes from real-world datasets whilst the labels from multiple annotators are simulated following the *fully-synthetic data* set-up (see Eqs. (31) to (34)). Table VI shows the results concerning this second experiment. On average, our CCGPMA-C accomplishes the best predictive AUC; moreover, we note that CGPMA-C reaches the second-best performance. Furthermore, the GPs-based competitors achieve competitive results (GPC-MV, MA-GPC, MA-GPCV, and KAAR). On the other hand, the GPC-MV method obtains a significantly lower performance than our CCGPMA-C, which is explained because GPC-MV is the most naive approach since it considers that the whole annotators exhibit the same performance. Conversely, analyzing the results from MA-GPC, MA-GPCV, and KAAR, we note that they perform worse than GPC-MV. We explain such an outcome in two ways. First, these approaches do not model the relationship between the input features and the annotators' performance. Second, as exposed in a previous experiment MA-GPC, MA-GPCV, and KAAR use a *one-*

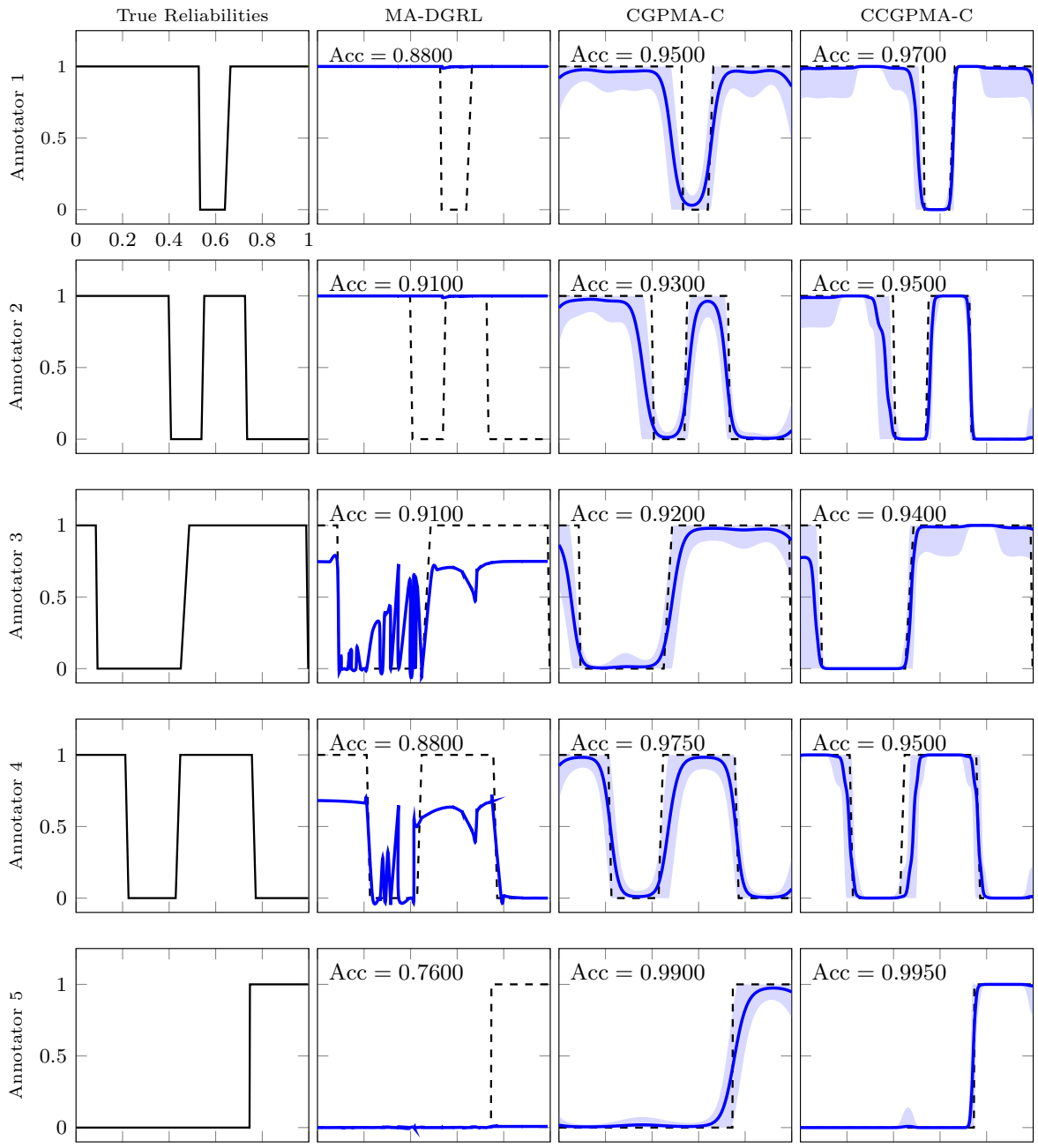


Fig. 2. Fully synthetic data reliability results. From top to bottom, the first column exposes the true reliabilities (λ_r). The subsequent columns present the estimation of the reliabilities performed by state-of-the-art models, where the correct values are provided in dashed lines. The shaded region in CGPMA-C and CCGPMA-C indicates the area enclosed by the mean $\pm$ two standard deviations. Also, the accuracy (Acc) is provided.

vs-all to deal with multi-class problems, which can lead to ambiguously classified regions (47). The latter can be confirmed in the results for the multi-class dataset "Western" ($K = 4$), where the predictive AUC for such approaches are the lowest. Then, analyzing the results from the DL-based strategies, we note a slightly better performance compared with the GPs-based methods (excluding CGPMA-C and CCGPMA-C). However, the DL-based performs considerably worse than our proposal because the CrowdLayer provides straightforward codification of the labelers' performance to guarantee a low computational cost (37). Finally, from the linear models, we first analyze the outstanding performance from MA-DGRL, which defeats all its non-linear competitors. In particular, the simulated labels (see Section IV-A1) follows the MA-DGRL model, favoring its performance. Though MA-LFC-C achieves competitive performance compared to the DL-based methods, it is considerably lower than our proposal. In fact, the MA-LFC-C formulation assumes that the annotators' behavior is homogeneous across the input space, which does not correspond to the labels simulation procedure.

3) Fully real data results.: We test the *fully real datasets*, which configure the most challenging scenario. The input

features and the labels from multiple experts come from real-world applications. Table VII outlines the achieved AUC. First, we observe that for the voice data, G and R scales exhibit a similar AUC for all considered approaches; in fact, GPC-MV obtains a result comparable with the upper bound GPC-GOLD. The latter can be explained in the sense that the annotators exhibit a suitable performance for these scales, i.e., the provided labels are similar to the ground truth. On the other hand, a reduction in the predictive AUC is observed for scale B, which is a consequence of diminishing the labelers' performance compared with scales G and R, as demonstrated in (13). Our approaches exhibit the best generalization performances for the three scales in the voice dataset. Remarkably, CGPMA-C and CCGPMA-C do not suffer significant changes in the scale B, which is an outstanding outcome because it reflects that our method offers a better representation of the labelers' behavior against low-quality annotations. Finally, we review the AUC for the Music dataset. Achieved results show a low performance for the MA-GPC, even lower than their intuitive lower bound (GPC-MV). Notably, our CCGPMA-C reaches the best predictive AUC, being comparable with the intuitive upper bound.

TABLE VII
AUC CLASSIFICATION RESULTS FOR THE FULLY REAL DATASETS. BOLD: THE HIGHEST PERFORMANCE EXCLUDING THE GPC-GOLD BOUND.

Method	Voice			Music	Average
	G	R	B		
GPC-GOLD($M = 40$)	0.9481	0.9481	0.9481	0.9358	0.9450
GPC-GOLD($M = 80$)	0.9484	0.9484	0.9484	0.9178	0.9407
GPC-MV($M = 40$)	0.8942	0.9373	0.8001	0.8871	0.8797
GPC-MV($M = 80$)	0.9301	0.9377	0.7962	0.8897	0.8884
MA-LFC-C	0.9122	0.9130	0.8406	0.8599	0.8814
MA-DGRL	0.9127	0.9164	0.8259	0.8832	0.8845
MA-GPC	0.8660	0.8597	0.4489	0.8253	0.7500
MA-GPCV	0.9283	0.9208	0.8835	0.8677	0.9001
MA-DL-MW	0.8957	0.8966	0.8123	0.8567	0.8653
MA-DL-VW	0.8942	0.8929	0.8092	0.9167	0.8782
MA-DL-VW+B	0.9030	0.8937	0.8218	0.8573	0.8689
KAAR	0.9109	0.9351	0.8969	0.8896	0.9081
CGPMA-C($M = 40$)	**0.9324**	0.9406	0.8696	0.9025	0.9113
CGPMA-C($M = 80$)	**0.9324**	0.9417	0.8708	0.8987	0.9109
CCGPMA-C($M = 40$)	0.9318	**0.9422**	**0.9002**	0.9446	**0.9297**
CCGPMA-C($M = 80$)	0.9243	0.9383	0.8907	**0.9456**	0.9247

B. Regression

1) Fully synthetic data results : We perform a controlled experiment aiming to verify the capability of our CGPMA and CCGPMA to estimate the performance of inconsistent annotators as a function of the input space and taking into account their dependencies. For this first experiment, we use the *fully synthetic* dataset described in Section IV-B1. We simulate five labelers ($R = 5$) with different levels of expertise. To simulate the error-variances, we define $Q = 3$ functions $\hat{\mu}_q(\cdot)$, which are given as

$$\hat{\mu}_1(x) = 4.5\cos(2\pi x + 1.5\pi) - 3\sin(4.3\pi x + 0.3\pi) + \cdots + 4\cos(7\pi x + 2.4\pi), \quad (35)$$

$$\hat{\mu}_2(x) = 4.5\cos(1.5\pi x + 0.5\pi) + 5\sin(3\pi x + 1.5\pi) - \cdots - 4.5\cos(8\pi x + 0.25\pi), \quad (36)$$

$$\hat{\mu}_3(x) = 1, \quad (37)$$

where $x \in [0, 1]$. Besides, we define the following combination matrix $\hat{\boldsymbol{W}} \in \mathbb{R}^{Q\times R}$, where

$$\hat{\boldsymbol{W}} = \begin{bmatrix} -0.10 & 0.01 & -0.05 & 0.01 & -0.01 \\ 0.10 & -0.01 & 0.01 & -0.05 & 0.05 \\ -2.3 & -1.77 & 0.54 & 0.9 & 1.42 \end{bmatrix}, \quad (38)$$

holding elements $w_{l_r,q}$.

Fig. 3 shows the predictive performance of all methods in this first experiment. The results show two clear groups: those based on GPs (GPR-Av, MA-GPR, CGPMA-R, and CCGPMA-R), which expose the best performance in terms of the R^2 score, and those based on other types of approaches (MA-LFCR, and MA-DL), whose performance is not satisfactory. The behavior of MA-LFCR is low since it only can deal with linear problems. Besides, concerning MA-DL and its three variations (S, B, and S+B), we note that this approach can deal with non-linear dynamics. However, MA-DL reaches a significantly low performance (even lower than the most naive approach, GPR-Av). To explain such an outcome, we remark that MA-DL comprises the introduction of an additional layer, the "CrowdLayer", which allows the training of neural networks directly from the noisy labels of multiple annotators (18). Yet, such a CrowdLayer provides a very simple codification of the annotators' performance to guarantee a low computational cost (37); therefore, MA-DL does not provide a proper codification of the annotators' behavior. On the other hand, among the GP-based methods, the proposed CCGPMA-R achieves the best performance in terms of R^2, followed closely by CGPMA-R and MA-GPR.

Besides, concerning the high performance of our CCGPMA-R (the best in terms of R^2 score), we hypothesize that such an outcome is a consequence of our method offers a better representation of the labelers' behavior when compared with its competitors. To empirically support the above hypothesis, Fig. 4 shows the estimated error-variances for this first experiment; here, we only take into account the models that include these parameters in their formulations. As seen in Fig. 4, MA-LFCR and MA-GPR offer the worst representation for the annotator's performance, which is due to such methods do not take into account the relationship between the annotators and the input space. Conversely, CGPMA-R and CCGPMA-R outperform the models named previously. This outcome is a consequence that such two approaches compute the error-variance as a function of the input features, allowing for a better codification of the labelers' behavior. Besides, by making a visual inspection and analyzing the R^2 scores, CCGPMA-R performs better than CGPMA-R because the former codes properly the annotators' interdependencies (26). Finally, we remark that although our CCGPMA-R achieves the best representation of the annotators' performance, Annotator 4 exhibits a lower performance in terms of R^2 score compared with the other labelers. Such an outcome is caused by the quasi-periodic behavior in the error-variances for those labelers, which cannot be captured because we are using an RBF-based kernel.

2) Results over semi-synthetic data: Table VIII shows the results of the *semi synthetic datasets*. On average, our CCGPMA-R exhibits the best generalization performance in

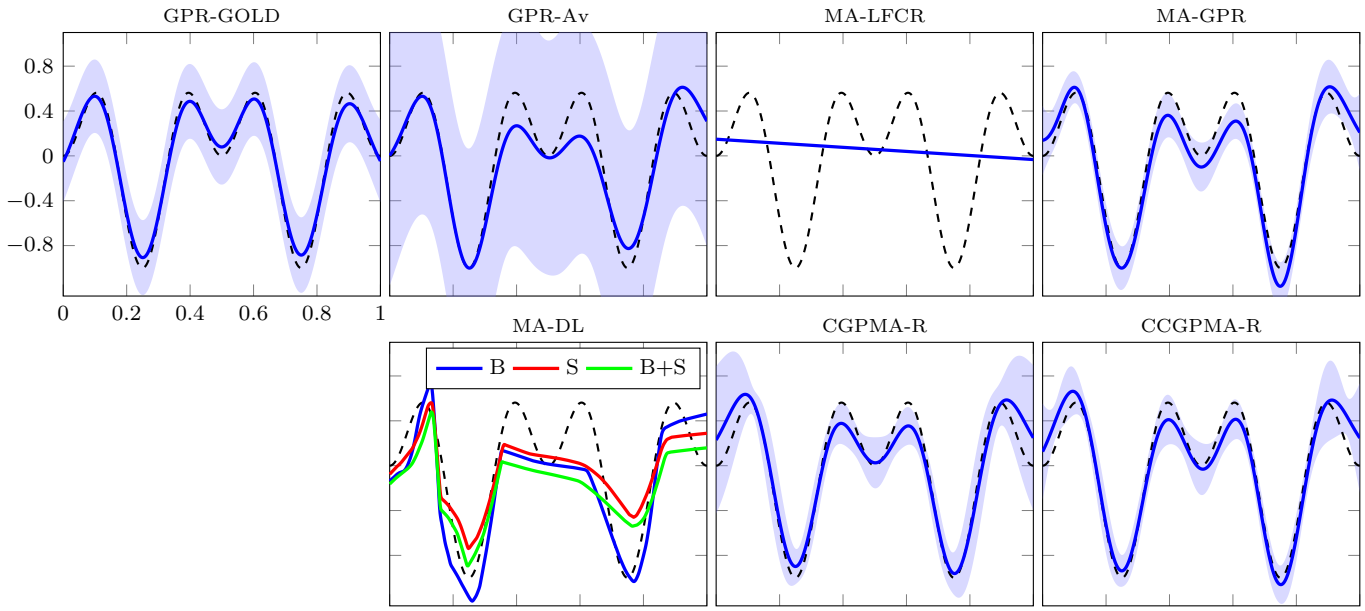


Fig. 3. Fully synthetic dataset results. We compare the prediction of our CCGPMA-R($R^2 = 0.9438$), and CGPMA-R($R^2 = 0.9280$) with the theoretical upper bound GPR-GOLD($R^2 = 0.9843$) and lower bound GPR-Av($R^2 = 0.8718$), and state-of-the-art approaches, MA-LFCR($R^2 = -0.0245$), MA-GPR($R^2 = 0.9208$), MA-DL-B($R^2 = 0.7020$), MA-DL-S($R^2 = 0.6559$), MA-DL-B+S($R^2 = 0.5997$). Note that we provided the Gold Standard in dashed lines. The shaded region in GPR-Av, MA-GPR, CGPMA-R, and CCGPMA-R indicates the area enclosed by the mean plus or minus two standard deviations. We remark that there is no shaded region for MA-LFCR, and DLMA since they do not provide information about the prediction uncertainty.

terms of the R^2 score. On the other hand, regarding its GPs-based competitors (GPR-Av, MA-GPR, and CGPMA-R), we first note that the performance of CGPMA-R exhibits a similar (but lower) performance than CCGPMA-R. The above is a consequence of that conversely to CGPMA-R, our CCGPMA-R models the annotators' interdependencies. Secondly, the intuitive lower bound GPR-Av exhibits a significantly worse prediction than our approaches. We remark on MA-GPR's behavior, which is lowest compared with its GPs-based competitors, even far worse than the supposed lower bound GPR-Av. The key to this abnormal outcome lies in the formulation of this approach; MA-GPR models the annotators' behavior by assuming that their performance does not depend on the input features and considering that the labelers make their decisions independently, which does fit the process that we use to simulate the labels.

