Stochastic Representations with Gaussian Processes and Geometry

Martin Jørgensen

Kongens Lyngby 2020

Technical University of Denmark
Department of Applied Mathematics and Computer Science
Richard Petersens Plads, building 324,
2800 Kongens Lyngby, Denmark
Phone +45 4525 3031
compute@compute.dtu.dk
www.compute.dtu.dk

Summary (English)

This thesis consists of 4 independent pieces of work and each of these have a dedicated chapter in the manuscript. The first chapter investigates contemporary methodologies for estimating predictive variance networks in regression neural networks. The second chapter goes beyond regression task, and studies Gaussian processes to present a Bayesian non-parametric way of inferring stochastic differential equations for both regression and continuous-time dynamical modelling. The third chapter unifies theory of geometry and Gaussian processes to present a latent variable model that respects both the distances and the topology of unlabelled data. The fourth, and last, chapter shortly reviews current methodologies for bivariate causal invariance and propose an algorithm using a non-parametric estimator robust towards a causal invariant: changes in the marginal distributions.

Summary (Danish)

Denne afhandling består af 4 uafhængige forskningsprojekter og hvert projekt har et tilhørende kapitel i afhandlingen. Det første kapitel omhandler metoder til at kvantificere varians i neurale netværk til regression. Kapitel 2 betragter mere end regressionsopgaver og studerer Gaussiske processer for at præsentere en Bayesiansk ikke-parametrisk måde at deducere i stokastiske differentialligninger. Disse kan bruges til både regression og dynamisk modellering i kontinuert tid. Det tredje kapitel benytter teori fra geometri og Gaussiske processer til at præsentere en latent variabel model som respekterer både afstande og topologi af ikke-annoteret data. Fjerde, og sidste, kapitel omhandler bivariat kausal inferens og beskriver en algoritme som bruger ikke-parametriske estimatorer til at undersøge en kausal invariant: ændringer i marginale fordelinger.

Preface

The present thesis was written at the Section for Cognitive Systems, DTU Compute, Technical University of Denmark in fulfillment of the requirements for acquiring a PhD degree at the Technical University of Denmark.

Professor Søren Hauberg and professor Lars Kai Hansen supervised the project. The project was funded by VILLUM FONDEN (15334).

The project was carried out from September 2017 to August 2020 at the Technical University of Denmark, with an exception of four months external stay at Imperial College London. The supervision at this time was conducted by Marc Peter Deisenroth (now at University College London).

The work of this thesis amounts to four papers, and is presented with a thorough introduction to each. All papers are appended in this thesis.

Lyngby, 31-08-2020

Martin Jørgensen

Acknowledgements

There is a series of people that I am grateful for assisting me while writing this thesis. Firstly, I owe a million thanks to my advisor Søren Hauberg. Thank you for giving me the opportunity to begin, and thank you for your invaluable guidance and expertise; without it, I could never have reached the end. I also am grateful for everyone at the Section of Cognitive Systems for creating a stimulating and pleasant environment.

I also owe thanks to Marc Deisenroth, and everyone in his group at Imperial College London, for the hospitality and supervision I was given during my stay in London.

A big thank you to my family, my parents, for whom I would never have made it to this point. Their support was and is priceless and I hope they know how much it means.

Lastly, a thank you to my girlfriend, Nanna; your marvellous personality and silly nature have made every day a good day, and without that, I could not have made it through this journey. There is so much more I need to thank her for. I love you, Honeybear.

List of Publications

Published

Reliable training and estimation of variance networks
Nicki Skafte Detlefsen*, Martin Jørgensen* and Søren Hauberg
Advances in Neural Information Processing Systems, (NeurIPS) 2019
(* Equal contribution)

Stochastic Differential Equations with Variational Wishart Diffusions
Martin Jørgensen, Marc Peter Deisenroth and Hugh Salimbeni
International Conference on Machine Learning, (ICML) 2020

Preprints

Isometric Gaussian Process Latent Variable Model for Dissimilarity Data
Martin Jørgensen and Søren Hauberg
ArXiv: 2006.11741

Reparametrization Invariance for non-parametric Causal Discovery
Martin Jørgensen and Søren Hauberg
ArXiv: 2008.05552

Preprints not covered in this thesis

Probabilistic Spatial Transformers for Bayesian Data Augmentation
Pola Schwöbel, Frederik Warburg, Martin Jørgensen, Kristoffer H. Madsen and Søren Hauberg
ArXiv: 2004.03637

Contents

Chapter 0

Introduction

A major challenge in completing this thesis was finding a suiting title. The work, which it consists of, is varying among some fields that are not easily combined. It was however an aim for me to find a red thread and I hope the reader will find reading it easy. To illustrate this thesis will deal both with variance prediction in neural networks, Bayesian methods, Gaussian processes, and causal inference. it is perhaps not impossible to connect these, and I hope I have found a good thread, but giving one title to fit all was a major challenge. The title: 'Stochastic Representation using Gaussian processes and Geometry' is my best attempt in saying what will appear over the next approximately 80 pages. Representations are a forgiving word as it can represent many things at once. In Chapter 1 it will refer to representing variance in the simpler neural network models. Chapters 2 and 3 will revolve around Gaussian processes and latent representations, while Chapter 4 we will represent causal connections and how we can infer them.

The geometry role, insinuated by the title, is not immediately clear why it deserves a place there. Nevertheless, geometry has been a fundamental driver in many of the ideas on which this thesis is based. In Chapter 3, I argue why variance and uncertainty, in general, are connected when the geometry is viewed from a stochastic viewpoint rather than the usual deterministic perspective.

It was a goal for me to write this thesis such that it should be readable without reference to a load of prior works and without giving the reader a headache. To this aim, I will occasionally slack on minor technical details to preserve the overall profile throughout. So who should read this thesis? Readers will likely have a better time if

they are interested in probabilistic modelling, especially in the domains of regression and generative modelling. Most of the problems this thesis navigates about come from the intersection of statistics and machine learning with a pinch of differential geometry.

Chapter 1 is on uncertainty quantification in general, only the main focus is on variance estimation with neural networks. Initially, it introduces some elements of uncertainty and introduces the Bayesian language, which is a key actor while talking uncertainty. Nevertheless, it is seldom the case in large neural network models that the Bayesian regime is practical, and some of the best contemporary methodologies are simply Bayesian-inspired. The method presented at the end of the chapter is loosely Bayesian-inspired but has a focus on the optimisation process, which is claimed to behave differently for variance networks than for prediction networks.

Chapter 2 is about stochastic processes. It introduces, in particular, Gaussian processes from a constructive viewpoint and builds upon this to introduce both Wishart processes and stochastic differential equations. The latter has recently won much acclaim in the machine learning literature as a building block which unites neural networks with classical theory of differential equations. The contribution we propose here, unifies the theory of common dynamical models, such as auto-regressive models and Gaussian process, with the continuous-time model of a stochastic differential equation.

Chapter 3 is the most geometrically founded chapter of this thesis. The aim of it is to present a manifold learning algorithm based on Gaussian processes and Riemannian geometry. It additionally touches on topological considerations, which subsume the usual geometric features. A thorough introduction to understanding Riemannian geometry and manifold learning, in general, is provided. The presented method can likewise be seen as a Gaussian process Latent Variable Model, where the data is not given in an ambient coordinate system, but as pairwise distance.

Chapter 4 is, in my own opinion, the odd one out of this thesis. It deals with causal inference, in particular in the bivariate case. However, the connection to the rest of the thesis is present, as one of the main goals of the introduced algorithm is to formulate a language of uncertainty within causal inference based on causal invariances. To be more distinct on this, it is impossible to perform causal inference in the bivariate case, so we try and evaluate how delicate the (naive) estimators in this field are by perturbing them with a causal invariant — namely bijections of the marginal distributions.

I hope the reader will find the whole, or at least parts, of this thesis useful; if you are interested in similar topics, please do not hesitate to contact me.

Chapter 1

Uncertainty Quantification

Uncertainty Quantification (UQ) aims to inform us, in a quantitative way, of how much we do not know. However, there is also a qualitative element to it in answering question such as: *why do we not know?* and *how can we know more?* The latter question here is the driving force in Active Learning [Settles, 2009], where the target is to collect more data in order to expand the knowledge we already possess.

In this chapter, we describe some of the sources of uncertainty, and to which extent it is possible to differentiate them. We also look at Bayesian and approximate Bayesian inference, which are frameworks to describe uncertainty in the language of probabilities. Lastly, we go over some of the prominent methods for UQ in neural networks and present the contributions that is based on the paper *Reliable training and estimation of variance networks* [Detlefsen, Jørgensen, and Hauberg, 2019].

1.1 Separation of Uncertainty

Generally, all probabilistic inference fits some parameters θ of a given probability distribution p_θ. In machine learning the task can almost always be phrased as predicting $y(x)$, which probabilistically means optimising some form of $p_\theta(y)$ or $p_\theta(y|x)$. *Predictive uncertainty* refers to the uncertainty, or *variation*, in y under the predictive distribution p_θ: what is the range of likely outcomes of y? UQ is quantifying this variation.

Inference only cares about predictive uncertainty, but for many applications deeper questions about this quantity need answers. One occasionally hears people talk about *sources* of uncertainty or randomness, i.e. the question is, why are we uncertain about predicting y? Common sources are measurement noise, propagated uncertainty and uncertainty due to lack of measurements.

This *qualitative* assessment of uncertainty is central to tasks like Bayesian optimisation, where we wish to minimise a black-box function $f(x)$ over some bounded set $\mathcal{X}$. Here the *exploration-exploitation* trade-off is the key driver in the algorithm: should we *exploit* the information we already possess, or should we *explore* regions of $\mathcal{X}$ to gain more information? In Section 1.2 we introduce Bayesian inference, which — roughly said — is a treatment of UQ in the language of probability theory.

The next two subsections provide some notions of sources of uncertainty. I have to say that although aleatoric and epistemic uncertainty are 'well-established' notions in the literature, their uses are almost always ad-hoc. They do not provide an exhaustive or unique separation of uncertainty, nor are they mutually exclusive. In in this ad-hoc view, I present both aleatoric and epistemic uncertainty.

1.1.1 Aleatoric Uncertainty

Aleatoric originates from the Latin word *aleator*, meaning gambler or dice-player. The dice-player serves as a suiting analogy for explaining what aleatoric uncertainty is. Throwing a dice countless times to predict the outcome of the next throw, the experimenter will soon realise that misprediction occurs on average $\frac{5}{6}$ times. This rate is independent of the amount of data gathered in the experiment; the uncertainty is *irreducible*.

The irreducibility of aleatoric uncertainty is always *conditioned* on the model. We could reduce the noise by considering a different model, for example considering new features or changing the structure of the model. This only highlights there is uncertainty associated with our choice of model too; often we call this *structural uncertainty* or *bias*. Let ϵ be some noise with variance 1 and mean 0, and consider data generated by the additive form $y(x) = \mathbb{E}[(y(x)|X = x] + \sigma^2\epsilon$. Our model, $f(x)$, would try to approximate $\mathbb{E}[y(x)|X = x]$ and we could define, and split, the error as:

$$\begin{aligned} \mathbf{Error}(x) :&= \mathbb{E}[(y(x) - f(x))^2] && (1.1)\\ &= \underbrace{\big(\mathbb{E}[f(x)] - \mathbb{E}[y(x)|X = x]\big)^2}_{Bias^2} + \mathrm{Var}\big(f(x)\big) + \sigma_f^2, && (1.2) \end{aligned}$$

where σ_f^2 would be the aleatoric uncertainty conditioned on the model f.

Overfitting is a common issue in machine learning, and it is a consequence of aleatoric uncertainty *underestimation*: we start detecting a signal in the noise. On this inspection, regularization techniques are tools for accurately estimating the size

of the noise in the data, or equivalently removing model bias. Overfitting is exactly the situation in (1.2) where $\sigma_f^2 < \sigma^2$, because we introduce a bias towards the training data.

1.1.2 Epistemic Uncertainty

Again, we assume a model $f(x) = \mathbb{E}[Y|X = x]$. The uncertainty that is associated with f is what we will call *epistemic* uncertainty. The theorem of total variance states

$$\mathrm{Var}(Y) = \mathrm{Var}(\mathbb{E}[Y|X]) + \mathbb{E}[\mathrm{Var}(Y|X)], \tag{1.3}$$

which if we assume there is no bias, then σ_f^2 from (1.2) approximates the last term on the right-hand side of (1.3) and $\mathrm{Var}(f(x))$ approximates the $\mathrm{Var}(\mathbb{E}[Y|X])$. In this additive model f, $\mathrm{Var}(\mathbb{E}[Y|X])$ is what we will call epistemic variance. In this simplistic case, the predictive variation is merely the sum of aleatoric and epistemic variation. While the aleatoric uncertainty is irreducible, we get more and more certain about f as we see more data.

Example 1.1 *Assume we have N i.i.d. samples $\{x_i\}_{i=1}^N$ from a univariate Gaussian $\mathcal{N}(\mu, \sigma^2)$, but μ and σ^2 are unknowns. The maximum likelihood estimator for μ is given by the average*

$$\hat{\mu} = \frac{1}{N}\sum_{i=1}^{N} x_i \sim \mathcal{N}\Big(\mu, \frac{\sigma^2}{N}\Big), \tag{1.4}$$

and the maximum likelihood estimator for the variance σ^2 is

$$\hat{\sigma}^2 = \frac{1}{N-1}\sum_{i=1}^{N}(x_i - \hat{\mu})^2 \sim \frac{\sigma^2}{N-1}\chi^2(N-1). \tag{1.5}$$

We observe that our model, i.e. $\mathcal{N}(\hat{\mu}, \hat{\sigma}^2)$, has less uncertainty associated to its parameters as the sample size N increases.

It is in light of examples like the one above that epistemic uncertainty is often also called *model uncertainty*. It vanishes as the sample size increases. This however is subsumed by model identifiability, which means that our model is defined such that the *true* parameters can be obtained in the limit of infinite data. Identifiability is a condition, that is severely challenged by most modern machine learning models.

1.2 Bayesian Inference

Bayesian modelling assists us in formalising the discussion of epistemic uncertainty. I will review Bayesian probability theory and later on the approximate inference that allows us to efficiently model in this regime.

Example 1.1 illustrates that, given finite data, our estimators are uncertain. The distributions of the estimators indicate that the parameters could be estimated well by a range of point estimates. Alternatively, (1.4) states

$$\mu \sim \mathcal{N}(\hat{\mu}, \sigma^2/N), \tag{1.6}$$

meaning that the *ground truth* parameter μ follows a distribution that is dependent on the data through $\hat{\mu}$. In fact, the distribution in (1.6) is the *posterior* distribution of the mean parameters, given an improper *prior* distribution. These words are yet to be described, but this illustrates what Bayesian modelling is: we can represent our uncertainty about quantities using probability distributions.

A statistical model is a mathematical formulation of assumptions we make about how to generate data. It is helpful to think of this as a relationship between random variable(s) and possibly some non-random variables. To perform inference, we place a *likelihood* function at the top of the model. This allows us to compute the 'probability' of an event under our model. If we denote our model $\mathcal{S}$, which is parametrized with some parameters $\boldsymbol{\theta}$, we can generate data $\boldsymbol{y}$. The model then specifies a probability distribution over $\boldsymbol{\theta}$ and $\boldsymbol{y}$

$$p(\boldsymbol{y}, \boldsymbol{\theta}|\mathcal{S}) = p(\boldsymbol{y}|\boldsymbol{\theta}, \mathcal{S})p(\boldsymbol{\theta}|\mathcal{S}), \tag{1.7}$$

where $p(\boldsymbol{y}|\theta, \mathcal{S})$ is the likelihood function and $p(\boldsymbol{\theta}|\mathcal{S})$ is the prior distribution of our parameter(s) $\boldsymbol{\theta}$. Both of these are determined by our model assumptions.

The prior distribution is, as the name suggests, determined before (*a priori*) seeing data, and thus represents our uncertainty about the parameters before seeing the data. Bayesian inference is about determining the *posterior* distribution of the parameter(s); that is, the distribution after (*a posteriori*) seeing data: $p(\boldsymbol{\theta}|\boldsymbol{y}, \mathcal{S})$. Bayesian inference has its name from the equation that links these distribution together, *Bayes' rule*:

$$p(\boldsymbol{\theta}|\boldsymbol{y}, \mathcal{S}) = \frac{p(\boldsymbol{y}|\boldsymbol{\theta}, \mathcal{S})p(\boldsymbol{\theta}|\mathcal{S})}{p(\boldsymbol{y}|\mathcal{S})}. \tag{1.8}$$

Here, the denominator on the right-hand side, is called the *marginal likelihood*, given by

$$p(\boldsymbol{y}|\mathcal{S}) = \int p(\boldsymbol{y}|\boldsymbol{\theta}, \mathcal{S})p(\boldsymbol{\theta}|\mathcal{S})d\boldsymbol{\theta}, \tag{1.9}$$

which illustrates one of the dominant difficulties of Bayesian inference, this integral is rarely easily available.

The distinction between model $\mathcal{S}$ and parameters $\boldsymbol{\theta}$ is not crystal clear, and as such from the Bayesian perspective this is not an issue, as we could also marginalise the model by selecting a prior over models $p(\mathcal{S})$ and using Bayes' rule

$$p(\mathcal{S}|\boldsymbol{y}) = \frac{p(\boldsymbol{y}|\mathcal{S})p(\mathcal{S})}{p(\boldsymbol{y})} = \frac{p(\boldsymbol{y}|\mathcal{S})p(\mathcal{S})}{\int p(\boldsymbol{y}|\mathcal{S})p(\mathcal{S})d\mathcal{S}}, \tag{1.10}$$

which shows that the Bayesian framework also is suitable for model selection.

1.2.1 Approximate Inference

In the previous section, we stated that the marginal likelihood (1.9) is intractable for many models. This section is dedicated to approximating the intractable posterior that arises from this problem. In particular, we will focus on *variational inference*, that aims to minimise the 'distance' or 'difference' between the true posterior[1] $p(\boldsymbol{\theta}|\boldsymbol{y})$ and a *tractable* approximate distribution $q(\boldsymbol{\theta})$.

First of, we need to make clear how we measure the difference between two distributions. To this end, we consider the *Kullback-Leibler divergence* (KL-divergence) defined as

$$\mathrm{KL}(q\|p) = \int_{\mathcal{X}} \log\left(\frac{q(x)}{p(x)}\right) q(x)dx, \tag{1.11}$$

where $\mathcal{X}$ is the support of the probability measure p. The first important thing to notice is that $\mathrm{KL}(q\|p) = 0$ iff $p = q$ almost surely wrt. q. Further, $\mathrm{KL}(q\|p) \geq 0$ always. With this in mind, we can think of the KL-divergence as a quantification of how different two distributions are. We note that it is not a metric in any geometric sense, as neither symmetry nor triangle inequality is satisfied.