Next, we analyze the results concerning the linear model MA-LFR; attained to the results, we note that this approach's prediction capacity is far lower than ours. The above outcome suggests that there may exist a non-linear structure in most databases. However, we highlight a particular result for the dataset CT, where MA-LFCR exhibits the best performance defeating all its competitors based on non-linear models. From the above, we intuit that the CT dataset may have a linear structure. To confirm this supposition, we perform an additional experiment over CT by training a regression scheme based on LR with the actual labels (we follow the same scheme as for GPR-GOLD). We obtain an R^2 score equal to 0.8541 (on average), which is close to GPR-GOLD results. Thus, we can elucidate that there exists a linear structure in the dataset CT. Finally, we analyze the results for the DL-based models. Similar to the experiments over *fully synthetic datasets*, we note a considerable low prediction capacity; in fact, they are even defeated by the linear model MA-LFR. Again, we attribute

TABLE VIII
REGRESSION RESULTS IN TERMS OF R^2 SCORE OVER *semi synthetic datasets*. BOLD: THE HIGHEST R^2 EXCLUDING THE UPPER BOUND GPR-GOLD.

Method	Auto	Bike	Concrete	Housing	Yacht	CT	Average
GPR-GOLD($M = 40$)	0.8604 ± 0.0271	0.5529 ± 0.0065	0.8037 ± 0.0254	0.8235 ± 0.0419	0.8354 ± 0.0412	0.8569 ± 0.0055	0.7888
GPR-GOLD($M = 80$)	0.8612 ± 0.0279	0.5603 ± 0.0063	0.8271 ± 0.0230	0.8275 ± 0.0399	0.8240 ± 0.0339	0.8648 ± 0.0047	0.7942
GPR-Av($M = 40$)	0.8425 ± 0.0286	0.5280 ± 0.0100	0.7589 ± 0.0279	0.7834 ± 0.0463	0.7588 ± 0.0498	0.8070 ± 0.0130	0.7464
GPR-Av($M = 80$)	0.8406 ± 0.0304	0.5397 ± 0.0085	0.7765 ± 0.0274	0.7903 ± 0.0451	0.7676 ± 0.0535	0.8167 ± 0.0089	0.7552
MA-LFCR	0.7973 ± 0.0218	0.3385 ± 0.0051	0.6064 ± 0.0384	0.7122 ± 0.0509	0.6403 ± 0.0186	0.8400 ± 0.0014	0.6558
MA-GPR	0.8456 ± 0.0281	0.4448 ± 0.0187	0.7769 ± 0.0367	0.7685 ± 0.0632	0.7842 ± 0.1027	0.0105 ± 0.0045	0.6051
MA-DL-B	0.7766 ± 0.0253	$\mathbf{0.5854 \pm 0.0107}$	0.2319 ± 0.0328	0.5317 ± 0.1005	0.2089 ± 0.0783	0.6903 ± 0.2689	0.5041
MA-DL-S	0.7761 ± 0.0279	0.5828 ± 0.0149	0.2363 ± 0.0252	0.5352 ± 0.0948	0.1822 ± 0.0985	$\mathbf{0.8418 \pm 0.2288}$	0.5257
MA-DL-B+S	0.7717 ± 0.0239	0.5816 ± 0.0181	0.2369 ± 0.0322	0.5330 ± 0.0850	0.1974 ± 0.0895	0.5517 ± 0.2316	0.4787
CGPMA-R($M = 40$)	0.8476 ± 0.0229	0.5464 ± 0.0069	0.8169 ± 0.0231	0.7244 ± 0.2973	0.8049 ± 0.0482	0.8236 ± 0.0132	0.7606
CGPMA-R($M = 80$)	0.8342 ± 0.0217	0.5560 ± 0.0074	0.8190 ± 0.0254	0.7259 ± 0.3018	0.7928 ± 0.0884	0.8371 ± 0.0104	0.7608
CCGPMA-R($M = 40$)	$\mathbf{0.8558 \pm 0.0248}$	0.5284 ± 0.0117	0.7976 ± 0.0270	0.8169 ± 0.0468	0.8409 ± 0.0548	0.8219 ± 0.0062	0.7769
CCGPMA-R($M = 80$)	0.8534 ± 0.0243	0.5467 ± 0.0069	$\mathbf{0.8220 \pm 0.0259}$	$\mathbf{0.8215 \pm 0.0466}$	$\mathbf{0.8691 \pm 0.0473}$	0.8252 ± 0.0083	**0.7897**

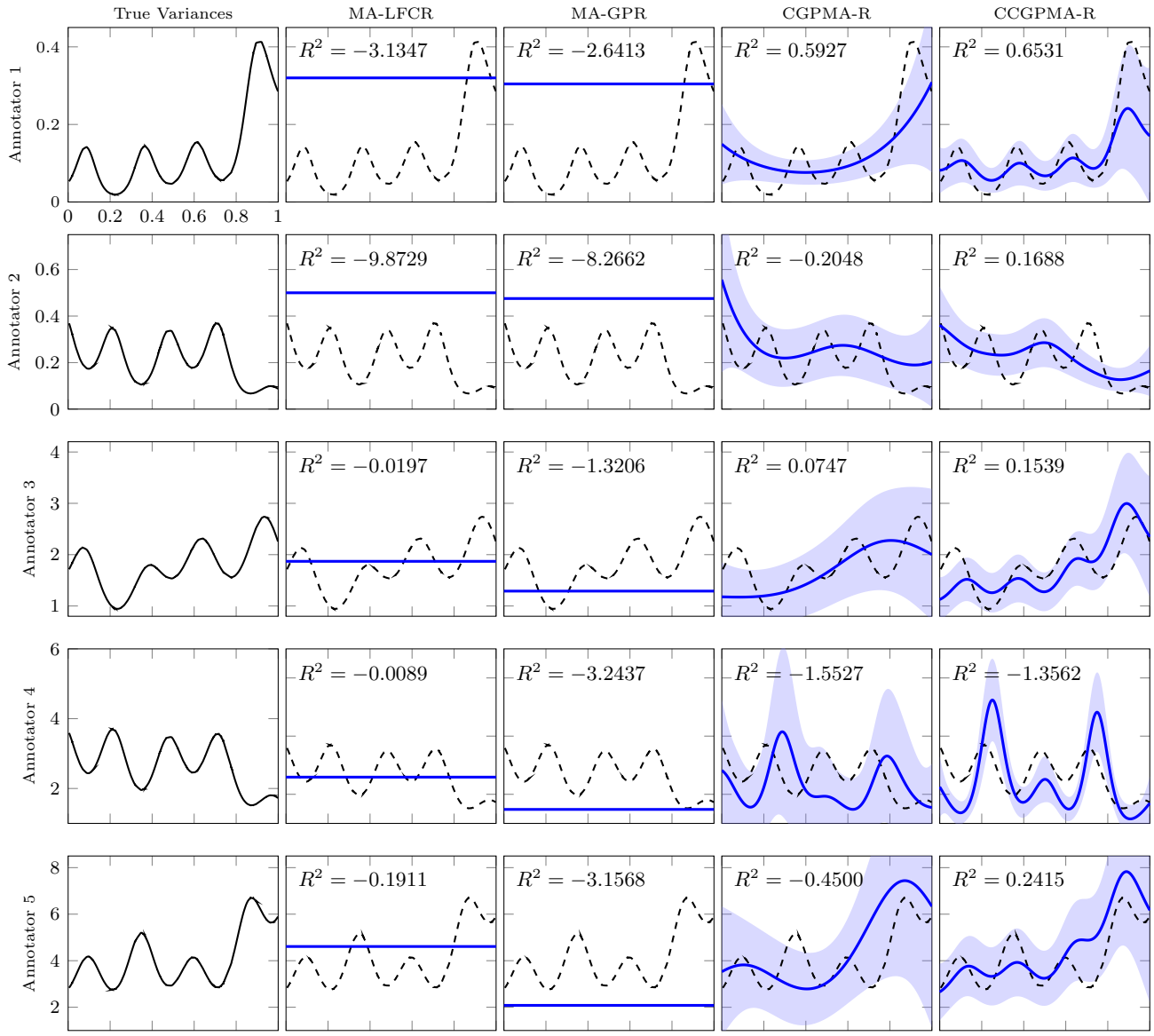


Fig. 4. Estimated values of error-variance for the five annotators in the *fully synthetic* experiment. In the first column, from top to bottom, we expose the error-variances used to simulate the labels from each annotator. Furthermore, the subsequent columns from top to bottom present the estimation of such error-variances performed by state-of-the-art models that include these kinds of parameters in their formulation; moreover, the true error-variances are provided in dashed lines. The shaded region in CGPMA-R and CCGPMA-R indicates the area enclosed by the mean plus or minus two standard deviations. We remark that there is no shaded region for MA-LFCR, and MA-GPR since these approaches perform a fixed-point estimation for the annotators' parameters. Finally, we remark that the R^2 score between the true and estimated error variances are provided.

this behavior to the fact that the CrowdLayer (used to manage the data from multiple annotators) does not offer a suitable codification of the labelers' behavior. Nevertheless, taking the above into account, we observe a remarkable result in the Bike dataset. The DL-based approaches offer the best performance, even defeating the supposed upper-bound GPR-GOLD. To explain that, it is necessary to analyze the meaning of the target variable in such a dataset. Regarding the description of this dataset,[6] the target variables indicate the count of total rental bikes, including both casual and registered in a day. The above suggests that there may exist a quasi-periodic structure in the dataset, which the GPR-GOLD cannot capture since it uses a non-periodic kernel (RBF). To support our suppositions, an additional experiment was performed over this dataset by training the model GPR-GOLD with the following kernel:

$$\kappa(\boldsymbol{x}_n, \boldsymbol{x}_{n'}) = \varphi \exp\left[-\frac{1}{2}\sum_{p=1}^{P}\left(\frac{\sin\left(\frac{\pi(x_{p,n}-x_{p,n'})}{T_p}\right)}{l_p}\right)^2\right], \quad (39)$$

where $\varphi \in \mathbb{R}$ is the variance parameter, $l_p \in (\mathbb{R}^+)$ is the length-scale parameter for the p-th dimension, and $T_p \in (\mathbb{R}^+)$ is the period for the p-th dimension. Therefore, we obtain an R^2 score equal to 0.5952 (on average), which is greater than

[6] https://archive.ics.uci.edu/ml/datasets/bike+sharing+dataset

the obtained by the DL-based approaches, indicating a quasi-periodic structure in the Bike dataset as we had supposed.

3) Fully real data results: Finally, we use the *fully real datasets*, which present the most challenging scenario, where both the input samples and the labels come from real-world applications. Table IX outlines the achieved performances. We

TABLE IX
REGRESSION RESULTS IN TERMS OF R^2 SCORE OVER *fully real dataset*. BOLD: THE HIGHEST R^2 EXCLUDING THE UPPER BOUND GPR-GOLD.

Method	Music
GPR-GOLD($M = 40$)	0.4704
GPR-GOLD($M = 80$)	0.4889
GPR-Av($M = 40$)	0.2572
GPR-Av($M = 80$)	0.2744
MA-LFCR	0.1404
MA-GPR	0.0090
MA-DL-B	0.2339
MA-DL-S	0.2934
MA-DL-B+S	0.3519
CGPMA-R($M = 40$)	0.3345
CGPMA-R($M = 80$)	0.3531
CCGPMA-R($M = 40$)	0.3337
CCGPMA-R($M = 80$)	**0.3872**

remark that our CCGPMA-R with $M = 80$ obtains the best generalization performance in terms of the R^2 score. Further, as theoretically expected, its performance lies between that of GPR-GOLD and GP-Av. Moreover, regarding the GPs-based competitors (MA-GPR and CGPMA-R), we note that our CGPMA-R is just a bit lower than CCGPMA-R. On the other hand, MA-GPR exhibits the worst prediction capability with a R^2 close to zero. We suppose the above is a symptom of overfitting, which can be confirmed because the training R^2 score for MA-GPR is 0.4731, comparable with GPR-GOLD. Conversely, the linear approach MA-LFCR exhibits the second-lowest performance and performs worse than the theoretical lower bound GP-Av, which indicates a non-linear structure in the Music dataset. Finally, analyzing the results from the deep learning approaches, we note that the variation MA-DL-B+S exhibits a similar performance compared with our CGPMA-R; however, it is slightly lower than our CCGPMA-R. We highlight that despite deep learning capacities, our approach CCGPMA-R offers a better representation of annotators' behavior, unlike the deep learning techniques, which measure such performance using a single parameter.

Also, we observe that all regression models presented a lower generalization performance than previous results (see Table V in the paper) over the same dataset. The above is a repercussion of solving a multi-class classification problem with regression models. Such an outcome is not uncommon, and it can be founded in works (18; 15).

VI. CONCLUSION

This paper introduces a novel Gaussian Process-based approach to deal with Multiple Annotators scenarios, termed Correlated Chain Gaussian Process for Multiple Annotators (CCGPMA). Our method is built as an extension of the chained GP (27), introducing a semi-parametric latent factor model (SLFM) to exploit correlations between the GP latent functions that model the parameters of a given likelihood function. To the best of our knowledge, CCGPMA is the first attempt to build a probabilistic framework that codes the annotators' expertise as a function of the input data and exploits the correlations among the labelers' answers. Besides, we highlight that our approach can be used with different likelihood, which allows us to deal with both categorical data (classification) and real-valued (regression). We tested our approach for classification tasks using different scenarios concerning the provided annotations: synthetic, semi-synthetic, real-world experts. According to the results, we remark that our CCGPMA can achieve robust predictive properties for the studied datasets, outperforming state-of-the-art methods.

As future work, CCGPMA can be extended by using convolution processes (48) instead of the SLFM, aiming to obtain a better representation of the correlations among the labelers. Also, our approach can be extended for multi-task learning in the context of multiple annotators (49). Finally, we note that the performance of our approach heavily depend on kernel selection (see Section V-B2); accordingly, it would be interesting to automatically perform such kernel selection (50) as an input block of our framework.