Thus, variational inference tries to infer a distribution $q(\boldsymbol{\theta})$, from some tractable family, that is close — in the KL sense — to the true posterior $p(\boldsymbol{\theta}|\boldsymbol{y})$. In general, it is impossible evaluate $\mathrm{KL}(q(\boldsymbol{\theta})\|p(\boldsymbol{\theta}|\boldsymbol{y}))$ as the true posterior is unknown, but we can use Bayes' rule

$$\begin{aligned}
\mathrm{KL}(\boldsymbol{q}(\theta)\|p(\boldsymbol{\theta}|\boldsymbol{y})) &= \int \log\left(\frac{q(\boldsymbol{\theta})}{p(\boldsymbol{\theta}|\boldsymbol{y})}\right) q(\boldsymbol{\theta})d\boldsymbol{\theta} \\
&= \int \log\left(\frac{p(\boldsymbol{y})q(\boldsymbol{\theta})}{p(\boldsymbol{y}|\boldsymbol{\theta})p(\boldsymbol{\theta})}\right) q(\boldsymbol{\theta})d\boldsymbol{\theta} \\
&= \int \log p(\boldsymbol{y})q(\boldsymbol{\theta})d\boldsymbol{\theta} - \int \log p(\boldsymbol{y}|\boldsymbol{\theta})q(\boldsymbol{\theta})d\boldsymbol{\theta} + \int \log\left(\frac{q(\boldsymbol{\theta})}{p(\boldsymbol{\theta})}\right) q(\boldsymbol{\theta})d\boldsymbol{\theta} \\
&= \log p(\boldsymbol{y}) - \mathbb{E}_q[\log p(\boldsymbol{y}|\boldsymbol{\theta})] + \mathrm{KL}(q(\boldsymbol{\theta})\|p(\boldsymbol{\theta})).
\end{aligned} \tag{1.12}$$

Thus, we have

$$0 \leq \log p(\boldsymbol{y}) - \mathbb{E}_q[\log p(\boldsymbol{y}|\boldsymbol{\theta})] + \mathrm{KL}(q(\boldsymbol{\theta})\|p(\boldsymbol{\theta})) \tag{1.13}$$

$$\iff \log p(\boldsymbol{y}) \geq \mathbb{E}_q[\log p(\boldsymbol{y}|\boldsymbol{\theta})] - \mathrm{KL}(q(\boldsymbol{\theta})\|p(\boldsymbol{\theta})) \tag{1.14}$$

with equality in (1.14) if and only if $q(\boldsymbol{\theta}) = p(\boldsymbol{\theta}|\boldsymbol{y})$. Since the left hand side of (1.14) does not depend on q, we can maximise the right hand side, which consists only of tractable quantities. Hence, minimising $\mathrm{KL}(\boldsymbol{q}(\theta)\|p(\boldsymbol{\theta}|\boldsymbol{y}))$ is equivalent to *maximising*

$$\mathcal{L}(q) := \mathbb{E}_q[\log p(\boldsymbol{y}|\boldsymbol{\theta})] - \mathrm{KL}(q(\boldsymbol{\theta})\|p(\boldsymbol{\theta})), \tag{1.15}$$

where $\mathcal{L}(q)$ is called the *evidence lower bound*, or for short: the *ELBO*.

[1] In this section, we leave out the $\mathcal{S}$-notation for the model.

This technique turns a difficult integration problem into a simple optimisation problem. However, for practical and computational reasons it is often necessary to have q be of a rather simple form and assume all the parameters are independent, often referred to as mean-field approximation. These approximations scale to complex and large models, however not result efficient.

Without detailing it, we note here another common technique for approximate Bayesian inference is *Markov Chain Monte Carlo* (MCMC). Here, one averts the integration problem by sampling from a Markov Chain which has an ergodic distribution equal to the true posterior. For a broader and deeper review on MCMC methods, we refer to Asmussen and Glynn [2007].

1.3 Neural Networks

Deep learning is a principal discipline of machine learning. The deepness does not entail the learning is more profound, but refers to a (deep) hierarchy of simpler models. These simpler models are often linear regression models

$$h(\boldsymbol{x}) = \boldsymbol{W}\boldsymbol{x} + \boldsymbol{b}, \tag{1.16}$$

where $\boldsymbol{W}$ is a matrix of parametric values often referred to as *weights*. $\boldsymbol{b}$ is the interception, often referred to as *bias*. A *deep* neural network is a composition of such simple models. However, a composition of linear regressions is a linear regression itself, and they simply do not cut it for large and complex datasets. To this end, to each of the simple models we associate *non-linearites*, so

$$h(\boldsymbol{x}) = \sigma(\boldsymbol{W}\boldsymbol{x} + \boldsymbol{b}). \tag{1.17}$$

The non-linearities σ, called the *activation functions*, are element-wise surjective maps into subsets of $\mathbb{R}$, such as $[0, \infty)$, $[-1, 1]$ or $[0, 1]$. Popular choices are the ReLU, tanh or sigmoid functions. The simplest of neural networks, the multi-layer perceptron (MLP), can then be written

$$f(\boldsymbol{x}) = h_L(h_{L-1}(\cdots h_2(h_1(\boldsymbol{x}))\cdots)), \tag{1.18}$$

and we say that f is a neural networks with L layers.

The simplicity of the layers means that much research in deep learning deals with how to connect and stack these simple models to compose models that can handle complicated data. Such data could for example be images, to which end *convolutional neural networks* [LeCun and Bengio, 1998] are helpful. They usually consist of some convolutional and pooling layers. Convolutional layers are linear operation that summarise spatial information of images, while pooling layers reduce the dimensionality of the pixels — also in a spatially consistent way.

To handle temporal or other sequential data one would use *recurrent neural networks*. The simplest description of these is that they generate a sequence of hidden states

$\boldsymbol{z}_1, \ldots, \boldsymbol{z}_T$, based on observed sequence $\boldsymbol{x}_1, \ldots, \boldsymbol{x}_T$,

$$\boldsymbol{z}_{t+1} = \sigma_z(\boldsymbol{W}_z \boldsymbol{x}_t + \boldsymbol{U}\boldsymbol{z}_t + \boldsymbol{b}_z), \tag{1.19}$$

and the model output $\boldsymbol{y}_t$, $t = 1, \ldots, T$, can then be computed from the hidden states

$$\boldsymbol{y}_t = \sigma_y(\boldsymbol{W}_y \boldsymbol{z}_t + \boldsymbol{b}_y). \tag{1.20}$$

More advanced recurrent structures are available through *Long Short-term Memory* units [Hochreiter and Schmidhuber, 1997] and similar.

In this thesis, we will only consider the simplest neural networks — the MLP, or fully connected feed-forward network, as presented with (1.18). We will not focus on *prediction*, but on assessing uncertainty to the predictions.

1.3.1 Uncertainty Quantification in Neural Networks

Neural networks can be used for diverse tasks such as medical diagnosis from images, autonomous driving, and voice recognition. The potential consequences of these decision-making systems are immense. What happens when your vehicle is confronted within an unforeseen event? The vehicle was perhaps trained on data from Los Angeles and has never seen a bicycle before. Although silly, this example illustrates the need for solutions in more realistic cases. One first attempt at a solution is to make the model treat such cases by returning not only a prediction, but return a notice that the observation lies outside of the data distribution used for training the neural network. Such information can be encoded by returning a high uncertainty in the prediction.

Good uncertainty estimation can also improve the model's prediction. This approach is often known as *active learning* [Settles, 2009]. Here, the model will identify which *unlabelled* data will increase its own performance by asking a human annotator to label these new data points. Often the model will use its own uncertainty quantification to choose these points. If done successfully, this can dramatically decrease the amount of required data needed for good model performance.

We will consider the task of quantifying the uncertainty in MLPs. To this end, we will consider a likelihood function of our target variable $\boldsymbol{y}$, and we will at first consider the Gaussian likelihood

$$p(\boldsymbol{y}|\boldsymbol{x}) = \prod_{i=1}^{N} p(y_i|\boldsymbol{x}_i) = \prod_{i=1}^{N} \mathcal{N}\big(y_i|\mu(\boldsymbol{x}_i), \sigma^2(\boldsymbol{x}_i)\big), \tag{1.21}$$

hence now μ and σ are functions, that can be modelled as neural networks.

As a first attempt for uncertainty quantification we could try giving neural networks the Bayesian treatment. Bayesian neural networks [MacKay, 1995] place prior distributions over the neural network's weights. This causes a distribution over

functions. Often one places the unit Gaussian prior distributions over each individual weight and often no priors on the bias vectors. The issue with Bayesian neural network is not their construction, but the difficulty of inferring the posterior distribution over all weights. Modern neural networks count their parameters in billions rather than hundreds, and the ultimate success of Bayesian neural networks is still awaiting. Approximate Bayesian inference schemes to scale to so large models is an active field of research.

1.3.2 Pseudo-Bayesian methods

Currently, Bayesian inference is delivering results that are subpar to the strongest candidates for uncertainty quantification. We will here focus on two of these methods that are inspired by the Bayesian paradigm, but neither involve priors nor any kind of posterior approximation. Their similarity lie in the fact that the models provide a distribution over functions, and this will provide a language to talk about model uncertainty without the Bayesian framework.

Monte Carlo Dropout. Neural networks have many parameters, which often makes it necessary to introduce some kind of regularization to avoid overfitting. One regularisation technique that is popular in the field is *Dropout* [Srivastava et al., 2014]. A Dropout layer is an exact copy of the previous layer, but each element is multiplied with a Bernoulli variable with probability parameter p. That is

$$\boldsymbol{h}_{l+1} = \boldsymbol{h}_l \circ \boldsymbol{U}, \tag{1.22}$$

where $\boldsymbol{U}$ is a vector of independent Bernoulli variables and $\circ$ denotes the Hadamard product (element-wise multiplication). Thus, during training we switch off — at random — some of the *neurons* in the neural network to make the model more robust to such perturbations.

Gal and Ghahramani [2016] propose to let these Dropout layers approximate the predictive uncertainty. That is for a new observation $\boldsymbol{x}^*$ they can approximate the predictive distribution $p(y^*|\boldsymbol{x}^*, \boldsymbol{X}, \boldsymbol{Y})$ by making M forward passes through the network including the Dropout layers at test time. Then we can estimate the mean $\mu(\boldsymbol{x})^*$ and the variance $\sigma^2(\boldsymbol{x}^*)$ with

$$\hat{\mu}(\boldsymbol{x}^*) = \frac{1}{M}\sum_{m=1}^{M} y_m^*, \tag{1.23}$$

$$\hat{\sigma}^2(\boldsymbol{x}^*) = \hat{\tau}^{-1} + \frac{1}{M}\sum_{m=1}^{M} y_m^* y_m^* - \hat{\mu}(\boldsymbol{x}^*)^2, \tag{1.24}$$

where τ is a measure of the noise in data, and is found by cross-validation.

Deep ensembles. A more straight-forward way of constructing a 'distribution' over functions is by training more than one deterministic function. This approach

was considered by Lakshminarayanan et al. [2017], and was inspired by *bootstrapping* [Efron, 1979] and *random forests* [Ho, 1995]. They use an ensemble of M individual neural networks, which each outputs a mean and a variance estimate $[\hat{\mu}(\boldsymbol{x}), \hat{\sigma}^2(\boldsymbol{x})]$. Each network is initialised independent and separately and is trained by maximising the log-likelihood. The outputs of the ensemble generates a mixture of Gaussians, which they use to compute by a single Gaussian with mean and variance

$$\mu(\boldsymbol{x}) = \frac{1}{M} \sum_{m=1}^{M} \hat{\mu}_m(\boldsymbol{x}), \tag{1.25}$$

$$\sigma^2(\boldsymbol{x}) = \frac{1}{M} \sum_{m=1}^{M} \left(\hat{\sigma}_m^2(\boldsymbol{x}) + \hat{\mu}_m(\boldsymbol{x}) \right) - \mu(\boldsymbol{x}), \tag{1.26}$$

and this variance is used as the predictive uncertainty.

1.4 A locally-aware sampler

In Detlefsen, Jørgensen, and Hauberg [2019] we considered how to train variance networks in a robust manner. By *variance network* we refer to a neural network which output is used as the variance in a likelihood evaluation. Hence $\sigma^2(\cdot)$ is a variance network in the above. In this paper we propose some tricks for more reliable estimations without looking to the Bayesian regime. Before detailing the approach, we will consider again some of the desired properties a variance network would satisfy

(i) Generally, neural networks are overconfident, i.e. they underestimate the predictive variance. We want well-calibrated variance.

(ii) We want the network to recognise out-of-distribution samples, and associates such with a high variance.

(iii) Improved variance estimation must not harm predictive performance.

We consider inferring function μ and σ^2 by gradient descent methods. Usually, in large models it is favourable to consider *stochastic* gradient descent (SGD), which means only considering a subset of the training set when taking gradient steps. SGD often avoids local minima, that usual gradient descent otherwise gets stuck in. SGD works when each subset of points — we will refer to this as a mini-batch — is an *unbiased* approximation of the actual loss function, which is a sum of many summands (one for each training point). In other words, the mini-batch is a representative sample of the training set.

First, let us consider one explanation why variance estimation struggles when we do not consider mini-batching, i.e. we compute our gradients with all the samples in the batch. Then consider the gradient of the log-likelihood — our objective function —

with respect to σ^2

$$\frac{\partial \log p(y|\mu, \sigma^2)}{\partial \sigma^2} = \frac{1}{2\sigma^2}\left(\frac{(y-\mu)^2}{\sigma^2} - 1\right). \tag{1.27}$$

Thus, the gradients are larger for small values of σ^2, which introduce a bias for heteroskedastic noise, since regions of small variation have more dominating gradients. This was first noted by Nix and Weigend [1994].

If we use mini-batching, it is a problem that the gradients wrt. σ^2 have a very high variance themselves. If we only have one sample, the maximum likelihood estimator of the variance does not even exist, and the estimator converges much slower than the mean estimator. Again if we have heteroskedastic noise, this introduces a problem as some regions' variance estimates are based on only one or two samples. We propose a sampler that averts this issue.

Locality sampler. We construct σ^2 as a continuous function, thus we are implicitly ensuring that $\sigma^2(\boldsymbol{x})$ correlates with $\sigma^2(\boldsymbol{x} + \boldsymbol{\delta})$ for sufficiently small $\|\boldsymbol{\delta}\|$. The idea behind the *locality sampler*, as presented in Detlefsen, Jørgensen, and Hauberg [2019], is to use this to construct an estimator of smaller variation. We present the algorithm first.

Pre-compute For each datapoint $\boldsymbol{x}_i$ in $\boldsymbol{X}$, compute the k nearest neighbours in $\boldsymbol{X}$. Store them in a matrix $\boldsymbol{\kappa}$ of size $N \times k$. Go to **Primary units**.

Primary units Sample m integers between 1 and N uniformly. We call these the m primary units. Go to **Secondary units**.

Secondary units For each of the primary unit, say i, sample n points from the ith row of $\boldsymbol{\kappa}$. Call these points the secondary units. Go to **Output**.

Output All unique values of the secondary units are the output. If a new sample is needed, go to **Primary units.**

The first step is a pre-computation to find the k nearest neighbours of each datapoint. This is a daunting computational task, but luckily it is a *one-off* computation. At the same time, there is no restriction to be exact and there exists computationally attractive approximate methods for this task [Fu and Cai, 2016].

The primary sampling units are sampled uniformly among the data points, this way they represent something global. In general, the number of primary units m should be kept small; between 1 and 3 works fine for all our experiments.

Secondary units, are those that end up in the new mini-batch. Each primary unit samples in a neighbourhood of itself, through the precomputed neighbours $\boldsymbol{\kappa}$. In this stage of sampling, we are interested in keeping things local, thus n should be kept

small, but large enough to have a decent Monte Carlo estimate; here we recommend in the range of $8 - 20$, depending on how dense the training set is.

Figure 1.1 visualise samples generated by the locality sampler. The red dots indicate the chosen mini-batch. On the x-axis is the explanatory variable in which we measure locality, and on the y-axis is the target variable in which we are interested in the variation. From this visualisation it should be clear that it is easier to estimate the variation locally, based on these sample as opposed to having red dots spread out with little local information.

The locality sampler raises another issue: the mini-batch is not representative, since some datapoints are more likely to appear in the batch than others. We can adjust for this by using the *Horvitz-Thompson* estimator [Horvitz and Thompson, 1952], i.e. rescaling the log-likelihood contribution of each sample $\boldsymbol{x}_i$ by its inclusion probability π_i. An unbiased estimate of the log-likelihood (up to an additive constant) is then

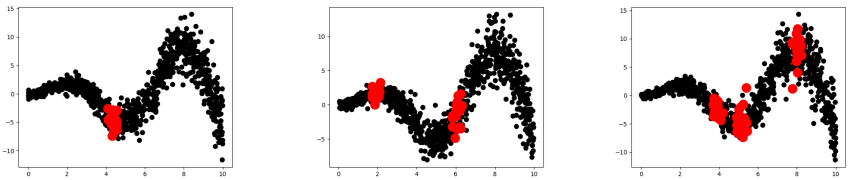
$$\sum_{i=1}^{N} \left\{ -\frac{1}{2}\log(\sigma^2(\boldsymbol{x}_i)) - \frac{(y_i - \mu(\boldsymbol{x}_i))^2}{2\sigma^2(\boldsymbol{x}_i)} \right\} \approx \sum_{\boldsymbol{x}_j \in \mathcal{O}} \frac{1}{\pi_j} \left\{ -\frac{1}{2}\log(\sigma^2(\boldsymbol{x}_j)) - \frac{(y_j - \mu(\boldsymbol{x}_j))^2}{2\sigma^2(\boldsymbol{x}_j)} \right\}$$

where $\mathcal{O}$ denotes the mini-batch. Based on the two-stage sampling algorithm, the inclusion probabilities can be computed easily. The probability that observation j is in the sample is n/k if it is among the k nearest neighbours of one of the initial m points, which are chosen with probability m/N, i.e.

$$\pi_j = \frac{m}{N} \sum_{i=1}^{N} \frac{n}{k} \, \mathbf{1}_{j \in \boldsymbol{\kappa}(i,\,)}, \tag{1.28}$$

where $\boldsymbol{\kappa}(i,\,)$ denotes the k nearest neighbours of $\boldsymbol{x}_i$, which is also the ith row of $\boldsymbol{\kappa}$.

Figure 1.2 shows a small experiment where we track the variance of the gradients during training. We train for 5000 iterations in this small example, and the first



(a) Local mini-batch with 1 primary unit.

(b) Local mini-batch with 2 primary units.

(c) Local mini-batch with 3 primary units.