ACKNOWLEDGMENT

Under grants provided by the Minciencias project: "Desarrollo de un prototipo funcional para el monitoreo no intrusivo de vehículos usando data analitycs para innnovar en el proceso de mantenimiento basado en la condición en empresas de transporte público."-code 643885271399. J. Gil is funded by the program "Doctorados Nacionales - Convocatoria 785 de 2017". MAA has been financed by the EPSRC Research Projects EP/R034303/1 and EP/T00343X/1. MAA has also been supported by the Rosetrees Trust (ref: A2501). A.M. Alvarez is financed by the project "Prototipo de interfaz cerebro-computador multimodal para la detección de patrones relevantes relacionados con trastornos de impulsividad" (Universidad Nacional de Colombia - code 50835).

REFERENCES

[1] J. Zhang, V. S. Sheng, and J. Wu, "Crowdsourced label aggregation using bilayer collaborative clustering," *IEEE Transactions on Neural Networks and Learning Systems*, vol. 30, no. 10, pp. 3172–3185, 2019.

[2] Y. Liu, W. Zhang, Y. Yu *et al.*, "Truth inference with a deep clustering-based aggregation model," *IEEE Access*, vol. 8, pp. 16 662–16 675, 2020.

[3] Y. Kara, G. Genc, O. Aran, and L. Akarun, "Modeling annotator behaviors for crowd labeling," *Neurocomputing*, vol. 160, pp. 141–156, 2015.

[4] R. Snow, B. O'Connor, D. Jurafsky, and A. Ng, "Cheap and fast—but is it good?: evaluating non-expert annotations for natural language tasks," in *EMNLP*. ACL, 2008, pp. 254–263.

[5] D. Tao, J. Cheng, Z. Yu, K. Yue, and L. Wang, "Domain-weighted majority voting for crowdsourcing," *IEEE transactions on neural networks and learning systems*, vol. 30, no. 1, pp. 163–174, 2018.

[6] G. Rizos and B. W. Schuller, "Average jane, where art thou?–recent avenues in efficient machine learning under subjectivity uncertainty," in *International Conference on Information Processing and Management of Uncertainty in Knowledge-Based Systems*. Springer, 2020, pp. 42–55.

[7] P. Morales-Álvarez, P. Ruiz, S. Coughlin, R. Molina, and A. K. Katsaggelos, "Scalable variational Gaussian processes for crowdsourcing: Glitch detection in LIGO," *arXiv preprint arXiv:1911.01915*, 2019.

[8] J. Zhang, X. Wu, and V. S. Sheng, "Imbalanced multiple noisy labeling," *IEEE Transactions on Knowledge and Data Engineering*, vol. 27, no. 2, pp. 489–503, 2014.

[9] A. Dawid and A. Skene, "Maximum likelihood estimation of observer error-rates using the EM algorithm," *Appl. Stat.*, pp. 20–28, 1979.

[10] P. Ruiz, P. Morales-Álvarez, R. Molina, and A. K. Katsaggelos, "Learning from crowds with variational Gaussian processes," *Pattern Recognition*, vol. 88, pp. 298–311, 2019.

[11] V. C. Raykar, S. Yu, L. H. Zhao, G. H. Valadez, C. Florin, L. Bogoni, and L. Moy, "Learning from crowds," *J. Speech Lang. Hear. Res.*, vol. 11, pp. 1297–1322, 2010.

[12] F. Rodrigues, F. C. Pereira, and B. Ribeiro, "Gaussian process classification and active learning with multiple annotators." in *ICML*, 2014, pp. 433–441.

[13] J. Gil, M. Álvarez, and Á. Orozco, "Automatic assessment of voice quality in the context of multiple annotations," in *EMBC*. IEEE, 2015, pp. 6236–6239.

[14] P. Groot, A. Birlutiu, and T. Heskes, "Learning from multiple annotators with Gaussian processes," in *ICANN*. Springer, 2011, pp. 159–164.

[15] F. Rodrigues, M. Lourenco, B. Ribeiro, and F. Pereira, "Learning supervised topic models for classification and regression from crowds," *IEEE transactions on PAMI*, 2017.

[16] F. Rodrigues, F. Pereira, and B. Ribeiro, "Sequence labeling with multiple annotators," *Machine learning*, vol. 95, no. 2, pp. 165–181, 2014.

[17] S. Albarqouni, C. Baur, F. Achilles, V. Belagiannis, S. Demirci, and N. Navab, "Aggnet: deep learning from crowds for mitosis detection in breast cancer histology images," *IEEE transactions on medical imaging*, vol. 35, no. 5, pp. 1313–1321, 2016.

[18] F. Rodrigues and F. C. Pereira, "Deep learning from crowds," in *Thirty-Second AAAI Conference on Artificial Intelligence*, 2018.

[19] M. Y. Guan, V. Gulshan, A. M. Dai, and G. E. Hinton, "Who said what: Modeling individual labelers improves classification," in *Thirty-Second AAAI Conference on Artificial Intelligence*, 2018.

[20] E. Rodrigo, J. Aledo, and J. Gámez, "Machine learning from crowds: a systematic review of its applications," *Wiley Interdisciplinary Reviews: Data Mining and Knowledge Discovery*, vol. 9, no. 2, p. e1288, 2019.

[21] M. Venanzi, J. Guiver, G. Kazai, P. Kohli, and M. Shokouhi, "Community-based Bayesian aggregation models for crowdsourcing," in *Proceedings of the 23rd international conference on World wide web*. ACM, 2014, pp. 155–164.

[22] W. Tang, M. Yin, and C.-J. Ho, "Leveraging peer communication to enhance crowdsourcing," in *The World Wide Web Conference*. ACM, 2019, pp. 1794–1805.

[23] P. Zhang and Z. Obradovic, "Learning from inconsistent and unreliable annotators by a Gaussian mixture model and Bayesian information criterion," in *Joint European Conference on Machine Learning and Knowledge Discovery in Databases*. Springer, 2011, pp. 553–568.

[24] J. Surowiecki, *The wisdom of crowds*. Anchor, 2005.

[25] U. Hahn, M. von Sydow, and C. Merdes, "How communication can make voters choose less well," *Topics in cognitive science*, 2018.

[26] T. Zhu, M. A. Pimentel, G. D. Clifford, and D. A. Clifton, "Unsupervised bayesian inference to fuse biosignal sensory estimates for personalising care," *IEEE J. BIOMED. HEALTH*, vol. 23, no. 1, p. 47, 2019.

[27] A. Saul, J. Hensman, A. Vehtari, and N. Lawrence, "Chained Gaussian processes," in *Artificial Intelligence and Statistics*, 2016, pp. 1431–1440.

[28] M. A. Álvarez, L. Rosasco, N. D. Lawrence *et al.*, "Kernels for Vector-Valued functions: A review," *Foundations and Trends® in Machine Learning*, vol. 4, no. 3, pp. 195–266, 2012.

[29] Y. Teh, M. Seeger, and M. Jordan, "Semiparametric latent factor models," in *AISTATS 2005-Proceedings of the 10th International Workshop on Artificial Intelligence and Statistics*, 2005.

[30] M. Álvarez, D. Luengo, M. Titsias, and N. D. Lawrence, "Efficient multioutput Gaussian processes through variational inducing kernels," in *Proceedings of the Thirteenth International Conference on Artificial Intelligence and Statistics*, 2010, pp. 25–32.

[31] M. D. Hoffman, D. M. Blei, C. Wang, and J. Paisley, "Stochastic variational inference," *The Journal of Machine Learning Research*, vol. 14, no. 1, pp. 1303–1347, 2013.

[32] F. Rodrigues, F. Pereira, and B. Ribeiro, "Learning from multiple annotators: Distinguishing good from random labelers," *Pattern Recognition Letters*, vol. 34, no. 12, pp. 1428–1436, 2013.

[33] Y. Yan, R. Rosales, G. Fung, R. Subramanian, and J. Dy, "Learning from multiple annotators with varying expertise," *Machine learning*, vol. 95, no. 3, pp. 291–327, 2014.

[34] X. Wang and J. Bi, "Bi-convex optimization to learn classifiers from multiple biomedical annotations," *IEEE/ACM transactions on computational biology and bioinformatics*, vol. 14, no. 3, pp. 564–575, 2016.

[35] H. Xiao, H. Xiao, and C. Eckert, "Learning from multiple observers with unknown expertise," in *Pacific-Asia Conference on Knowledge Discovery and Data Mining*. Springer, 2013, pp. 595–606.

[36] J. Gil-Gonzalez, A. Alvarez-Meza, and A. Orozco-Gutierrez, "Learning from multiple annotators using kernel alignment," *Pattern Recognition Letters*, vol. 116, pp. 150–156, 2018.

[37] P. Morales-Álvarez, P. Ruiz, R. Santos-Rodríguez, R. Molina, and A. K. Katsaggelos, "Scalable and efficient learning from crowds with Gaussian processes," *Information Fusion*, vol. 52, pp. 110–127, 2019.

[38] G. Hua, C. Long, M. Yang, and Y. Gao, "Collaborative active visual recognition from crowds: A distributed ensemble approach," *IEEE transactions on pattern analysis and machine intelligence*, vol. 40, no. 3, pp. 582–594, 2018.

[39] C. E. Rasmussen and C. K. Williams, *Gaussian processes for machine learning*. MIT press Cambridge, 2006, vol. 1.

[40] J. Hensman, A. G. Matthews, and Z. Ghahramani, "Scalable variational Gaussian process classification," *Proceedings of Machine Learning Research*, vol. 38, pp. 351–360, 2015.

[41] P. Moreno-Muñoz, A. Artés, and M. Alvarez, "Heterogeneous multi-output Gaussian process prediction," in *Advances in neural information processing systems*, 2018, pp. 6711–6720.

[42] D. M. Blei, A. Kucukelbir, and J. D. McAuliffe, "Variational inference: A review for statisticians," *Journal of the American statistical Association*, vol. 112, no. 518, pp. 859–877, 2017.

[43] J. A. Hernández-Muriel, J. B. Bermeo-Ulloa, M. Holguin-Londoño, A. M. Álvarez-Meza, and Á. A. Orozco-Gutiérrez, "Bearing health monitoring using relief-f-based feature relevance analysis and HMM," *Applied Sciences*, vol. 10, no. 15, p. 5170, 2020.

[44] J. Arias, J. Godino, J. Gutiérrez, V. Osma, and N. Sáenz, "Automatic GRBAS assessment using complexity measures and a multiclass GMM-based detector," *Models and Analysis of Vocal Emissions for Biomedical Applications*, pp. 111–114, 2011.

[45] L. v. d. Maaten and G. Hinton, "Visualizing data using t-SNE," *Journal of machine learning research*, vol. 9, pp. 2579–2605, 2008.

[46] T. Fawcett, "An introduction to ROC analysis," *Pattern recognition letters*, vol. 27, no. 8, pp. 861–874, 2006.

[47] C. Bishop, *Pattern recognition and machine learning*. springer, 2006.

[48] M. A. Álvarez and N. D. Lawrence, "Computationally efficient convolved multiple output Gaussian processes," *The Journal of Machine Learning Research*, vol. 12, pp. 1459–1500, 2011.

[49] M. A. Alvarez, L. Rosasco, and N. D. Lawrence, "Kernels for vector-valued functions: A review," *arXiv preprint arXiv:1106.6251*, 2011.

[50] A. B. Abdessalem, N. Dervilis, D. J. Wagg, and K. Worden, "Automatic kernel selection for gaussian processes regression with approximate bayesian computation and sequential monte carlo," *Frontiers in Built Environment*, vol. 3, p. 52, 2017.

J. Gil-Gonzalez received his undergraduate degree in electronic engineering (2014) from the Universidad Tecnológica de Pereira, Colombia. His M.Sc. in electrical engineering (2016) from the same university. Currently, he is PhD student from the same university. His research interests include probabilistic models for machine learning, learning from crowds, and Bayesian inference.

Juan-José Giraldo Received a degree in Electronics Engineering (B. Eng.) with Honours, from Universidad del Quindío, Colombia in 2009, a master degree in Electrical Engineering (M. Eng.) from Universidad Tecnológica de Pereira, Colombia in 2015. Currently, Mr. Giraldo is a Ph.D student in Comp. Science at the University of Sheffield, UK

A.M. Álvarez-Meza received his undergraduate degree in electronic engineering (2009), his M.Sc. degree in engineering (2011), and his Ph.D. in automatics from the Universidad Nacional de Colombia. He is a Professor in the Department of Electrical, Electronic and Computation Engineering at the Universidad Nacional de Colombia - Manizales. His research interests include machine learning and signal processing.

A. Orozco-Gutierrez received his undergraduate degree in electrical engineering (1985) and his M.Sc. degree in electrical engineering (2004) from Universidad Tecnológica de Pereira, and his Ph.D. in bioengineering (2009) from Universidad Politecnica de Valencia (Spain). He received his undergraduate degree in law (1996) from Universidad Libre de Colombia. He is a Professor in the Department of Electrical Engineering at the Universidad Tecnologica de Pereira. His research interests include bioengineering.

M. A. Álvarez received the BEng degree in electronics engineering from the Universidad Nacional de Colombia (2004), the M.Sc. degree in electrical engineering from the Universidad Tecnologica de Pereira, Colombia (2006), and the PhD degree in computer science from The University of Manchester, UK (2011). Currently, he is a Lecturer of Machine Learning in the Department of Computer Science, University of Sheffield, United Kingdom. His research interests include probabilistic models, kernel methods, and stochastic processes.

Appendix M

Paper iii: Correlated Chained Gaussian Processes for Modelling Citizens Mobility using a Zero-inflated Poisson Likelihood

Paper under review in the Journal IEEE Transactions on Intelligent Transportation Systems.

Correlated Chained Gaussian Processes for Modelling Citizens Mobility using a Zero-Inflated Poisson Likelihood

Juan-José Giraldo, Jie Zhang, and Mauricio A. Álvarez

***Abstract*—Modelling the mobility of people in a city depends on counting data with inherent problems of overdispersion. Such dispersion issues are caused by massive amounts of data with zero values. Though traditional machine learning models have been used to overcome said problems, they lack the ability to appropriately model the spatio-temporal correlations in data. To improve the modelling of such spatio-temporal correlations, in this work we propose to model the citizens mobility, for the Chinese city of Guangzhou, by means of a Zero-inflated Poisson likelihood in conjunction with Gaussian process priors generated from convolution processes. We follow the idea of chaining the likelihood's parameters to latent functions drawn from Gaussian process priors; this way allowing a higher flexibility to model heteroscedasticity. Additionally, we derive a stochastic variational inference framework that allow us to use two types of convolution process models in the context of large datasets: 1. correlated chained Gaussian processes with a convolution processes model, and 2. correlated chained Gaussian processes with variational inducing kernels. We reproduce quantitative and qualitative results comparing the performance between Poisson and Zero-inflated Poisson likelihoods, both in combination with three types of Gaussian process priors: a linear model of coregionalisation, and our two proposed methods based on a convolution processes model and variational inducing kernels.**

***Index Terms*—Citizens Mobility, Correlated Chained Gaussian Processes, Zero-inflated Poisson, Convolution Processes, Variational Inducing Kernels, Stochastic Variational Inference.**

I. Introduction

MODELLING the mobility of persons in a city depends on counting data that inherently involve problems of overdispersion. Such overdispersion issues are caused by an excess of observations with values at zero, i.e., zero-inflated counts [1]. In order to tackle those issues, different types of machine learning models have focused on fitting the excessive dispersion in data caused by the zero-inflation. For instance, via Generalised Linear Models (GLMs) [2] using likelihoods like the zero-inflated Poisson (ZIP or ZI-Poisson) [3], [4], the zero-inflated negative Binomial (ZINB) [3], or the Tweedie distribution [5], [6], etc. Though these models have been useful to overcome the problems associated to the zero-inflation, they still lack of the ability to appropriately model the spatio-temporal correlations of data associated to mobility. A more powerful alternative for exploiting such spatio-temporal correlations relies on Gaussian process (GP) models; nonetheless, few works have taken advantage of their application to improve the forecasting for zero-inflated data. For instance, the work in [7] proposes the use of GPs together with a Zero-inflated Poisson likelihood for the analysis of sickness absence; the authors discuss that GP models might yield better predictive performance than hurdle models. Although they did not implement them because of numerical stability issues. The authors in [8] propose a zero-inflated formalism that consists of a Gaussian likelihood whose mean follows a latent GP, and a separate 'on-off' probit-linked GP for generating a sparse kernel that allows the model to predict zeros; this work lacks of capturing heteroscedastic noise (an inherent trait of counting data), this due to assuming a Gaussian likelihood where the noise variance is considered constant along all the observations. Also, in [9], the authors use a ZINB likelihood with a GP prior to model temporal and spatial counting data from RNA-sequencing experiment; this approach presents a unique latent function that models the mean of the Negative Binomial term with a GP prior, while the dispersion parameter and the so called Michalis parameter are assumed free parameters. To the best of our knowledge there are not other works based on GPs that have been concerned about solving the zero-inflation issues while exploiting the spatio-temporal correlations, but the ones mentioned before.

In this work, we aim to model the citizens mobility in the Chinese city of Guangzhou, this from counting data of persons present at a delta area of the city. Since there are not previous works that explore the behaviour of the Zero-inflated Poisson likelihood with GP priors, here we concentrate on such a likelihood; a ZIP likelihood is more appropriate than the Gaussian likelihood used in [8], given that the statistical data type of the observations are counts, i.e., non-negative values. Also, though the work in [9] considers the statistical data type of counts by assuming a ZINB likelihood, it only uses a GP prior to model the mean of the Negative Binomial term while the other parameters are considered a constant, this way limiting the modelling flexibility. Unlike this latter work, here we propose that the likelihood's parameters are modelled as latent functions drawn from correlated GP priors, this way allowing a higher flexibility to model heteroscedasticity.

In the context of GP models, where each parameter of the likelihood is chained to a GP latent function, three ways to generate such latent functions include: 1. each latent function follows an independent GP prior [10]; 2. each latent function is generated from a linear model of coregionalisation (LMC), i.e.,

J. J. Giraldo and M. A. Álvarez are with the Department of Computer Science, The University of Sheffield, UK; J. Zhang is with the Department of Electronic and Electrical Engineering, The University of Sheffield, UK. (e-mail: jjgiraldogutierrez1@sheffield.ac.uk, mauricio.alvarez@sheffield.ac.uk, jie.zhang@sheffield.ac.uk)

a weighted sum of GP priors [11], [12]; and 3. from convolution processes, i.e., a convolution integral between smoothing kernels and GP priors [13], [14], [15]. The above generative alternatives for the latent functions have been broadly used to model either a single or multiple outputs in diverse application scenarios [11]; particularly, the independent GP priors have been used in applications that require the modelling of a single output [16]. In the specific ambit of modelling urban traffic in different areas of a city, the work in [17] focuses on forecasting vehicles traffic speeds using an intrinsic coregionalisation model (a particular case of the LMC). Also, the work in [18] uses a model based on convolution processes to fit spatial and temporal patterns in crowdsourced traffic data. Nevertheless, these previous works become prohibitive in the context of a large number of data observations. To tackle such scalability issues, we additionally derive a stochastic variational inference (SVI) [19], [20] framework that allows the use of this type of models when having massive amounts of data observations. Besides the convolution processes model, we also introduce a scalable version of the variational inducing kernels (VIKs) approach [14]. This VIKs approach is an alternative form to generate the LPFs through the convolution processes formalism, by using a double convolution integral; i.e., the LPF is drawn from a convolution integral between a smoothing kernel and an *inducing function* (IF), where such an IF is an artificial construction generated from another convolution integral between a smoothing kernel and a GP prior.

The main contributions of this work include the following:

- We model the citizens mobility in the Chinese city of Guangzhou, through the use of a ZIP likelihood in conjunction with GP priors. To the best of our knowledge, a ZIP likelihood has not been previously implemented together with a GP model.
- Unlike previous works based on GPs that mainly model the mean parameter of the likelihood with a unique GP prior, here we propose that each of those likelihood's parameters are modelled as LPFs that follow correlated GPs; thus, allowing a higher flexibility to model heteroscedasticity.
- We derive an SVI framework that allow us to use two types of convolution process models in the context of large datasets: 1. correlated chained GP (CCGP) with a convolution processes model (CPM), and 2. CCGP with VIKs.
- Former works have not developed GP models based on CPM and VIKs for other type of likelihoods beyond a Gaussian. In this work, we derive equations that can be used for any type of likelihood. Particularly, we provide results for both CCGP models based on CPM and VIKs for ZIP and Poisson likelihoods.

II. Chained Gaussian Processes model

A GP is a non-parametric stochastic process that extends a multivariate normal probability distribution from finite dimensional vectors to functions [21]. Let us define a collection of N data observations with a matrix of inputs $\mathbf{X} = [\mathbf{x}_1, ..., \mathbf{x}_N]^\top \in \mathbb{R}^{N\times P}$ and a vector of outputs $\mathbf{y} = [y_1, ..., y_N]^\top$, where, for instance, each $\mathbf{x}_n$ might represent a spatio-temporal observation associated to a measurement y_n. Usually, for a regression model, each observation y_n is modelled as a noisy version of a latent function evaluated at the n-th input observation, $f(\mathbf{x}_n)$. All data observations can be modelled by means of a likelihood function, $\prod_{n=1}^{N} p(y_n|f(\mathbf{x}_n))$, where the latent function follows a GP prior, i.e., $f(\mathbf{x}) \sim \mathcal{GP}(m(\mathbf{x}), k(\mathbf{x}, \mathbf{x}'))$. The GP is characterised by a mean function $m(\mathbf{x}) = \mathbb{E}[f(\mathbf{x})]$ and a kernel covariance function $k(\mathbf{x}, \mathbf{x}') = \mathrm{Cov}[f(\mathbf{x}), f(\mathbf{x}')] = \mathbb{E}[f(\mathbf{x})f(\mathbf{x}')] - \mathbb{E}[f(\mathbf{x})]\mathbb{E}[f(\mathbf{x}')]$; here $\mathrm{Cov}\,[\cdot,\cdot]$ represents a covariance function. Such a kernel determines the nature of the latent functions involved in a GP model, for instance, through the kernel we can induce latent functions with: smoothness, periodicity, stationarity, non-stationarity, etc. It is important to emphasize that a kernel is a covariance function that depends on a set of hyper-parameters, that generally have to be fitted during an optimisation process when training the model. For example, a popular covariance option is the exponentiated quadratic kernel, $k(\mathbf{x}, \mathbf{x}') = \sigma_f^2 \exp\left(-\frac{\|\mathbf{x}-\mathbf{x}'\|^2}{2l^2}\right)$, which depends on the hyper-parameters σ_f^2 and l that control the amplitude and length-scale of the latent functions, respectively.

Probably, the best known GP regression model is based on a Gaussian likelihood, $\prod_{n=1}^{N} \mathcal{N}(y_n|f(\mathbf{x}_n), \sigma_\epsilon^2)$, which can be understood as assuming that each observed value y_n is a version of the GP latent function, $f(\mathbf{x}_n)$, corrupted by an independent Gaussian noise $\epsilon \sim \mathcal{N}(0, \sigma_\epsilon^2)$ [2][21]. This approach models the likelihood's mean parameter with a GP latent function, while σ_ϵ^2 is treated as a hyper-parameter. Since the noise variance of the observations y_n is not necessarily a constant, treating σ_ϵ^2 as a hyper-parameter limits the flexibility of the model for capturing heteroscedasticity. On the other hand, setting a likelihood function directly depends on the statistical data type of the observations $\mathbf{y}$. A more general case, regardless the type of distribution $p(y_n|f(\mathbf{x}_n))$, consists on chaining a GP prior to each parameter of the likelihood as follows:

$$p(\mathbf{y}|\mathbf{f}) = \prod_{n=1}^{N} p\left(y_n|\psi_1\left(\mathbf{x}_n\right), ..., \psi_J\left(\mathbf{x}_n\right)\right), \tag{1}$$

where each $\psi_j\left(\mathbf{x}_n\right) = \alpha(f_j(\mathbf{x}_n))$ represents the j-th parameter of the likelihood chained to a latent GP prior $f_j(\cdot)$ through a link function $\alpha(\cdot)$ [10], and J represents the number of likelihood's parameters. We will refer to each $f_j(\cdot)$ as a latent parameter function (LPF). In the equation above $\mathbf{f} = [\mathbf{f}_1^\top, ..., \mathbf{f}_J^\top]^\top$ is a vector that stacks all the LPFs, with $\mathbf{f}_j = [f_j(\mathbf{x}_1), ..., f_j(\mathbf{x}_N)]^\top$. From a GPs perspective, we aim to model each LPF in the likelihood's equation above as a GP prior. There can be different generative ways to build the GP priors. For instance, the work in [10] assumes that each LPF follows an independent GP prior, but such an assumption does not allow the model to capture possible correlations between the likelihood's parameters [11]. Instead of the independence supposition, the work in [12] uses a linear model of coregionalisation [22], to generate correlated LPFs from a weighted sum of GP priors; this model scales the computational complexity by applying SVI. Another common approach to model the LPFs, particularly used for multi-output

regression, relies on using convolution processes by means of solving a convolution integral between a smoothing kernel and a GP [14], [11]. Nonetheless, this latter work has not been developed to deal with a big number of data observations neither for other type of likelihoods beyond a Gaussian. In the following subsections, we will describe how to obtain tractable variational bounds permitting to scale the convolutional model for being trained in the context of a large number of data observations even when the LPFs are correlated.

III. Correlated Chained GP with a Convolution Processes Model

This section explains the construction of a chained GPs [10] model that introduces correlations between the GP latent functions through the use of Convolution Processes. We term this type of model as the Correlated Chained GP with convolution processes [23], [13]. Also, we explain how the inducing variables approach allows the model to obtain tractable variational bounds suitable to SVI.

A. Convolution Processes for Generating the LPFs

A more general way to derive the latent parameter functions relies on the convolution processes model [14], [11]. In this type of model, the LPFs are generated by convolving Q latent processes $u_q(\cdot)$ with smoothing kernels $G_{j,q}(\cdot)$, i.e., $f_j(\mathbf{x}) = \sum_{q=1}^{Q} \int_{\mathcal{X}} G_{j,q}(\mathbf{x}-\mathbf{r}')\, u_q(\mathbf{r}')\, \mathrm{d}\mathbf{r}'$. Alternatively, we can express the latter equation as influenced by multiple latent functions $u_{q,i}(\cdot)$:

$$f_j(\mathbf{x}) = \sum_{q=1}^{Q} \sum_{i=1}^{R_q} \int_{\mathcal{X}} G_{j,q,i}(\mathbf{x}-\mathbf{r}')\, u_{q,i}(\mathbf{r}')\, \mathrm{d}\mathbf{r}', \tag{2}$$

where each $u_{q,i}(\cdot)$ represents a latent function drawn independent and identically distributed (IID) from $u_q(\cdot) \sim \mathcal{GP}(0, k_q(\cdot,\cdot))$; and R_q represents the number of said IID samples drawn per q-th latent function $u_q(\cdot)$ [15]. Notice that the equation above is analogous to the linear model of coregionalisation, where there are usually Q groups of latent functions $u_q(\cdot)$, and each IID sample $u_{q,i}(\cdot)$ has the same covariance $k_q(\cdot,\cdot)$ [22]. To ease the derivations in the following sections, we will refer to R instead of R_q, i.e., the number of samples $u_{q,i}(\cdot)$ per q-th latent function $u_q(\cdot)$ is the same for all Q groups.

B. Augmented Gaussian Process Prior

Inference in GP models is computationally expensive. A common approach to improve the computational complexity is to augment the GP prior with a set of *inducing variables* $u(\cdot)$. Such *inducing variables* represent additional function evaluations of some unknown inducing points $\mathbf{Z} = [\mathbf{Z}_1^\top, ..., \mathbf{Z}_Q^\top]^\top \in \mathbb{R}^{QM\times P}$, with $\mathbf{Z}_q = [\mathbf{z}_q^{(1)}, ..., \mathbf{z}_q^{(M)}]^\top \in \mathbb{R}^{M\times P}$ [24], [25]. We can write the augmented GP prior as follows,

$$p(\mathbf{f}|u)p(u|\mathbf{u})p(\mathbf{u}) = \prod_{j=1}^{J} p(\mathbf{f}_j|u)p(u|\mathbf{u})p(\mathbf{u}), \tag{3}$$

where $u = [u_{1,1}^\top, ..., u_{Q,1}^\top, ..., u_{1,R}^\top, ..., u_{Q,R}^\top]^\top$ represents a vector of functions that stacks all R IID samples $u_{q,i}(\cdot)$ of all groups Q; here, u is seen as a continuous version infinitely evaluated at all possible values $\mathbf{x} \in \mathbb{R}^P$. A finite evaluation of u, for instance over the set of inducing points, is expressed as $\mathbf{u} = [\mathbf{u}_{1,1}^\top, ..., \mathbf{u}_{Q,1}^\top, ..., \mathbf{u}_{1,R}^\top, ..., \mathbf{u}_{Q,R}^\top]^\top \in \mathbb{R}^{QMR\times 1}$ with $\mathbf{u}_{q,i} = [u_{q,i}(\mathbf{z}_q^{(1)}), \ldots, u_{q,i}(\mathbf{z}_q^{(M)})]^\top \in \mathbb{R}^{M\times 1}$ [26], [14]. Particularly, the distributions of the GP prior follow the form: $p(\mathbf{f}_j|u) = \mathcal{N}(m_{f_j u}(\mathbf{X}), \mathbf{0}) = \delta(\mathbf{f}_j - m_{f_j u}(\mathbf{X}))$, where $m_{f_j u}(\mathbf{X}) = [m_{f_j u}(\mathbf{x}_1), ..., m_{f_j u}(\mathbf{x}_N)]^\top \in \mathbb{R}^{N\times 1}$ is a vector built from:

$$m_{f_j u}(\mathbf{x}) = \sum_{q=1}^{Q} \sum_{i=1}^{R} \int_{\mathcal{X}} G_{j,q,i}(\mathbf{x}-\mathbf{r}')\, u_{q,i}(\mathbf{r}')\, \mathrm{d}\mathbf{r}',$$

and $p(u|\mathbf{u}) = \mathcal{N}(u|k_{u\mathbf{u}}\mathbf{K}_{\mathbf{uu}}^{-1}\mathbf{u}, V_u)$ is a distribution over the vector of functions, conditioned on the finite vector of inducing variables $\mathbf{u}$; $\mathbf{K}_{\mathbf{uu}} \in \mathbb{R}^{QMR\times QMR}$ is a block-diagonal matrix with blocks $\mathbf{K}_{\mathbf{u}_{q,i}\mathbf{u}_{q,i}}$ which entries are calculated with $\mathrm{Cov}[u_{q,i}(\cdot), u_{q,i}(\cdot)] = k_q(\cdot,\cdot)$, between all pairs of inducing points $\mathbf{Z}_q$; $V_u = k_{uu} - k_{u\mathbf{u}}\mathbf{K}_{\mathbf{uu}}^{-1}k_{u\mathbf{u}}$, where $k_{uu} = \mathrm{Cov}[u,u]$ can be understood as a continuos matrix covariance infinitely evaluated, and $k_{u,\mathbf{u}} = \mathrm{Cov}[u,\mathbf{u}]$ is a cross-covariance matrix with continuous rows and finite columns; and finally $p(\mathbf{u}) = \mathcal{N}(\mathbf{u}|\mathbf{0}, \mathbf{K}_{\mathbf{uu}})$.