Figure 1.1: Three examples of mini-batches (marked with red) generated from the locality sampler. In all cases $n = 10$, thus the batch sizes are 10, 20 and 30 from left to right. Black dots are the training set.

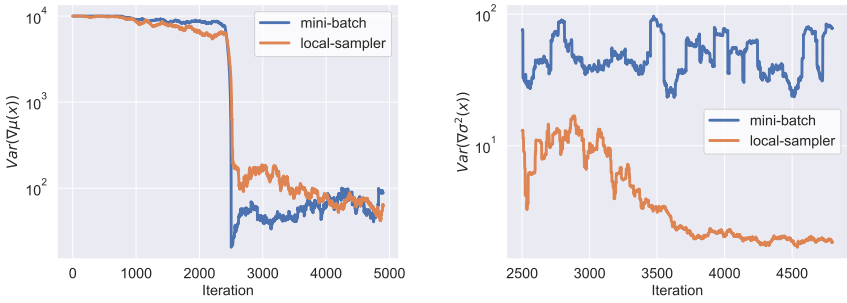


Figure 1.2: *Left*: Variance of mean gradient. *Right*: Variance of variance gradient. The variance network was disabled for the first 2500 iterations, to warm up the mean function for stable convergence.

2500 is solely training the mean with a constant fixed value for σ^2. We are naturally interested in the gradients of σ^2. We note that the variance is significantly smaller using the local sampler, and further there is a trend of it decreasing to some fixed small value indicating that the gradients from this point are close to 0.

Sequential training. A small additional trick we introduce is to sequentially update the networks μ and σ. The intuition is that when training σ^2 we want to keep μ fixed. This gives one extra degree of freedom in variance estimation (recall Example 1.1). We do not find that this training trick improve likelihood performance, but it seems to stabilise training and be less sensitive to initialisation.

Student-t likelihood. Even with the locality-sampler, we may end up with little data to approximate the true gradients. We propose a robust pseudo-Bayesian workaround; instead of point estimating $\sigma(\boldsymbol{x})^2$ we fit a distribution. This is *not* imposing a prior, we are merely training the parameters of a hierarchical model. We choose the inverse-Gamma distribution, which is conjugate prior of σ^2 when the likelihood is Gaussian. This means we have two parameters, $\alpha, \beta > 0$, which are the shape and scale parameter of the inverse-Gamma respectively. So the log-likelihood is now calculated by integrating out σ^2, which is inverse-Gamma

$$\log p_\theta(y_i) = \log \int \mathcal{N}(y_i|\mu_i, \sigma_i^2) \mathrm{d}\sigma_i^2 = \log t_{\mu_i,\alpha_i,\beta_i}(y_i), \tag{1.29}$$

where $\alpha_i = \alpha(\boldsymbol{x}_i), \beta_i = \beta(\boldsymbol{x}_i)$ are modelled as neural networks. The predictive distribution is now a located-scaled Student-t distribution, parametrized by μ, α and β. This is a common replacement of the Gaussian when data is scarce and the true variance is unknown and yields a *robust* regression [Gelman et al., 2014, Lange et al., 1989].

Variance extrapolation. Up to this points we are yet to consider how we

detect out-of-distribution observations. The method we will consider for this is inspired by the variance extrapolation seen in Gaussian processes with stationary kernels. Gaussian processes will be thoroughly introduced in Chapter 2. We will also take inspiration in the inducing point methods from these field.

We will mimic this behaviour by letting the variance network go to an *a priori* determined value η, if evaluated at a point $\boldsymbol{x}^*$ far away from the training data. To this end, let $\{\boldsymbol{c}_i\}_{i=1}^L$ be points in $\mathbb{R}^D$ that capture the structure of the training data, akin to inducing points in sparse Gaussian processes [Snelson and Ghahramani, 2006]. We may think of e.g. k-means. Now define $\delta(\boldsymbol{x}^*) = \min_i \|\boldsymbol{c}_i - \boldsymbol{x}^*\|$ and

$$\sigma^2(\boldsymbol{x}^*) = \big(1 - s(\delta(\boldsymbol{x}^*))\big)\hat{\sigma}^2 + \eta s(\delta(\boldsymbol{x}^*)), \tag{1.30}$$

where $s : [0, \infty) \to [0, 1]$ is an increasing function. Here $\hat{\sigma}$ is what the variance network outputs, and we can think of this as a post-processing of the variance estimate. This variance estimate will then tend to η as $\delta \to \infty$ at a rate determined by s. In practice, we let s to be a scaled-and-translated sigmoid function

$$s(x) = \text{sigmoid}((x + a)/\gamma), \tag{1.31}$$

and meanwhile ensure that $s(0) \approx 0$ by fixing a to a linear function of γ. The inducing points $\boldsymbol{c}_i$ are initialised with k-means and optimised during training.

In summary, the final variance estimate is a convex combination of η and the variance network output. It lies close to the network output, when $\boldsymbol{x}^*$ lies close to the inducing points $\{\boldsymbol{c}_i\}_{i=1}^L$, but close to η if not. Thus, the behaviour mimics that of Gaussian process posteriors.

1.5 Model uncertainty and noise

One thing that the methods presented by Detlefsen, Jørgensen, and Hauberg [2019] pay little attention to is a separation of epistemic and aleatoric uncertainty. This section provides a way to think about these notions in terms of the Student-t likelihood.

Recall, the conjugate prior of the variance σ^2 in a Gaussian with known mean μ is an inverse Gamma distribution with shape and rate parameters $\alpha > 0$ and $\beta > 0$. Marginalising this prior out yields to the following Student's t-distribution:

$$\begin{aligned} p(y \mid \mu, \alpha, \beta) &= \int_0^\infty \frac{\beta^\alpha}{\Gamma(\alpha)} \tau^{\alpha-1} e^{-\tau\beta} \left(\frac{\tau}{2\pi}\right)^{\frac{1}{2}} e^{\left(-\frac{\tau}{2}(x-\mu)^2\right)} d\tau \\ &= \frac{\beta^\alpha}{\Gamma(\alpha)} \frac{1}{\sqrt{2\pi}} \int_0^\infty \tau^{\left(\alpha+\frac{1}{2}\right)-1} e^{-\tau(\beta+(x-\mu)^2/2} d\tau \\ &= \frac{\beta^\alpha}{\Gamma(\alpha)} \frac{1}{\sqrt{2\pi}} \frac{\Gamma(\alpha + \frac{1}{2})}{\left(\beta + \frac{1}{2}(x-\mu)^2\right)^{\alpha+\frac{1}{2}}}, \end{aligned} \tag{1.32}$$

where we reparametrize in terms of precision, i.e. if $\sigma^2 \sim \text{Inv-Gamma}(\alpha, \beta)$ then $\tau = \frac{1}{\sigma^2} \sim \Gamma(\alpha, \beta)$ and use the properties of Gamma integral. This is the same as (1.29).

We can compute the first two moments

$$\mathbb{E}[y] = \mu, \tag{1.33}$$

$$\text{Var}(y) = \frac{\beta}{\alpha - 1} = \frac{\beta}{\alpha}\frac{\alpha}{\alpha - 1}. \tag{1.34}$$

By a bijective transformation of the parameters $\nu = 2\alpha$ and $\kappa = \beta/\alpha$, we can now see that

$$y \sim \mu + \kappa t(\nu), \tag{1.35}$$

where ν denotes the degrees of freedom of the t-distribution and κ is a scaling parameter. Note we here constrain $\alpha > 1$ to have positive variance. We recall that $\text{Var}(t(\nu)) = \nu/(\nu - 2)$, or equivalently $\alpha/(\alpha - 1)$, and its limits for $\nu \to \infty$ and $\nu \to 2$ are 1 and ∞, respectively.

This inspection suggests we may regard $\alpha/(\alpha - 1)$ as a proxy for the epistemic uncertainty, since it vanishes as we see more data — the degrees of freedom increase. By *vanishing* we mean the predictive variance, (1.34), reduces to β/α, which we interpret as the variation from noise in the data — aleatoric uncertainty. The model we propose for uncertainty quantification and extrapolation consists of three neural networks. One for the mean prediction μ, one to account for epistemic uncertainty and extrapolation α, and one for aleatoric matters β.

In practice, we train μ and α together and β on its own. We do this sequentially as suggested above. We use the locality sampler to train β. For α, we train with the variance extrapolation (inducing points method), but *not* with a predefined value η. Instead we can set

$$\alpha(\boldsymbol{x}) = \hat{\alpha}(\boldsymbol{x})\Big(1 - s(\delta(\boldsymbol{x})\Big) + s(\delta(\boldsymbol{x})), \tag{1.36}$$

where $\hat{\alpha}$ is the output of the network. Thus, when $\alpha(\boldsymbol{x}^*) \to 1$, because the point $\boldsymbol{x}^*$ moves away from the training data, then $\text{Var}(y^*) \to \infty$, as a consequence of (1.34). It is this model we will evaluate in the next section.

1.6 Experiments

I thank Federico Bergamin for good discussions and help with the experiments in this section.