C. The Evidence Lower Bound

Similar to the works in [14][27][12], we follow a mathematical derivation based on variational inference. Such a way of derivation grant us the application of our model in the context of large datasets. Here, our aim consists on approximating the true posterior $p(\mathbf{f}, u, \mathbf{u}|\mathbf{y})$ with a variational distribution $q(\mathbf{f}, u, \mathbf{u})$ by optimising the following evidence lower bound (ELBO) [20]:

$$\mathcal{L} = \mathbb{E}_{q(\mathbf{f},u,\mathbf{u})}\Big[\log \frac{p(\mathbf{y}|\mathbf{f})p(\mathbf{f}|u)p(u|\mathbf{u})p(\mathbf{u})}{q(\mathbf{f},u,\mathbf{u})}\Big]. \tag{4}$$

We set a variational posterior as follows: $q(\mathbf{f},u,\mathbf{u}) = p(\mathbf{f}|u)p(u|\mathbf{u})q(\mathbf{u}) = \prod_{j=1}^{J} p(\mathbf{f}_j|u)p(u|\mathbf{u})q(\mathbf{u})$, for which $q(\mathbf{u}) = \mathcal{N}(\mathbf{u}|\mathbf{m},\mathbf{V})$ is a free parametrised distribution, with mean $\mathbf{m} \in \mathbb{R}^{QMR\times 1}$ and a block-diagonal covariance matrix $\mathbf{V} \in \mathbb{R}^{QMR\times QMR}$, with blocks given by $\mathbf{V}_{q,i} \in \mathbb{R}^{M\times M}$. After replacing the posterior distribution at Eq. (4) and arranging terms, we end up with the following objective for the ELBO:

$$\mathcal{L} = \sum_{n=1}^{N} \mathbb{E}_{q(\mathbf{f})}\big[g_n\big] - \mathbb{D}_{KL}\big(q(\mathbf{u})||p(\mathbf{u})\big), \tag{5}$$

where $g_n = \log p(y_n|\psi_1(\mathbf{x}_n), ..., \psi_J(\mathbf{x}_n))$ is the Log Likelihood (LL) function and $\mathbb{D}_{KL}(\cdot\,||\,\cdot)$ is a Kullback-Leibler divergence. The expectation above associated to the LL is computed with regard to the marginal posterior, $q(\mathbf{f}) = \int\int \prod_{j=1}^{J} p(\mathbf{f}_j|u)p(u|\mathbf{u})q(\mathbf{u})d\mathbf{u}du$. Solving for the integrals above, we arrive to:

$$q(\mathbf{f}) := \mathcal{N}(\mathbf{f}|\tilde{\mathbf{m}}_{\mathbf{fu}}, \tilde{\mathbf{V}}_{\mathbf{fu}}), \tag{6}$$

having the following definitions, $\tilde{\mathbf{m}}_{\mathbf{fu}} := \mathbf{A}_{\mathbf{fu}}\mathbf{m}$; $\mathbf{A}_{\mathbf{fu}} = \mathbf{K}_{\mathbf{fu}}\mathbf{K}_{\mathbf{uu}}^{-1}$; and $\tilde{\mathbf{V}}_{\mathbf{fu}} := \mathbf{K}_{\mathbf{ff}} + \mathbf{A}_{\mathbf{fu}}(\mathbf{V} - \mathbf{K}_{\mathbf{uu}})\mathbf{A}_{\mathbf{fu}}^\top$;

where $\mathbf{K_{fu}} = [\mathbf{K}_{\mathbf{f}_1\mathbf{u}}^\top, ..., \mathbf{K}_{\mathbf{f}_J\mathbf{u}}^\top]^\top \in \mathbb{R}^{JN\times QMR}$ is a cross covariance matrix built with blocks $\mathbf{K}_{\mathbf{f}_j\mathbf{u}} = [\mathbf{K}_{\mathbf{f}_j\mathbf{u}_{1,1}}, ..., \mathbf{K}_{\mathbf{f}_j\mathbf{u}_{Q,1}}, ..., \mathbf{K}_{\mathbf{f}_j\mathbf{u}_{1,R}}, ..., \mathbf{K}_{\mathbf{f}_j\mathbf{u}_{Q,R}}] \in \mathbb{R}^{N\times QMR}$, with $\mathbf{K}_{\mathbf{f}_j\mathbf{u}_{q,i}} \in \mathbb{R}^{N\times M}$ constructed with entries calculated from $\mathrm{Cov}\left[f_j(\mathbf{x}), u_{q,i}(\mathbf{z})\right]$ between the data observations $\mathbf{X}$ and the inducing points $\mathbf{Z}_q$; and $\mathbf{K_{ff}} \in \mathbb{R}^{JN\times JN}$ is a matrix built with evaluations of the covariance function $\mathrm{Cov}\left[f_j(\mathbf{x}), f_{j'}(\mathbf{x}')\right]$ between all pairs of data observations $\mathbf{X}$. In the following subsection, we describe the specific form of the covariance functions introduced above.

D. Covariance Functions for CCGP with CPM

For all our models we assume kernel covariance functions with an Exponentiated Quadratic (EQ) form as follows:

$$\mathcal{E}(\boldsymbol{\tau}|\mathbf{0}, \mathbf{L}) = \frac{|\mathbf{L}|^{-1/2}}{(2\pi)^{p/2}} \exp\left[-\frac{1}{2}\boldsymbol{\tau}^\top \mathbf{L}^{-1}\boldsymbol{\tau}\right], \tag{7}$$

where $\boldsymbol{\tau} := \mathbf{x} - \mathbf{x}'$ and $\mathbf{L}$ is a diagonal matrix of length-scales. With the above functional form we can define the following kernels for our CCGP model with CPM:

$$k_q(\mathbf{x}, \mathbf{x}') = \mathcal{E}(\boldsymbol{\tau}|\mathbf{0}, \mathbf{L}_q), \tag{8}$$

$$G_{j,q,i}(\mathbf{x}, \mathbf{x}') = S_{j,q,i}\mathcal{E}(\boldsymbol{\tau}|\mathbf{0}, \boldsymbol{\kappa}_j), \tag{9}$$

where $\mathbf{L}_q$ and $\boldsymbol{\kappa}_j$ are diagonal matrices of length-scales, and $S_{j,q,i}$ is a weight associated to the LPF $f_j(\cdot)$ and to the i-th sample of the latent function $u_q(\cdot)$. Having the definitions of our kernels, we can solve the convolution integrals associated to the covariance functions, $\mathrm{Cov}\left[f_j(\mathbf{x}), u_{q,i}(\mathbf{z})\right] = \int_{\mathcal{X}} G_{j,q,i}(\mathbf{x} - \mathbf{r}')k_q(\mathbf{r}', \mathbf{z})\mathrm{d}\mathbf{r}'$ and $\mathrm{Cov}\left[f_j(\mathbf{x}), f_{j'}(\mathbf{x}')\right] = \sum_{q=1,i=1}^{Q,R} \int_{\mathcal{X}} G_{j,q,i}(\mathbf{x} - \mathbf{r}) \int_{\mathcal{X}} G_{j',q,i}(\mathbf{x}' - \mathbf{r}')\, k_q(\mathbf{r}, \mathbf{r}')\mathrm{d}\mathbf{r}\mathrm{d}\mathbf{r}'$. To solve such integrals above, we follow the work in [15], where the authors apply methodically an identity for the product of two Gaussian distributions. This way, we arrive to: $\mathrm{Cov}\left[f_j(\mathbf{x}), u_{q,i}(\mathbf{x}')\right] = S_{j,q,i}\mathcal{E}(\boldsymbol{\tau}|\mathbf{0}, \boldsymbol{\kappa}_j + \mathbf{L}_q)$, $\mathrm{Cov}\left[f_j(\mathbf{x}) f_{j'}(\mathbf{x}')\right] = \sum_{q=1}^{Q} S_{j,q,i}S_{j',q,i}\mathcal{E}(\boldsymbol{\tau}|\mathbf{0}, \mathbf{P}_{j,j',q})$, where $\mathbf{P}_{j,j',q}$ represents a diagonal matrix of length-scales, $\mathbf{P}_{j,j',q} = \boldsymbol{\kappa}_j + \boldsymbol{\kappa}_{j'} + \mathbf{L}_q$.

E. Making Predictions with CCGP based on a CPM

To make predictions with our proposed model, we have to compute $p(\mathbf{y}_* \mid \mathbf{y}) \approx \int p(\mathbf{y}_* \mid \mathbf{f}_*)\, q(\mathbf{f}_*)\, d\mathbf{f}_*$, where $q(\mathbf{f}_*)$ can be computed using Eq. (6), but building the different covariances matrices $\mathbf{K}_{\mathbf{f}_*\mathbf{u}}$ and $\mathbf{K}_{\mathbf{f}_*\mathbf{f}_*}$ with evaluations at the new inputs $\mathbf{X}_*$ using equations from section III-D.

IV. Correlated Chained GP with Variational Inducing Kernels

This section describes the construction of the CCGP model with variational inducing kernels [14]. Also, it explains how to obtain a variational objective of the model which is suitable for training by means of SVI [19], [20].

A. Variational Inducing Kernels for Generating the LPFs

The concept of variational inducing kernels was proposed in [14] as an alternative and more powerful way of defining an inducing variable [24], [25]. It consists on applying a convolution of the latent function $u_q(\cdot)$ with a smoothing kernel as follows:

$$\lambda_q(\mathbf{z}) = \int_{\mathcal{X}} T_q(\mathbf{z} - \mathbf{r})u_q(\mathbf{r})\mathrm{d}\mathbf{r}, \tag{10}$$

where $T_q(\mathbf{z} - \mathbf{r})$ is a smoothing kernel, also known as the *inducing kernel* (IK) and $\lambda_q(\mathbf{z})$ is called an *inducing function*; and the latent function is drawn from a GP, $u_q(\cdot) \sim \mathcal{GP}(0, k(\cdot, \cdot))$. The VIKs allow us to define more general inducing variables with higher approximation capacities than the inducing variables $u_q(\cdot)$ used in Eq. (2) for the CCGP with a convolution processes model [11]. Though the motivation to use the VIKs in our work relies on the fact of increasing the predictive capabilities of our model, this approach is also useful to deal with possible white noise latent functions $u_q(\cdot)$ when applicable.

As we mentioned before in Eq. (1), each latent parameter function, $f_j(\cdot)$, aims to model the j-th parameter of the likelihood, i.e., each $\psi_j(\mathbf{x}_n) = \alpha(f_j(\mathbf{x}_n))$. Unlike the convolution processes model in Eq. (2), which is particularly based on the inducing variables $u_q(\cdot)$, the LPFs can also be drawn from a convolution integral between a smoothing kernel and an inducing function $\lambda_q(\cdot)$ as follows:

$$f_j(\mathbf{x}) = \sum_{q=1}^{Q}\sum_{i=1}^{R_q} \int_{\mathcal{X}} G_{j,q,i}(\mathbf{x} - \mathbf{r}')\, \lambda_{q,i}(\mathbf{r}')\, \mathrm{d}\mathbf{r}', \tag{11}$$

where $G_{j,q,i}(\cdot)$ represents the smoothing kernel; and $\lambda_{q,i}(\cdot)$ is an inducing function associated to the i-th sample $u_{q,i}(\cdot)$ taken IID from $u_q(\cdot) \sim \mathcal{GP}(0, k_q(\cdot, \cdot))$, i.e., as per Eq. (10): $\lambda_{q,i}(\mathbf{z}) = \int_{\mathcal{X}} T_q(\mathbf{z} - \mathbf{r})u_{q,i}(\mathbf{r})\mathrm{d}\mathbf{r}$; and R_q represents the number of IID samples drawn per q-th inducing function $\lambda_q(\cdot)$ [15]. Thereby, the equation above is an alternative approach to generate the GP priors for modelling the likelihood's parameters under the VIKs approach. As we assumed for the CPM, instead of R_q in Eq. (11), we will refer to the same number R of IID samples for all Q groups of inducing functions.

B. Augmented Gaussian Process Prior

We follow a similar inducing variables framework used for the model CCGP with Convolution processes. It is worth noticing that for the convolution processes model, the function $u(\cdot)$ is the one representing the inducing variable that augments the GP prior (see Eq. (3)). Conversely, in this case of VIKs, the vector function $\lambda = [\lambda_{1,1}^\top, ..., \lambda_{Q,1}^\top, ..., \lambda_{1,R}^\top, ..., \lambda_{Q,R}^\top]^\top$ is the one used to augment the GP prior, and from which we compute additional evaluations over the set of unknown inducing points $\mathbf{Z} = [\mathbf{Z}_1^\top, ..., \mathbf{Z}_Q^\top]^\top \in \mathbb{R}^{QM\times P}$, with $\mathbf{Z}_q = [\mathbf{z}_q^{(1)}, ..., \mathbf{z}_q^{(M)}]^\top \in \mathbb{R}^{M\times P}$ [14]. We express the augmented GP prior as follows, $p(\mathbf{f}|\lambda)p(\lambda|\boldsymbol{\lambda})p(\boldsymbol{\lambda}) = \prod_{j=1}^{J} p(\mathbf{f}_j|\lambda)p(\lambda|\boldsymbol{\lambda})p(\boldsymbol{\lambda})$, where λ, the inducing function, is a continuous function infinitely computed at all possible $\mathbf{x} \in \mathbb{R}^P$; whilst a finite evaluation of λ,

for example over the inducing points can be expressed as $\boldsymbol{\lambda} = [\boldsymbol{\lambda}_{1,1}^\top, ..., \boldsymbol{\lambda}_{Q,1}^\top, ..., \boldsymbol{\lambda}_{1,R}^\top, ..., \boldsymbol{\lambda}_{Q,R}^\top]^\top \in \mathbb{R}^{QMR\times 1}$ with $\boldsymbol{\lambda}_{q,i} = [\lambda_{q,i}(\mathbf{z}_q^{(1)}), \ldots, \lambda_{q,i}(\mathbf{z}_q^{(M)})]^\top \in \mathbb{R}^{M\times 1}$. The specific terms of the augmented GP prior can be written as: $p(\mathbf{f}_j|\lambda) = \mathcal{N}(m_{f_j\lambda}(\mathbf{X}), \mathbf{0}) = \delta(\mathbf{f}_j - m_{f_j\lambda}(\mathbf{X}))$, where $m_{f_j\lambda}(\mathbf{X}) = [m_{f_j\lambda}(\mathbf{x}_1), ..., m_{f_j\lambda}(\mathbf{x}_N)]^\top \in \mathbb{R}^{N\times 1}$ is a vector built from:

$$m_{f_j\lambda}(\mathbf{x}) = \sum_{q=1}^{Q}\sum_{i=1}^{R}\int_{\mathcal{X}} G_{j,q,i}\left(\mathbf{x}-\mathbf{r}'\right)\lambda_{q,i}\left(\mathbf{r}'\right)\mathrm{d}\mathbf{r}',$$

and $p(\lambda|\boldsymbol{\lambda}) = \mathcal{N}(\lambda|k_{\lambda\boldsymbol{\lambda}}\mathbf{K}_{\boldsymbol{\lambda}\boldsymbol{\lambda}}^{-1}\boldsymbol{\lambda}_j, V_\lambda)$, is a distribution over the continuous inducing function λ, conditioned on $\boldsymbol{\lambda}$; $\mathbf{K}_{\boldsymbol{\lambda}\boldsymbol{\lambda}} \in \mathbb{R}^{QMR\times QMR}$ is a block-diagonal matrix with blocks $\mathbf{K}_{\boldsymbol{\lambda}_{q,i}\boldsymbol{\lambda}_{q,i}}$ with entries calculated with $k_{\lambda_{q,i}}(\mathbf{z},\mathbf{z}') := \mathrm{Cov}\left[\lambda_{q,i}(\mathbf{z}), \lambda_{q,i}\left(\mathbf{z}'\right)\right] = \int_{\mathcal{X}} T_q(\mathbf{z}-\mathbf{r})\int_{\mathcal{X}} T_q(\mathbf{z}'-\mathbf{r}')k_q(\mathbf{r},\mathbf{r}')\mathrm{d}\mathbf{r}\mathrm{d}\mathbf{r}'$ between all pairs of inducing points $\mathbf{Z}_q$;$V_\lambda = k_{\lambda\lambda} - k_{\lambda\boldsymbol{\lambda}}\mathbf{K}_{\boldsymbol{\lambda}\boldsymbol{\lambda}}^{-1}k_{\boldsymbol{\lambda}\lambda}$, where $k_{\lambda\lambda} = \mathrm{Cov}\left[\lambda,\lambda\right]$ is a continuos matrix covariance infinitely evaluated, and $k_{\lambda,\boldsymbol{\lambda}} = \mathrm{Cov}\left[\lambda,\boldsymbol{\lambda}\right]$ is a cross-covariance matrix with continuous rows and finite columns; and $p(\boldsymbol{\lambda}) = \mathcal{N}(\boldsymbol{\lambda}|\mathbf{0}, \mathbf{K}_{\boldsymbol{\lambda}\boldsymbol{\lambda}})$.

C. The Evidence Lower Bound

In a similar form to the ELBO derivation for the CCGP with CPM, here we approximate the true posterior $p(\mathbf{f},\lambda,\boldsymbol{\lambda}|\mathbf{y})$ with a variational distribution $q(\mathbf{f},\lambda,\boldsymbol{\lambda})$ for constructing the following ELBO [20]:

$$\mathcal{L} = \mathbb{E}_{q(\mathbf{f},\lambda,\boldsymbol{\lambda})}\Big[\log\frac{p(\mathbf{y}|\mathbf{f})p(\mathbf{f}|\lambda)p(\lambda|\boldsymbol{\lambda})p(\boldsymbol{\lambda})}{q(\mathbf{f},\lambda,\boldsymbol{\lambda})}\Big]. \tag{12}$$

We set a variational posterior as follows: $q(\mathbf{f},\lambda,\boldsymbol{\lambda}) = p(\mathbf{f}|\lambda)p(\lambda|\boldsymbol{\lambda})q(\boldsymbol{\lambda}) = \prod_{j=1}^{J} p(\mathbf{f}_j|\lambda)p(\lambda|\boldsymbol{\lambda})q(\boldsymbol{\lambda})$, where $q(\boldsymbol{\lambda}) = \mathcal{N}(\boldsymbol{\lambda}|\mathbf{m},\mathbf{V})$ with mean $\mathbf{m}\in\mathbb{R}^{QMR\times 1}$ and a block-diagonal covariance matrix $\mathbf{V}\in\mathbb{R}^{QMR\times QMR}$, which blocks are given by $\mathbf{V}_{q,i}\in\mathbb{R}^{M\times M}$. By replacing the posterior $q(\mathbf{f},\lambda,\boldsymbol{\lambda})$ in Eq. (12), we obtain a scalable objective:

$$\mathcal{L} = \sum_{n=1}^{N}\mathbb{E}_{q(\mathbf{f})}\big[g_n\big] - \mathbb{D}_{KL}\big(q(\boldsymbol{\lambda})||p(\boldsymbol{\lambda})\big), \tag{13}$$

where g_n is the LL function [12]. In contrast to the objective function in Eq. (5) for the CCGP with a CPM, the expectation for the LL in the equation above is calculated with respect to the marginal posterior, $q(\mathbf{f}) = \int\int\prod_{j=1}^{J} p(\mathbf{f}_j|\lambda)p(\lambda|\boldsymbol{\lambda})q(\boldsymbol{\lambda})d\boldsymbol{\lambda}d\lambda$. When solving for the integrals above, we obtain the following:

$$q(\mathbf{f}) := \mathcal{N}(\mathbf{f}|\tilde{\mathbf{m}}_{\mathbf{f}\boldsymbol{\lambda}}, \tilde{\mathbf{V}}_{\mathbf{f}\boldsymbol{\lambda}}), \tag{14}$$

where we have defined, $\tilde{\mathbf{m}}_{\mathbf{f}\boldsymbol{\lambda}} := \mathbf{A}_{\mathbf{f}\boldsymbol{\lambda}}\mathbf{m}$; $\mathbf{A}_{\mathbf{f}\boldsymbol{\lambda}} = \mathbf{K}_{\mathbf{f}\boldsymbol{\lambda}}\mathbf{K}_{\boldsymbol{\lambda}\boldsymbol{\lambda}}^{-1}$; and $\tilde{\mathbf{V}}_{\mathbf{f}\boldsymbol{\lambda}} := \mathbf{K}_{\mathbf{f}\mathbf{f}} + \mathbf{A}_{\mathbf{f}\boldsymbol{\lambda}}(\mathbf{V} - \mathbf{K}_{\boldsymbol{\lambda}\boldsymbol{\lambda}})\mathbf{A}_{\mathbf{f}\boldsymbol{\lambda}}^\top$; with $\mathbf{K}_{\mathbf{f}\boldsymbol{\lambda}} = [\mathbf{K}_{\mathbf{f}_1\boldsymbol{\lambda}}^\top, ..., \mathbf{K}_{\mathbf{f}_J\boldsymbol{\lambda}}^\top]^\top \in \mathbb{R}^{JN\times QMR}$ as a cross covariance matrix built with blocks $\mathbf{K}_{\mathbf{f}_j\boldsymbol{\lambda}} = [\mathbf{K}_{\mathbf{f}_j\boldsymbol{\lambda}_{1,1}}, ..., \mathbf{K}_{\mathbf{f}_j\boldsymbol{\lambda}_{Q,1}}, ..., \mathbf{K}_{\mathbf{f}_j\boldsymbol{\lambda}_{1,R}}, ..., \mathbf{K}_{\mathbf{f}_j\boldsymbol{\lambda}_{Q,R}}] \in \mathbb{R}^{N\times QMR}$, in which, each $\mathbf{K}_{\mathbf{f}_j\boldsymbol{\lambda}_{q,i}}\in\mathbb{R}^{N\times M}$ has entries computed with the covariance function, $\mathrm{Cov}\left[f_j(\mathbf{x}), \lambda_{q,i}\left(\mathbf{z}\right)\right]$, between the data observations $\mathbf{X}$ and the inducing points $\mathbf{Z}_q$; and $\mathbf{K}_{\mathbf{f}\mathbf{f}}$ is a covariance matrix built with evaluations of $\mathrm{Cov}[f_j\left(\mathbf{x}\right), f_{j'}\left(\mathbf{x}'\right)]$, between all pairs of data observation $\mathbf{X}$. In the following subsection, we detail the form of the covariance functions introduced above.

D. Covariance Functions for CCGP with VIKs

For our CCGP model with VIKs we follows the same EQ form of the kernel covariance functions. Given that this type of model also relies on the latent function $u_q \sim \mathcal{GP}(0, k_q(\cdot,\cdot))$ and a smoothing kernel $G_{j,q,i}(\cdot)$, we make use of exactly the same equations (8) for $k_q(\cdot,\cdot)$, and (9) for $G_{j,q,i}(\cdot)$. We additionally need to define the inducing kernel $T_q(\cdot)$ in Eq. (10), so we use the functional form in Eq. (7) to define: $T_q(\mathbf{x},\mathbf{x}') = W_q\mathcal{E}(\boldsymbol{\tau}|\mathbf{0},\mathbf{t}_q)$, where W_q is a weight an $\mathbf{t}_q$ is a diagonal matrix of length-scales. Similar to the case of CPM, we base on the multiplication identity between Gaussian distribution applied in [15]. Thus, we solve for $k_{\lambda_{q,i}}(\mathbf{z},\mathbf{z}') := \mathrm{Cov}\left[\lambda_{q,i}(\mathbf{z}), \lambda_{q,i}\left(\mathbf{z}'\right)\right] = \int_{\mathcal{X}} T_q(\mathbf{z}-\mathbf{r})\int_{\mathcal{X}} T_q(\mathbf{z}'-\mathbf{r}')k_q(\mathbf{r},\mathbf{r}')\mathrm{d}\mathbf{r}\mathrm{d}\mathbf{r}'$ and $\mathrm{Cov}\left[f_j(\mathbf{x}), \lambda_{q,i}\left(\mathbf{z}\right)\right] = \int_{\mathcal{X}} G_{j,q,i}(\mathbf{x}-\mathbf{r}')\int_{\mathcal{X}} T_q(\mathbf{z}-\mathbf{r})k_q(\mathbf{r}',\mathbf{r})\mathrm{d}\mathbf{r}'\mathrm{d}\mathbf{r}$, and arrive to: $\mathrm{Cov}\left[\lambda_{q,i}(\mathbf{x}), \lambda_{q,i}\left(\mathbf{x}'\right)\right] = W_q^2\mathcal{E}(\boldsymbol{\tau}|\mathbf{0}, \mathbf{t}_q + \mathbf{t}_q + \mathbf{L}_q)$, $\mathrm{Cov}\left[f_j(\mathbf{x}), \lambda_{q,i}\left(\mathbf{x}'\right)\right] = S_{j,q,i}W_q\mathcal{E}(\boldsymbol{\tau}|\mathbf{0}, \boldsymbol{\kappa}_j + \mathbf{t}_q + \mathbf{t}_q + \mathbf{L}_q)$. Also, when solving for the covariance function:

$$\mathrm{Cov}[f_j\left(\mathbf{x}\right), f_{j'}\left(\mathbf{x}'\right)] = \sum_{q=1}^{Q}\sum_{i=1}^{R}\int_{\mathcal{X}} G_{j,q,i}\left(\mathbf{x}-\mathbf{v}\right) \times \int_{\mathcal{X}} G_{j',q,i}\left(\mathbf{x}'-\mathbf{v}'\right)k_{\lambda_{q,i}}(\mathbf{v},\mathbf{v}')\mathrm{d}\mathbf{v}\mathrm{d}\mathbf{v}',$$

we end up with: $\mathrm{Cov}\left[f_j\left(\mathbf{x}\right), f_{j'}\left(\mathbf{x}'\right)\right] = \sum_{q=1}^{Q} S_{j,q,i}S_{j',q,i}W_qW_q\mathcal{E}(\boldsymbol{\tau}|\mathbf{0}, \mathbf{T}_{j,j',q})$, where $\mathbf{T}_{j,j',q}$ represents a diagonal matrix of length-scales, $\mathbf{T}_{j,j',q} = \boldsymbol{\kappa}_j + \boldsymbol{\kappa}_{j'} + \mathbf{t}_q + \mathbf{t}_q + \mathbf{L}_q$.

E. Making Predictions with CCGP based on VIKs

In a similar way to section III-E, we compute $p\left(\mathbf{y}_* \mid \mathbf{y}\right) \approx \int p\left(\mathbf{y}_* \mid \mathbf{f}_*\right) q\left(\mathbf{f}_*\right) d\mathbf{f}_*$. Notice that the distribution $q\left(\mathbf{f}_*\right)$ now involves computations associated to $\boldsymbol{\lambda}$ instead of $\mathbf{u}$. Therefore, for a new set of inputs $\mathbf{X}_*$, we have to build $\mathbf{K}_{\mathbf{f}_*\boldsymbol{\lambda}}$ and $\mathbf{K}_{\mathbf{f}_*\mathbf{f}_*}$ as per Eq. (14).

V. Zero-Inflated Poisson distribution

Since we aim to model non-negative values that represent counts and also tackle the problems associated to zero-inflation data [1], [28], here we rely on the ZIP distribution for targeting such issues [4]. The ZIP probability distribution can be expressed as follows:

$$p(y_n|\phi_n,\rho_n) = \mathbf{1}_{y_n}\phi_n + (1-\phi_n)\frac{\exp(-\rho_n)\rho_n^{y_n}}{y_n!}, \tag{15}$$

where $\phi_n\in[0,1]$ is a parameter that represents a probability for the values at zero [29], $\rho_n > 0$ is the Poisson rate parameter and $\mathbf{1}_{y_n} := \mathbf{1}(y_n)$ is an indicator function defined as follows: $\mathbf{1}(y_n) = 1$ if $y_n = 0$, or $\mathbf{1}(y_n) = 0$ if $y_n \neq 0$. Following the notation in Eq. (1), here the distribution's parameters are associated as $\phi_n = \psi_1\left(\mathbf{x}_n\right)$ and $\rho_n = \psi_2\left(\mathbf{x}_n\right)$, where $\psi_1\left(\mathbf{x}_n\right) = \sigma(f_1(\mathbf{x}_n))$ and $\psi_2\left(\mathbf{x}_n\right) = \exp(f_2(\mathbf{x}_n))$; $\sigma(f_1(\mathbf{x}_n)) = 1/(1+\exp(-f_1(\mathbf{x}_n)))$ is a sigmoid function. With the definitions above we can plug Eq. (15) in the Log

Likelihood function, $g_n = \log p\left(y_n|\psi_1\left(\mathbf{x}_n\right), ..., \psi_J\left(\mathbf{x}_n\right)\right)$, of equations (5) and (13) as follows: g_n

$$= \log\left(\mathbf{1}_{y_n}\phi_n \exp(\rho_n) + (1-\phi_n)\right) + \log\left(\frac{\exp(-\rho_n)\rho_n^{y_n}}{y_n!}\right).$$

It is worth noticing that the equation above is exactly the same for both CCGP models based on either the CPM (Eq. (5)) or VIK (Eq. (13)); They just differ by construction in the way of generating the LPFs, but not in the form of the likelihood function. In the experiments section, we will compare the performance of our proposed CCGP methods, the CPM-based and VIK-based in conjunction with either a ZI-Poisson likelihood or a Poisson likelihood.

VI. Optimisation

In order to fit our proposed models, we make use of a recent algorithm that optimises: each variational parameter $\mathbf{m}$ and $\mathbf{V}$; each set of inducing points $\mathbf{Z}$; and all the kernels' hyper-parameters, $\mathbf{H} = \{\{\boldsymbol{\kappa}_j\}_{j=1}^{J}, \{\mathbf{L}_q\}_{q=1}^{Q}, \{\mathbf{t}_q\}_{q=1}^{Q}\}$, through a natural gradient scheme [30]. The algorithm consists on building an alternative variational optimisation bound of the form, $\tilde{\mathcal{F}} = \mathbb{E}_{q(\boldsymbol{\theta})}[-\mathcal{L}] + \mathbb{D}_{KL}(q(\boldsymbol{\theta})||p(\boldsymbol{\theta}))$, where $\mathcal{L}$ is any of the objective functions for either the CPM or VIK in Eq. (5) or (13) respectively; $q(\boldsymbol{\theta})$ represents a free parametrised exploratory distribution over the set of all parameters to optimise i.e., $\boldsymbol{\theta} = \{\mathbf{m}, \mathbf{V}, \mathbf{Z}, \mathbf{H}\}$; and $p(\boldsymbol{\theta})$ is a penalisation distribution. The main idea of the method relies on adjusting the distribution $q(\boldsymbol{\theta})$ during the optimisation by trading-off between minimising the expectation $\mathbb{E}_{q(\boldsymbol{\theta})}[-\mathcal{L}]$ and reducing the divergence $\mathbb{D}_{KL}(q(\boldsymbol{\theta})||p(\boldsymbol{\theta}))$. During inference, such a Kullback-Leibler divergence helps to gain additional exploration of the space of solutions for optimising the exploratory distribution. At the end of the inference process, $q(\boldsymbol{\theta})$'s mean becomes the best solution for the set of parameters of the model. We chose this optimisation method due to its ability for avoiding poor local optima solutions and having closed-form update equations for fitting the parameters.