1-D Regression Example. We analyse our method on a simple 1D regression example. We generate 500 observations from the function

$$y = x \cdot \sin(x) + 0.3 \cdot \epsilon_1 + 0.3 \cdot x \cdot \epsilon_2, \tag{1.37}$$

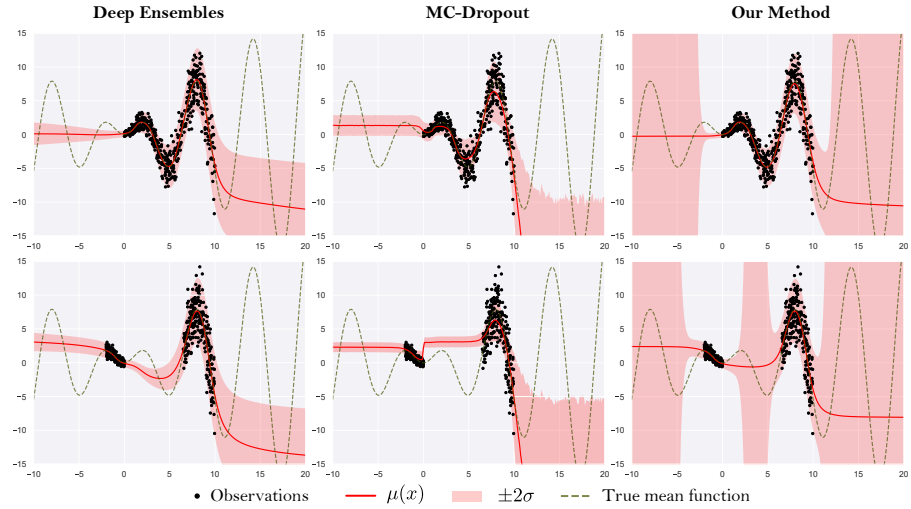


Figure 1.3: *Top row*: Simple one-dimensional target with a dense dataset. *Bottom row*: Same true function, but with a non-dense dataset. The columns correspond to the three considered methods. We observe how only our method extrapolates, and two standard deviations *covers* the true underlying function in both cases.

where $\epsilon_1, \epsilon_2 \sim \mathcal{N}(0,1)$. We sample our x uniformly in the interval $[0, 10]$. We will also consider sampling x uniformly in the intervals $[-2, 0]$ and $[6.5, 10]$. This is to evaluate the extrapolation features.

The results are shown in Fig. 1.3. Here the top row represents the dense dataset and the bottom row represents the dataset with observations in two disjoint intervals. Our method and Deep Ensembles capture the heteroscedastic variance within the data regions without underestimating it. MC-Dropout underestimates the variance within the data region. Outside the data region and in-between cluster regions we see only our method extrapolates high variance. Two standard deviations (shaded pink) covers the true underlying function for all x. Deep Ensembles and MC-Dropout fail to extrapolate high uncertainty both outside and in-between data region.

The relationship between α and β is plotted in Figure 1.4. We observe that α learns the absence of data by going to 1, while β learns the increasing noise as a function of x. The values of β are arbitrary outside the data domain. α has a tendency to decrease with x, inside the data domain; this is both a wanted and unwanted behaviour, as model uncertainty would correlate negatively with noise, but should also represent the density in x.

UCI Benchmark. We evaluate the method on the UCI Regression Benchmark

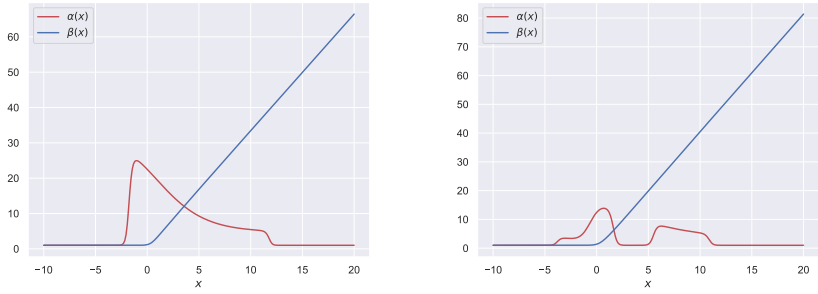


Figure 1.4: *Left*: α and β on dense data. *Right*: α and β on data with gap.

introduced by Hernández-Lobato and Adams [2015]. Beyond this, we will consider the same dataset but preprocessed as suggested in Foong et al. [2019]; they suggest making the test-sets be the middle third of some feature when that feature is sorted. That is, for each variable in the input space, sort it and keep the points in the middle third out to use as a test set. By doing so, they can hope to create holes as we saw in the 1-D simulated example. The aim is again to see the extrapolation capabilities, but on real data. We refer to these splits as *gap splits*. For the standard splits, we use 70/30 for train-test to make it comparable to the approach of Foong et al. [2019].

The evaluations are seen in Table 1.1 for the random splits, and in Table 1.2 for the gap splits. A good uncertainty estimator should have good performance on the random splits and avoid catastrophic failure on these tailored splits. We observe that our method is competitive in both regimes; we especially note the small variation over the gap splits our method has compared to Deep Ensembles and MC-Dropout. This indicates more robustness towards unexpected test-set, or out-of-distribution samples. This is especially noticeable on the *Energy* and *Naval* datasets.

Future directions

The contribution in this chapter introduces methods for a single network to represent the uncertainty, or variance, of a regression model. The natural extension could be to 'ensemble' these too, potentially getting even better epistemic uncertainty estimates. The approach we took by modelling the degrees of freedom as a proxy for the epistemic uncertainty shows some promise of delivering good results. This area generally has been studied scarcely in the literature of neural networks [Gao and Jojic, 2016]. However, some parallels can be drawn with investigations into influence functions [Madras et al., 2019].

Dataset	Deep Ensembles	MC-Dropout	Our
Boston	-2.722 ± 0.222	-2.461 ± 0.163	-2.497 ± 0.092
Concrete	-2.896 ± 0.150	-3.003 ± 0.051	-3.079 ± 0.093
Energy	-1.507 ± 0.671	-1.242 ± 0.041	-1.315 ± 0.010
Kin8nm	1.202 ± 0.013	1.053 ± 0.018	1.258 ± 0.029
Naval	4.327 ± 0.474	4.190 ± 0.080	5.526 ± 0.660
Power plant	-2.756 ± 0.018	-2.823 ± 0.018	-2.791 ± 0.012
Protein	-2.812 ± 0.007	-2.913 ± 0.005	-2.814 ± 0.016
Wine (red)	-1.163 ± 0.138	-0.938 ± 0.023	-0.920 ± 0.023
Yacht	-0.127 ± 0.208	-1.284 ± 0.052	-0.957 ± 0.156

Table 1.1: Test log-likelihoods on the UCI dataset when using random splits. Results are the average of 10 different train/test splits, apart from the *Protein* dataset where we only used 5 splits. Higher is better.

Dataset	Deep Ensembles	MC-Dropout	Our
Boston	-2.972 ± 0.445	-2.516 ± 0.190	-2.579 ± 0.107
Concrete	-3.742 ± 0.234	-3.368 ± 0.120	-3.614 ±0.206
Energy	-5.595 ± 7.905	-3.827 ± 2.987	-2.905 ± 0.477
Kin8nm	1.186 ± 0.052	0.978 ± 0.091	1.222 ± 0.066
Naval	0.195 ± 4.273	2.136 ± 0.815	2.742 ± 0.154
Power plant	-2.907 ± 0.085	-2.924 ± 0.037	-2.934 ± 0.050
Protein	-2.975 ± 0.068	-3.039 ± 0.048	-3.068 ± 0.057
Wine (red)	-1.553 ± 0.209	-0.970 ± 0.046	-0.930 ± 0.047
Yacht	-1.330 ± 0.331	-2.031 ± 0.624	-1.809 ± 0.296

Table 1.2: Test log-likelihoods on the UCI dataset when using gap splits, there are as many splits as features in the datasets. The aim is to study if the method is able to estimate the in-between uncertainty. Higher is better.

The issue of having 'inducing points' in the network was inspired by Gaussian processes — which will be formally introduced in the next chapter — and, recently, there has been inquiries into how many of these are needed to well-approximate the true posterior [Burt et al., 2019]. However, the issue here is different. Nevertheless, it would be important to have similar studies for how many of these points are needed; and just as important: how to train and initialise them. A radically different approach would be to discard them altogether. We use them to extrapolate variances, but this can potentially be achieved in other ways.

An idea to this end would be to, in an efficient manner, 'count' the number of activated neurons in the first layer. Intuitively, there should be numerous such activations 'on' for the training set (and naturally also for test points close to the training set). The number of activations away from the training set, however, is arbitrary. We have begun small experiments to this end, but are yet to overcome this arbitrary behaviour away from the training data to reach the coveted quantities of variance networks.

In the paper [Detlefsen, Jørgensen, and Hauberg, 2019], we further did experiments for variational autoencoders, but exclusively used the methods on the decoder part. But if good epistemic uncertainty estimates are available, it would be interesting to examine if they have a big effect on the encoder as well. Lastly, the impact on classification tasks has not been investigated.

Chapter 2

Stochastic Processes and Fields

The previous chapter focused mostly on uncertainty quantification in neural network models. In this chapter, we turn to a class of functions that are notoriously famous for exactly UQ – *Gaussian processes*. Their 'supremacy' in UQ is due to their Bayesian nature, and we will see that optimisation essentially is redundant, since posteriors can be computed in closed form. That sounds too good to be true, and unfortunately, in practice, it is. We will see approximations to Bayesian inference in Gaussian Processes, that allow us to scale these method to large datasets.

We will also focus on *Wishart processes* and *Itô processes*. With these three types of processes we present a very flexible model for both deep learning and dynamical systems. This is based on the paper *Stochastic Differential Equations with Variational Wishart Diffusions* [Jørgensen, Deisenroth, and Salimbeni, 2020].