VII. EXPERIMENTS

In this section, we make a quantitative and qualitative analysis of the predictions obtained by the there types of CCGP models based on: a LMC (implemented in [12]), our CPM proposed in Eq. (5) and also our VIK introduced in Eq. (13). As explained at section V, we implement a ZI-Poisson likelihood and compare its performance with a Poisson likelihood for modelling the citizens mobility in Guangzhou city. We run two types of experiments: the first corresponds to building a model per each day of the month; the second to building a model per each day of the week. [1]

A. Dataset of Guangzhou City

The dataset used to model the citizens' mobility in the region of Guangzhou was built from recordings of mobile phone GPS locations. In a nutshell, the users of a Guangzhou's mobile phone network share their longitude and latitude coordinates that are consequently preprocess through a counting algorithm. Such an algorithm consists on counting the citizens that coincide in a delta area of Guangzhou; i.e., the main region of Guangzhou is divided in a grid of 201×201, where each square (or delta area) of the grid contains a total number of citizens. The counting is performed every hour of the day, this during 31 days: from March 1 to 31 of 2019. The total number of data observations per day is $N = 201 \times 201 \times 24 = 969624$.

B. Model Training

Given that the models derived in Eq. (5) and Eq. (13) allow stochastic variational inference, we use a random mini-batching of 400 samples per iteration during training. We selected through cross-validation a number of latent functions $Q = 3$ and inducing points $M = 200$. It is worth noticing that the expectations of the Log Likelihood in such equations (5) and (13) cannot be computed in closed-form, so we opt for using the Gauss-Hermite quadrature approach [10], [31]. Also, it is important to highlight that there is not need to compute the full covariances $\mathbf{K}_{\mathbf{f}_j\mathbf{f}_j}$ in Eq. (6) for the CPM-based model or in Eq. (14) for the VIK-based model, but only the diagonal values randomly selected as per the mini-batching at each optimisation iteration. We carry out optimisation of each variational parameter $\mathbf{m}$ and $\mathbf{V}$, each set of inducing points $\mathbf{Z}_q$ and all kernels' hyper-parameters through the natural gradient algorithm described in section VI [30].

C. Quantitative Results: Models along the Month

The first experiment consists on building 31 CCGP models, one model per day during all the month of March. We use a dataset random splitting of 90% and 10% for training and testing respectively. In order to measure the uncertainty quantification capability of the models, we report the negative log predictive density (NLPD) error over the test set; such a NLPD metric takes into account the predictions' uncertainty [32]. Figure 1 shows the performance of the CCGP models based on VIK, CPM and LMC, when using a Poisson distribution (top figure) and a ZI-Poisson distribution (bottom figure). Take into account that low NLPD values mean better performance.

We can notice from Fig. 1 that the models with a Poisson likelihood presented metrics roughly within the interval $(2.2, 2.7)$, whilst the models with a ZI-Poisson likelihood obtained metrics approximately among the interval $(0.59, 1.02)$. Thereby, we can say that in general the models relying on a ZI-Poisson likelihood outperformed the ones based on a Poisson likelihood by achieving lower NLPD metrics. For the Poisson likelihood, the VIK model accomplished the lowest NLPD for 22 days of the month in comparison to the CPM and LMC; the CPM presented a better performance than VIK and LMC for six days; and the LMC only presented the lowest NLPD in the days: 13, 16 and 30. For the ZI-Poisson likelihood, the VIK reached better NLPD values for 13 days; the CPM obtained the lowest metrics in 18 times; and the LMC did not present a better performance at any day in comparison to the other CCGP models. Table I shows a summary of the main statistics

[1] The code with the proposed models in publicly available in the repository: https://github.com/juanjogg1987/CorrelatedChainedGPs_ConvolutionProcesses

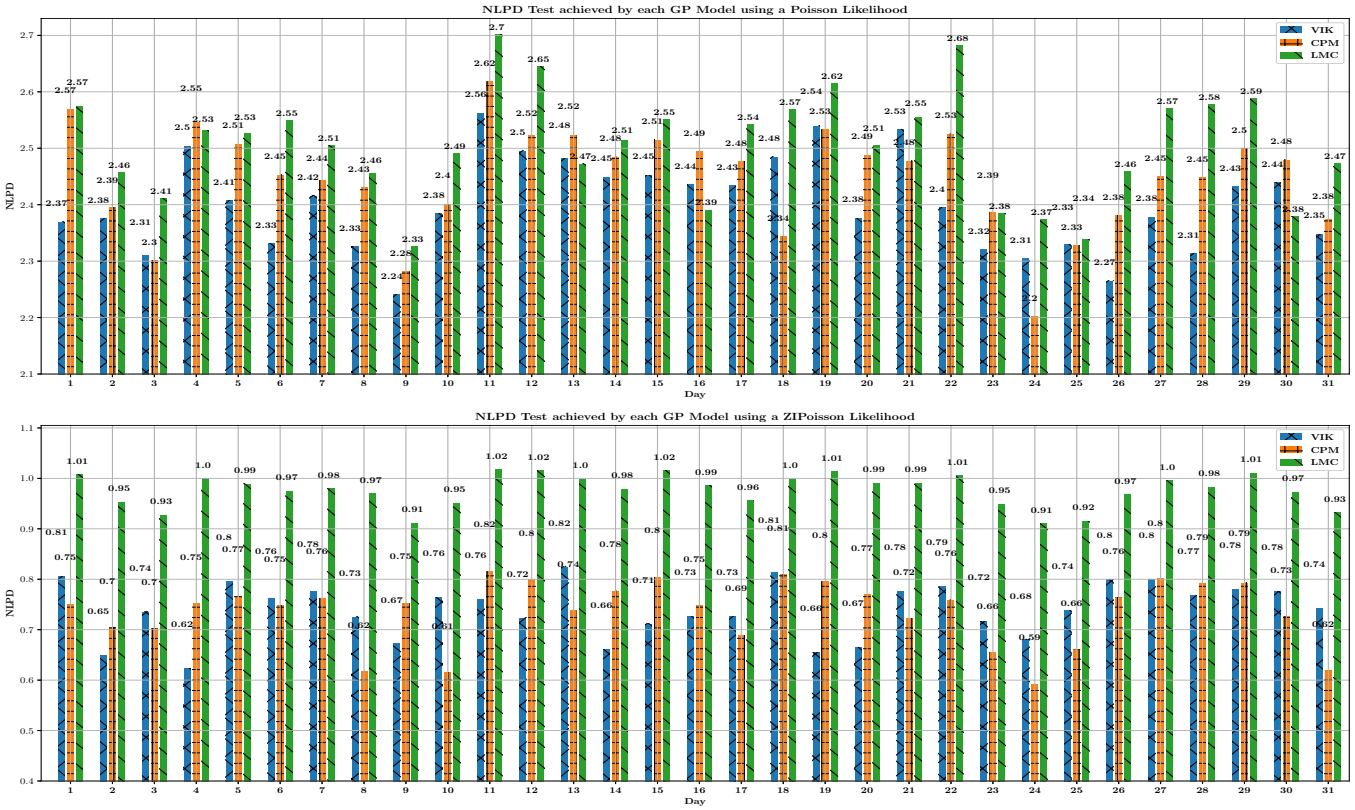


Fig. 1. NLPD-Test Performance along the Month for the CCGP models based on VIK, CPM and LMC. Top figure: Poisson likelihood. Bottom figure: ZI-Poisson likelihood. For each day there are three bars associated to the GP priors: left bar, VIK with pattern inscription "x"; middle bar, CPM with pattern "+"; and right bar, LMC with pattern "\". Low NLPD values mean better performance.

TABLE I
SUMMARY OF STATISTICS OF NLPD-TEST PERFORMANCE ALONG THE MONTH FOR THE CCGP MODELS BASED ON POISSON AND ZI-POISSON LIKELIHOODS USING THREE TYPES OF GP PRIORS.

	VIK		CPM		LMC	
	Avg ± Std	**Med**	**Avg ± Std**	**Med**	**Avg ± Std**	**Med**
Poisson	2.401± 0.081	**2.395**	2.448 ± 0.090	2.476	2.507 ± 0.094	2.515
ZIP	0.740 ± 0.053	**0.742**	0.736 ± 0.063	0.752	0.976 ± 0.032	0.982

obtained by the models along the Month. We can see from such a Table that the CCGP model based on VIK reached a better trending with the lowest median for both types of likelihoods; also it tended to present smaller standard deviations than the other methods. The CPM showed a slightly lower mean than the VIK when using a ZI-Poisson, but with a higher standard deviation. The LMC presented similar results to the CPM for the case of a Poisson likelihood; nevertheless, in comparison to VIK and CPM, its performance was poor when modelling with a ZI-Poisson likelihood. The results show that the use of a ZI-Poisson likelihood considerably improved the performance of the prediction capabilities of our CCGP models in the context of the zero-inflated data from Guangzhou city. Regarding the types of GP priors, the VIK and CPM showed to outperform the LMC by allowing better NLPD metrics, i.e., a better quantification of the uncertainty.

D. Quantitative Results: Models along the Week

For the second experiment, we trained seven CCGP models, one per day of the week, i.e., models for Monday, Tuesday, Wednesday, Thursday, Friday, Saturday and Sunday. In contrast to the first experiment, here we selected the observations related to the Monday March 4 to Sunday March 10 for training data; whilst the remaining days were used for testing: Mondays (March 11, 18, 25), Tuesday (March 12, 19, 26), Wednesday (March 13, 20, 27), Thursday (March 14, 21, 28), Friday (March 15, 22, 29), Saturday (March 16, 23, 30) and Sunday (March 17, 24, 31). For instance, we trained a model for Monday using data from March 4 and tested it over the remaining Mondays March 11, 18 and 25. Figure 2 shows the NLPD error obtained by the different GP models in combination with both types of likelihoods, Poisson and ZI-Poisson. From Figure 2 we can observe that the ranges of NLPD metrics accomplished by the CCGP models when using a ZI-Poisson likelihood were lower in comparison to the Poisson likelihood; ZIP metrics were within the range $(0.52, 0.78)$ and Poisson metrics are within $(1.72, 1.97)$. Comparing these latter results with the ones reached in the previous subsection of Models along the Month, we can regard that the CCGP models along the week present a better performance. For the Poisson likelihood, Figure 2 shows that the model based on VIK obtained the lowest NLPD values for all the days of the week, followed by the CPM and LMC. For the ZI-Poisson likelihood, Figure 2 shows that the model VIK-based reached the lowest NLPD values for Monday and Saturday in comparison to the other methods; whilst the model CPM-based attained the lowest NLPD values for Tuesday, Wednesday,

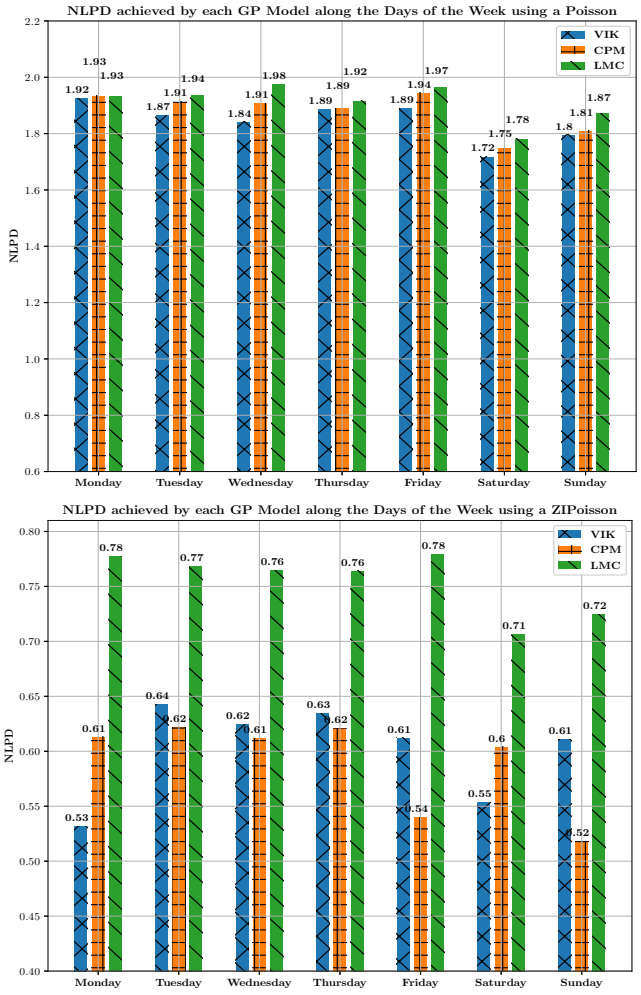


Fig. 2. NLPD-Test Performance along the Week for the CCGP models based on VIK, CPM and LMC. Top figure: Poisson likelihood. Bottom figure: ZIP likelihood. For each day there are three bars associated to the GP priors: left bar, VIK with pattern inscription "x"; middle bar, CPM with pattern "+"; and right bar, LMC with pattern "\". Low NLPD values mean better performance.

Thursday, Friday and Sunday; the model LMC-based did not present a better performance than the other methods on any of the days. Though, for the ZI-Poisson likelihood, the CPM presented better metrics on more days than the VIK, the summary of statistic in Table II shows that in general the VIK performs similar to the CPM for such a likelihood.

Table II allows us to see that generally the CCGP model based on VIK obtained better NLPD metrics in comparison to the other methods. The VIK performed quite similar to the CPM in the context of a ZIP likelihood; indeed, VIK and CPM presented a difference in the mean of just 0.011 for the ZIP likelihood, with equal standard deviations and equal medians. Regardless of the type of likelihood, both CPM and VIK models outperformed the LMC model.

E. Qualitative Results

Since our data from Guangzhou city presents zero-inflation issues, we aim to observe the effect in the predictions when using the Poisson and ZI-Poisson likelihoods to deal with said problem. Also, in order to visualise the qualitative traits of

TABLE II
SUMMARY OF STATISTICS OF NLPD-TEST PERFORMANCE ALONG THE WEEK FOR THE CCGP MODELS BASED ON POISSON AND ZI-POISSON LIKELIHOODS USING THREE TYPES OF GP PRIORS.

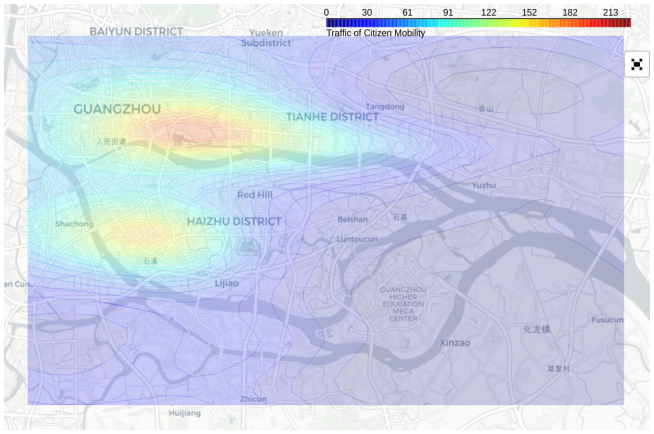
	VIK		**CPM**		**LMC**	
	Avg ± Std	**Med**	**Avg ± Std**	**Med**	**Avg ± Std**	**Med**
Poisson	1.847 ± 0.065	**1.865**	1.878 ± 0.067	1.907	1.911 ± 0.062	1.932
ZIP	0.601 ± 0.039	**0.612**	0.590 ± 0.039	0.612	0.755 ± 0.026	0.764

the model that showed the highest capabilities to generalise, we selected the CCGP model based on VIK (CCGP-VIK), particularly we chose the model for Saturday, March 9 (from Figure 2) given that it presented a relevant NLPD performance for both Poisson and ZI-Poisson likelihoods.

Figure 3(b) shows the mean prediction of the CCGP-VIK with Poisson likelihood and Figure 3(d) the mean prediction of the CCGP-VIK with ZI-Poisson likelihood. Figures 3(a) and 3(c) are a heatmap of the real test data of the citizens mobility on Saturday, March 16 at 11:00 am; both figures are exactly the same, but displayed twice to ease the comparison to our models' predictions provided in Figures 3(b) and 3(d). Also, to ease the description of the predictions we will refer to the names that appear in the maps as key locations, for instance the names: GUANGZHOU, TIANHE DISTRICT, Red Hill, Lijiao, Shachong, Xinzao, etc.

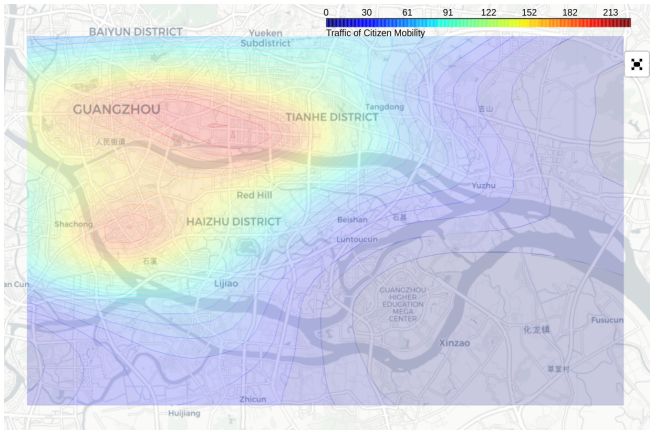
It can be seen from Figures 3(b) and Figure 3(d) that both CCGP-VIK models with Poisson and ZI-Poisson focus on the high concentrations of citizens in the city centre, that is the region between GUANGZHOU, TIANHE DISTRICT and Red Hill. Also those models focus on the region with high numbers of citizens located among Shachong, HAIZHU DISTRICT and Lijiao; that is a central-west region that gathers different Metro-stations like: Jiangnanxi station, Huadiwan station, Xilang station and Jushu station. For the case of the Poisson likelihood, we can notice from Figure 3(b) that the predictions of high concentrations of citizens are underestimated in comparison to the test data in Figure 3(a); whilst for the case of the ZI-Poisson likelihood, we observe that the density of citizens looks more akin to the test data in Figure 3(c). With respect to the regions of the city that present many zero value observations like the north-east and south-east quadrants, we can notice that both the CCGP-VIK models with Poisson and ZI-Poisson predict very low concentrations of citizens in those regions. Although, specifically the model with Poisson likelihood concentrates on predicting massive densities of zero values in the north-east quadrant of the map that extend until Tangdong region; in contrast, the model with ZI-Poisson remains a bit conservative not presenting as huge accumulations of zero values as the Poisson distribution, and allowing to predict moderate concentrations of people around Tangdong. Likewise, for the region in the south-west quadrant below Lijiao, the model with Poisson likelihood focuses on predicting very low concentrations of citizens, but the model with ZI-Poisson predicts moderate congregations of citizens that vanish from Lijiao towards Huijiang and Zhicun. Both ZIP and Poisson models neglect the concentrations of citizens in the region below Luntoucun and to the left hand side of GUANGZHOU HIGHER EDUCATION MEGA CENTER.

(a) Test Data Heatmap: Saturday, March 16 at 11:00 am



(b) Prediction at 11:00 am using CCGP-VIK with Poisson

(c) Test Data Heatmap: Saturday, March 16 at 11:00 am



(d) Prediction at 11:00 am using CCGP-VIK with ZI-Poisson

Fig. 3. Qualitative performance of the model CCGP-VIK (trained with data from Saturday, March 9) in comparison to real test data. To the left hand side, Figures 3(a) and 3(c) are a heatmap of the real test data of the citizens mobility on Saturday, March 16 at 11:00 am; both figures are the same, but displayed twice to ease the comparison with the predictions to the right hand side. Figure 3(b) shows the mean prediction of the CCGP-VIK with Poisson likelihood. Figure 3(d) presents the mean prediction of the CCGP-VIK with ZI-Poisson likelihood. The color bar associates the number of citizens in the map area.

The main quality of the model CCGP-VIK with ZIP consists on appropriately trading-off between making predictions in the regions with high concentration of zeros without underestimating the regions with substantial presence of citizens. Conversely, the zero-inflation inherent in Guangzhou data hinders the model based on a Poisson likelihood to adequately forecast in the regions with major presence of citizens.

VIII. Discussion and Conclusion

Through the different types of experiments we noticed that the use of a ZI-Poisson distribution significantly improved the performance for modelling the citizens' mobility in Guangzhou city, in comparison to a Poisson distribution. For the case of modelling with a Poisson likelihood, though the ranking of best performances usually showed the VIK at first, followed by CPM and LMC, we realised the mean NLPD metrics are very close between all the GP methods (with a difference not higher than 0.106) as shown in Table I. We believe those NLPD metrics are close to each other due to the few parameters present in the Poisson distribution, which limit the capabilities of the different GP models for achieving a higher predictive performance. On the other hand, when modelling with the ZI-Poisson, the CCGP models based on VIK and CPM presented a distinguished difference with the LMC. Such a difference can be attributed to the fact of having additional hyper-parameters that allow higher modelling flexibilities than the LMC, which only depends on a set of linear combination coefficients and matrices of length-scales $\mathbf{L}_q$ related to the latent functions $u_q(\cdot)$ [22][11]. Such additional hyper-parameters can be identified, for instance: from the CPM, in all weights $S_{j,q,i}$ and the matrices of length-scales $\boldsymbol{\kappa}_j$ associated to the smoothing kernels, and the matrices of length-scales $\mathbf{L}_q$ for the latent functions $u_q(\cdot)$ (see section III-D); and apart from the latter parameters present in the CPM, the VIKs model additionally presents weights W_q and the matrices of length-scales $\mathbf{t}_q$ associated to the inducing kernels (see section IV-D). Regarding the two types of experiments we carried out for modelling either along the month or along the week, the results showed a better performance in the models along the week. We associate the high performance reached by such models with a probable high correlation between the training and testing data; the mobility patterns of citizens for instance on Monday March 4 (training data) can be very similar to the remaining Mondays 11, 18 and 25 (testing data), a fact likely to happen also for the other days.

In this work we have modelled the citizens mobility in the Chinese city of Guangzhou by means of three types of CCGP models (based on an LMC, CPM or VIKs) with Poisson and ZI-Poisson likelihoods. We showed that all types of CCGP models in conjunction with a ZIP likelihood allow to overcome the issues associated to zero-inflated data, outperforming the predictive capabilities of such CCGP models when based on a Poisson likelihood. We derived a stochastic variational inference framework that grants the use of a CCGP model with CPM or VIKs in the context of a large number of data observations. As a future work, it might be worth to explore the behaviour of other types of likelihoods like the ZINB, or the Tweedie from the exponential dispersion family. The latter can be particularly challenging given that its probability distribution needs to be evaluated using a series expansion due to not having an analytical solution. On the other hand, we believe our models can also be used in the context of Multi-Output GPs, for instance: with a broader database information that not only contained citizens mobility, but in which we could discriminate the type of vehicles used for mobility; it is feasible to implement our GP models to exploit correlations that include information between types of transport vehicles. Likewise, we could explore the application of Multi-Output GPs for data imputation, i.e., predicting information in regions of the city where data is sensible to be lost due to failures in the mobile phone network that carries out the data collection.

ACKNOWLEDGMENT

The authors would like to thank Innovate UK for funding under the project 104316. JJG is being funded by a scholarship from the Dept. of Comp. Science, University of Sheffield. MAA has been financed by the EPSRC Research Projects EP/R034303/1 and EP/T00343X/1.

REFERENCES

[1] A. Zuur, E. N. Ieno, N. Walker, A. A. Saveiliev, and G. M. Smith, *Mixed Effects Models and Extensions in Ecology with R*. Springer, 2009.
[2] K. P. Murphy, *Machine Learning: A Probabilistic Perspective*. MIT Press, 2013.
[3] J. Long and J. Freese, *Regression models for categorical dependent variables using stata*, 3rd ed. Stata Press, 2014.
[4] E. S. Roemmele, "A flexible zero-inflated Poisson regression model," *PhD Thesis, University of Kentucky*, 2019.
[5] G. K. Smyth and B. Jørgensen, "Fitting Tweedie's compound Poisson model to insurance claims data: Dispersion modelling," *ASTIN Bulletin*, vol. 32, no. 1, pp. 143–157, 2002.
[6] W. H. Bonat, B. Jørgensen, C. C. Kokonendji, J. Hinde, and C. G. B. Demétrio, "Extended Poisson–Tweedie: Properties and regression models for count data," *Stat. Modelling*, vol. 18, no. 1, pp. 24–49, 2018.
[7] O. Kahilakoski, "Bayesian regression analysis of sickness absence," *Msc Thesis, Aalto University*, 2011.
[8] P. Hegde, M. Heinonen, and S. Kaski, "Variational zero-inflated Gaussian processes with sparse kernels," in *UAI*, vol. 1, 2018, pp. 361–371.
[9] N. BinTayyash, S. Georgaka, S. John, S. Ahmed, A. Boukouvalas, J. Hensman, and M. Rattray, "Non-parametric modelling of temporal and spatial counts data from RNA-seq experiments," *bioRxiv*, 2020.
[10] A. Saul, J. Hensman, A. Vehtai, and N. D. Lawrence, "Chained Gaussian processes," *AISTATS*, 2016.
[11] M. A. Álvarez, L. Rosasco, and N. D. Lawrence, "Kernels for vector-valued functions: A review," *Found. Trends Mach. Learn.*, vol. 4, no. 3, pp. 195–266, Mar. 2012.
[12] P. Moreno-Muñoz, A. Artés-Rodríguez, and M. A. Álvarez, "Heterogeneous multi-output Gaussian process prediction," in *NeurIPS*, 2018, pp. 6712–6721.
[13] P. Boyle and M. Frean, "Dependent Gaussian processes," in *NIPS*. MIT Press, 2005, pp. 217–224.
[14] M. A. Álvarez, D. Luengo, M. K. Titsias, and N. D. Lawrence, "Efficient multioutput Gaussian processes through variational inducing kernels," in *AISTATS*, 2010.
[15] M. A. Álvarez and N. D. Lawrence, "Computationally efficient convolved multiple output Gaussian processes," *JMLR*, vol. 12, p. 1459–1500, 2011.
[16] A. Saul, "Gaussian process based approaches for survival analysis," *PhD Thesis, University of Sheffield*, 2016.
[17] H. Rodriguez-Deniz, E. Jenelius, and M. Villani, "Urban network travel time prediction via online multi-output Gaussian process regression," in *20th IEEE-ITSC*. IEEE, 2017, pp. 1–6.
[18] F. Rodrigues, K. Henrickson, and F. Pereira, "Multi-output Gaussian processes for crowdsourced traffic data imputation," *IEEE Trans. on ITS*, vol. 20, no. 2, pp. 594 – 603, 2019.
[19] M. D. Hoffman, D. M. Blei, C. Wang, and J. Paisley, "Stochastic variational inference," *JMLR*, vol. 14, no. 1, pp. 1303–1347, May 2013.
[20] D. M. Blei, A. Kucukelbir, and J. D. McAuliffe, "Variational inference: A review for statisticians," *Journal of the American Statistical Association*, vol. 112, no. 518, pp. 859–877, 2017.
[21] C. E. Rasmussen, *Gaussian processes for machine learning*. MIT Press, 2006.
[22] A. G. Journel and C. J. Huijbregts, "Mining geostatistics," *Academic Press, London*, 1979.
[23] D. Higdon, "Space and space-time modeling using process convolutions." Springer London, 2002, pp. 37–56.
[24] E. Snelson and Z. Ghahramani, "Sparse Gaussian processes using pseudo-inputs," in *NIPS*, 2006, pp. 1257–1264.
[25] M. K. Titsias, "Variational learning of inducing variables in sparse Gaussian processes," in *AISTATS*, 2009.
[26] M. A. Álvarez and N. D. Lawrence, "Sparse convolved Gaussian processes for multi-output regression," in *NIPS*, 2009, pp. 57–64.
[27] J. Hensman, N. Fusi, and N. D. Lawrence, "Gaussian processes for big data," in *UAI*, 2013.
[28] T. M. Lukusa, S.-M. Lee, and C.-S. Li, "Review of zero-inflated models with missing data," *CR in Biostatistics*, vol. 7, no. 1, pp. 1–12, 2017.
[29] S. Beckett, J. Jee, T. Ncube, S. Pompilus, Q. Washington, A. Singh, and N. Pal, "Zero-inflated poisson (ZIP) distribution: parameter estimation and applications to model data from natural calamities," *Involve: A Journal of Mathematics*, vol. 7, no. 6, pp. 751–767, 2014.
[30] J.-J. Giraldo and M. A. Álvarez, "A fully natural gradient scheme for improving inference of the heterogeneous multi-output Gaussian process model," *arXiv preprint arXiv:1911.10225v2*, 2019.
[31] S. Jin and B. Andersson, "A note on the accuracy of adaptive Gauss–Hermite quadrature," *Biometrika*, vol. 107, no. 3, pp. 737–744, 2020.
[32] J. Quiñonero-Candela, C. E. Rasmussen, F. Sinz, O. Bousquet, and B. Schölkopf, "Evaluating predictive uncertainty challenge." Springer Berlin Heidelberg, 2006, pp. 1–27.

Juan-José Giraldo received a degree in Electronics Engineering (B. Eng.) with Honours, from Universidad del Quindío, Colombia in 2009, a master degree in Electrical Engineering (M. Eng.) from Universidad Tecnológica de Pereira, Colombia in 2015. Currently, Mr. Giraldo is a Ph.D student in Computer Science at the University of Sheffield, UK.

Jie Zhang received MEng and PhD in Industrial Automation from East China University of Science and Technology in 1992 and 1995. Dr. Zhang studied/worked with Imperial College London, Oxford University and University of Bedfordshire, becoming a Lecturer, Reader and Professor in 2002, 2005 and 2006 respectively. Dr. Zhang is co-founder and Board Director of RANPLAN, which is listed on NASDAQ First North stock exchange and produces a suite of world leading in-building DAS, indoor-outdoor small cell/HetNet network design and optimisation tools; also the founder of Cambridge AI+ Ltd. From January, 2011, Dr. Zhang held the Chair in Wireless Systems at the Department of Electronic and Electrical Engineering at The University of Sheffield, UK.

Mauricio A. Álvarez received a degree in Electronics Engineering (B. Eng.) with Honours, from Universidad Nacional de Colombia in 2004, a master degree in Electrical Engineering (M. Eng.) from Universidad Tecnológica de Pereira, Colombia in 2006, and a Ph.D. degree in Computer Science from The University of Manchester, UK, in 2011. After finishing his Ph.D., Dr. Álvarez joined the Dept. of Electrical Engineering at Universidad Tecnológica de Pereira, Colombia, where he was appointed as an Associate Professor until Dec 2016. From January, 2017, Dr. Álvarez joined the Department of Computer Science at The University of Sheffield, UK, where he is now a Senior Lecturer in Machine Learning.