We begin this chapter by formalising what a stochastic process is. As such, the word *process* makes one's mind think of something developing over time. This analogy is comprehensive for many use-cases of stochastic processes, but they are more flexible than so. A stochastic process has an associated non-empty index set $\mathcal{X}$ and state space $\mathcal{Y}$. A collection of random variables $\{Y(x)\}_{x\in\mathcal{X}}$, taking values in $\mathcal{Y}$, is then said to be a stochastic process.

Example 2.1 (Random walk) *Consider an index set $\mathcal{X} = \mathbb{N}_0$ and state space*

$\mathbb{Z}$, define a stochastic process by

$$Y(n) = 0 \quad \textit{for } n = 0, \quad \textit{and} \quad Y(n) = Y(n-1) + U_n \quad \textit{for } n > 1, \tag{2.1}$$

where U_n is a random variable, which is 1 with probability $\frac{1}{2}$ and -1 with probability $\frac{1}{2}$. Here Y is a stochastic process, usually referred to as a random walk, where each state is dependent on its left neighbour in the index set.

In the above example, it is intuitive to think of the index set as a discrete *temporal* feature: how many steps has been taken. $Y(n)$ is the state of the process after n steps, and it becomes more difficult to predict this state as n increases, unless we condition on one of the previous states, e.g. if we know $Y(n-1) = 2$, we know $Y(n) \in \{1, 3\}$.

We can think of situations where the index set does not have a temporal flavour. This could be geological locations, often the case in geostatistics, where *kriging* is a common method for interpolating locations.

Parameters or not. In Chapter 1 we consider parametric models, namely neural networks, which have the property that

$$p(y^*|\boldsymbol{\theta}, \mathcal{D}) = p(y^*|\boldsymbol{\theta}), \tag{2.2}$$

where $\mathcal{D}$ denotes the training set and $\boldsymbol{\theta}$ are the parameters of the model. This means that all information from the training set is conveyed by the parameters of the model. In this sense, we can not guarantee that new predictions, say y^*, make use of the *whole* training set. Parametric models are convenient for a multitude of reasons, especially the capability of compressing very large datasets. In a sense, what we will propose in the next sections can be viewed as *non-parametric*, as we will not compress the dataset in parameters, but keep the entire dataset in memory. Perhaps, unintuitively non-parametric models can be seen as having infinite-dimensional parameter spaces [Orbanz, 2012], hence providing very flexible models. *Bayesian non-parametrics* is a framework for placing priors over non-parametric models and *Gaussian processes* are in this framework.

2.1 Gaussian Processes

Definition 2.1 A *Gaussian process* is the stochastic process satisfying that for any finite collection $(\boldsymbol{x}_1, \boldsymbol{x}_2, \ldots, \boldsymbol{x}_N) \subset \mathcal{X}$, for any $N \in \mathbb{N}$, we have that $\big(Y(\boldsymbol{x}_1), \ldots, Y(\boldsymbol{x}_N)\big)$ follows a N-variate Gaussian distribution.

Necessarily, this implies that the state space of a Gaussian process is $\mathbb{R}$. Furthermore, it implies we can always write

$$\boldsymbol{Y} \sim \mathcal{N}\Big(\boldsymbol{\mu}, \mathrm{Cov}(\boldsymbol{Y}, \boldsymbol{Y})\Big), \tag{2.3}$$

where $\boldsymbol{Y} = \big(Y(\boldsymbol{x}_1), \dots, Y(\boldsymbol{x}_N)\big)^\top$, $\boldsymbol{\mu} = (\mu_{Y(\boldsymbol{x}_1)}, \dots, \mu_{Y(\boldsymbol{x}_N)})^\top$ and

$$\mathrm{Cov}(\boldsymbol{Y}, \boldsymbol{Y}) = \begin{pmatrix} \mathrm{Cov}(Y(\boldsymbol{x}_1), Y(\boldsymbol{x}_1)) & \mathrm{Cov}(Y(\boldsymbol{x}_1), Y(\boldsymbol{x}_2)) & \cdots & \mathrm{Cov}(Y(\boldsymbol{x}_1), Y(\boldsymbol{x}_N)) \\ \mathrm{Cov}(Y(\boldsymbol{x}_2), Y(\boldsymbol{x}_1)) & \ddots & & \vdots \\ \vdots & & \ddots & \vdots \\ \mathrm{Cov}(Y(\boldsymbol{x}_N), Y(\boldsymbol{x}_1)) & \cdots & \cdots & \mathrm{Cov}(Y(\boldsymbol{x}_N), Y(\boldsymbol{x}_N)) \end{pmatrix}, \tag{2.4}$$

is a symmetric and positive semi-definite $N \times N$-matrix.

Gaussian distributions are uniquely determined by their first two moments, and the same applies to Gaussian processes. In this light, it would be convenient if we could determine the first two moments solely with the index set. Initially, let us consider the simplistic case $\mathrm{Cov}(\boldsymbol{Y}, \boldsymbol{Y}) := \boldsymbol{X}\boldsymbol{X}^\top$, where $\boldsymbol{X} = (x_1, \dots, x_N)$. Here, we assume the index set is $\mathbb{R}$, but it generalises to $\mathbb{R}^d$, for $d > 1$. Clearly, $\boldsymbol{X}\boldsymbol{X}^\top$ is symmetric and positive semi-definite. Now let us consider the reparametrization trick, as it has been dubbed in the deep learning community. It holds that

$$\boldsymbol{Y} \sim \mathcal{N}\Big(\boldsymbol{\mu}, \mathrm{Cov}(\boldsymbol{Y}, \boldsymbol{Y})\Big) \sim \boldsymbol{\mu} + \boldsymbol{L}\boldsymbol{U}, \qquad \text{where } \boldsymbol{U} \sim \mathcal{N}(\boldsymbol{0}, \boldsymbol{I}_N), \tag{2.5}$$

if $\boldsymbol{L}$ is a $N \times N$-matrix such that $\mathrm{Cov}(\boldsymbol{Y}, \boldsymbol{Y}) = \boldsymbol{L}\boldsymbol{L}^\top$.

In the simplistic case, we can choose $\boldsymbol{L} := (\boldsymbol{X}, \boldsymbol{0}, \dots, \boldsymbol{0})$, i.e. the $N \times N$-matrix with first column $\boldsymbol{X}$ followed by $N - 1$ columns of $\boldsymbol{0}$. This implies that (2.5) further is simplified by

$$\boldsymbol{Y} \sim \boldsymbol{\mu} + U\boldsymbol{X}, \qquad \text{where } U \sim \mathcal{N}(0, 1), \tag{2.6}$$

which is a linear regression model with *random* standard Gaussian slope coefficient!

By inspecting this covariance matrix

$$\boldsymbol{X}\boldsymbol{X}^\top = \begin{pmatrix} \boldsymbol{x}_1\boldsymbol{x}_1 & \boldsymbol{x}_1\boldsymbol{x}_2 & \cdots & \boldsymbol{x}_1\boldsymbol{x}_N \\ \boldsymbol{x}_2\boldsymbol{x}_1 & \ddots & & \vdots \\ \vdots & & \ddots & \vdots \\ \boldsymbol{x}_N\boldsymbol{x}_1 & \cdots & \cdots & \boldsymbol{x}_N\boldsymbol{x}_N \end{pmatrix}, \tag{2.7}$$

we see that each entry is an inner product of observations pairwise.

The kernel trick. The linear regression above followed easily from the most naive choice of positive semi-matrix matrix, but linear regression is not a particularly flexible model. To introduce non-linearity we consider *the kernel trick*. The idea is to find a map $\varphi : \mathcal{X} \to \mathcal{H}$, and do the linear regression in the space $\mathcal{H}$, which often is of higher dimension than $\mathcal{X}$. This requires that the inner product $\langle \varphi(\boldsymbol{x}_i), \varphi(\boldsymbol{x}_j) \rangle_{\mathcal{H}}$ is

well-defined for all pairs $(\boldsymbol{x}_i, \boldsymbol{x}_j)$. We will also require that the covariance matrix

$$\begin{aligned}\boldsymbol{K} &= \begin{pmatrix} k(\boldsymbol{x_1}, \boldsymbol{x}_1) & k(\boldsymbol{x}_1, \boldsymbol{x_2}) & \cdots & k(\boldsymbol{x}_1, \boldsymbol{x}_N) \\ k(\boldsymbol{x}_2, \boldsymbol{x}_1) & \ddots & & \vdots \\ \vdots & & \ddots & \vdots \\ k(\boldsymbol{x}_N, \boldsymbol{x}_1) & \cdots & \cdots & k(\boldsymbol{x}_N, \boldsymbol{x}_N) \end{pmatrix} \\ &:= \begin{pmatrix} \langle \varphi(\boldsymbol{x_1}), \varphi(\boldsymbol{x}_1) \rangle_{\mathcal{H}} & \langle \varphi(\boldsymbol{x_1}), \varphi(\boldsymbol{x_2}) \rangle_{\mathcal{H}} & \cdots & \langle \varphi(\boldsymbol{x_1}), \varphi(\boldsymbol{x_N}) \rangle_{\mathcal{H}} \\ \langle \varphi(\boldsymbol{x_2}), \varphi(\boldsymbol{x}_1) \rangle_{\mathcal{H}} & \ddots & & \vdots \\ \vdots & & \ddots & \vdots \\ \langle \varphi(\boldsymbol{x_N}), \varphi(\boldsymbol{x}_1) \rangle_{\mathcal{H}} & \cdots & \cdots & \langle \varphi(\boldsymbol{x_N}), \varphi(\boldsymbol{x}_N) \rangle_{\mathcal{H}} \end{pmatrix},\end{aligned} \tag{2.8}$$

is positive semi-definite. In the above, we defined the function $k(\boldsymbol{x}_i, \boldsymbol{x}_j) := \langle \varphi(\boldsymbol{x_i}), \varphi(\boldsymbol{x}_j) \rangle_{\mathcal{H}}$. Thus k must ensure $\boldsymbol{K}$ to be positive semi-definite.

Definition 2.2 A positive semi-definite function is a function $k : \mathcal{X} \times \mathcal{X} \to \mathbb{R}$ such that for any finite collection $\{\boldsymbol{x}_i\}_{i=1}^N$ of elements from $\mathcal{X}$, the matrix $\boldsymbol{K}$ from (2.8) is positive semi-definite. Equivalently,

$$\sum_{i=1}^{N} \sum_{j=1}^{N} k(\boldsymbol{x}_i, \boldsymbol{x}_j) c_i c_j \geq 0, \tag{2.9}$$

for any sequence $(c_1, \ldots, c_n)$ of real numbers.

To sum this far, $\boldsymbol{K}$ is a valid covariance matrix, if k satisfies: (i) $k(\boldsymbol{x}_i, \boldsymbol{x_j}) \geq 0$ if $i = j$, (ii) $k(\boldsymbol{x}_i, \boldsymbol{x}_j) = k(\boldsymbol{x}_j, \boldsymbol{x}_i)$ and (iii) k is a positive semi-definite function. We say k is a *covariance function* if it satisfies (i)-(iii).

With this construction, it is true that

$$\boldsymbol{Y} \sim \mathcal{N}(\boldsymbol{\mu}, \boldsymbol{K}), \tag{2.10}$$

whenever k is a covariance function on $\mathcal{X}$. The Kombasi-Karhunen-Loève theorem also gives the other direction: if Y is a Gaussian process such that $\mathrm{Cov}(Y(\boldsymbol{x}_i), Y(\boldsymbol{x}_j)) = k(\boldsymbol{x}_i, \boldsymbol{x}_j)$, then k is a covariance function and further

$$Y(\boldsymbol{x}) = \mu(\boldsymbol{x}) + \sum_{n=1}^{\infty} U_n \phi_n(\boldsymbol{x}), \qquad \text{where } U_n \sim \mathcal{N}(0, \lambda_n), \tag{2.11}$$

where $\{\phi_n, \lambda_n\}_{n=1}^{\infty}$ are the eigenfunctions and eigenvalues of the covariance function respectively. That is, the possibly infinite number of pairs, which satisfy the equation

$$\int_{\mathcal{X}} k(\boldsymbol{x}, \boldsymbol{x}^*) \phi(\boldsymbol{x}) d\nu(\boldsymbol{x}) = \lambda \phi(\boldsymbol{x}^*), \tag{2.12}$$

where ν is a suitable probability measure on $\mathcal{X}$. The eigenfunctions make up an orthonormal basis of $\mathcal{H}$, and as such we should view (2.11) as a linear regression of

these basis functions in the space $\mathcal{H}$ — the space φ maps to. The 'trick' in the kernel trick is that we never have to visit this space, we never explicitly formulate the mapping φ, when the *kernel* k is known.

Kernels. We will go over some typical kernels, and show how modern approaches have increased the flexibility of these. As seen earlier, the simplest kernel is the inner products of inputs itself

$$k(\boldsymbol{x}_i, \boldsymbol{x}_j) = \alpha\langle \boldsymbol{x}_i, \boldsymbol{x}_j \rangle + \beta, \tag{2.13}$$

which, as we saw, gave a linear relationship between inputs and outputs. In a straightforward way, we can generalise to polynomial regression by multiplying the kernel with itself. That is,

$$k(\boldsymbol{x}_i, \boldsymbol{x}_j) = \big(\alpha\langle \boldsymbol{x}_i, \boldsymbol{x}_j \rangle + \beta\big)^d, \tag{2.14}$$

is also a covariance function and makes a polynomial relationship of inputs and outputs.

This construction is a feature of a more general design: the space of kernel functions are closed under both multiplication and addition. This implies, if k_1 and k_2 are both kernel functions, then both $k_1 + k_2$ and $k_1 k_2$ are kernel functions too. Designing kernels are one of the major challenges of Gaussian process modelling and this characteristic of kernels provide a flexible construction, as for example used by Duvenaud et al. [2011]. Here, we also remark another aspect of kernels that provide adaptable modelling choices. We can change the inputs to the kernel with a function $g : \mathcal{X} \to g(\mathcal{X})$ and $k(g(\boldsymbol{x}_i), g(\boldsymbol{x}_j))$ is still an inner product. This was used by Wilson et al. [2016] and Calandra et al. [2016].

Arguably, the most popular kernel in machine learning is the *Radial Basis Function* (RBF), also known as the *Gaussian kernel*, which is given by

$$k(\boldsymbol{x}_i, \boldsymbol{x}_j) = \sigma^2 \exp\Big(-\frac{\|\boldsymbol{x}_i - \boldsymbol{x}_j\|^2}{2\gamma}\Big), \tag{2.15}$$

where γ is known as the *lengthscale*, that dictates whether the outputs have long- or short range dependencies in the input space. σ^2 is referred to as the signal variance. By applying some of the aforementioned features, we can generalise this kernel using *Automatic Relevance Determination* (ARD) which transforms the input space with a positive diagonal matrix $\boldsymbol{A}$

$$k(\boldsymbol{x}_i, \boldsymbol{x}_j) = \sigma^2 \exp\Big(-\frac{\|\boldsymbol{A}\boldsymbol{x}_i - \boldsymbol{A}\boldsymbol{x}_j\|^2}{2}\Big) = \sigma^2 \exp\Big(-\sum_{l=1}^{d} \frac{\|x_{il} - x_{jl}\|^2}{2\gamma_l}\Big), \tag{2.16}$$

where d is the dimension of the input space and l denotes the l-th dimension of it. We see $\boldsymbol{A}_{ll} = \frac{1}{\gamma_l}$, and as such each dimension now has its own lengthscale. Inferring these lengthscales has the potential of determining the relevant features and turn off the irrelevant ones, hence the name ARD. We will refer to this kernel (2.16) as the ARD-kernel. Other notable mentions of popular kernels are the Matérn kernels, which we will not detail here.

Distributions over functions. Mathematically, the definition of a GP (Definition 2.1) does not mean that it is a *stochastic* process. For this to hold the infinite collections $\{Y(\boldsymbol{x})|\boldsymbol{x} \in \mathcal{X}\}$ needs to be measureable as well. Fortunately, Kolmogorov's extension theorem informs us that there *exists* a measurable stochastic process with marginals defined as the GP. It is in this light, that we can say that GPs provide distributions over function spaces. These distribution are then, as we have argued fully determined by their first two moments, which respectively is a mean function $\mu : \mathcal{X} \to \mathbb{R}$ and a covariance function $k : \mathcal{X} \times \mathcal{X} \to \mathbb{R}$. For this distribution over functions f we denote it

$$f \sim \mathcal{GP}(\mu, k), \tag{2.17}$$

and we say that f is a Gaussian process. For all our uses $\mathcal{X} = \mathbb{R}^d$. We will later look at the case when the *state space* is multidimensional, i.e. $\mathbb{R}^D$. But first, we will look at conditioning in Gaussian processes.

2.1.1 Conditioning and posterior distributions

Gaussian processes are distributions over functions, so this opens the possibility of giving them the Bayesian treatment. Naturally, we will then be interested in the posterior distribution of f. So now assume that our prior distribution of f is given $f(\boldsymbol{x}) \sim \mathcal{GP}\big(\mu(\boldsymbol{x}), k(\boldsymbol{x}, \boldsymbol{x}^*)\big)$ for well defined mean function $\mu : \mathbb{R}^d \to \mathbb{R}$ and valid covariance function $k : \mathbb{R}^d \times \mathbb{R}^d \to \mathbb{R}$.

Now suppose we have seen some data $\mathcal{D} = \{\boldsymbol{x}_i, f(\boldsymbol{x}_i)\}_{i=1}^N$. We will then be interested in the posterior of f, i.e. the distribution of $f|\mathcal{D}$. Fortunately, the Gaussian assumption makes this easy with the following theorem.

Theorem 2.3 *Let $\boldsymbol{X} \in \mathbb{R}^N$ and $\boldsymbol{Y} \in \mathbb{R}^M$ follow a joint multivariate Gaussian distribution*

$$\begin{pmatrix} \boldsymbol{X} \\ \boldsymbol{Y} \end{pmatrix} \sim \mathcal{N}\left(\begin{pmatrix} \boldsymbol{\mu}_X \\ \boldsymbol{\mu}_Y \end{pmatrix}, \begin{pmatrix} \boldsymbol{A} & \boldsymbol{C} \\ \boldsymbol{C}^\top & \boldsymbol{B} \end{pmatrix} \right), \tag{2.18}$$

then the conditional distribution $\boldsymbol{X}|\boldsymbol{Y}$ is given by

$$\boldsymbol{X}|\boldsymbol{Y} \sim \mathcal{N}\Big(\boldsymbol{\mu}_X + \boldsymbol{C}\boldsymbol{B}^{-1}(\boldsymbol{Y} - \boldsymbol{\mu}_Y), \boldsymbol{A} - \boldsymbol{C}\boldsymbol{B}^{-1}C^\top \Big). \tag{2.19}$$

This easily lets us compute the posterior. If we let $\boldsymbol{X} = (\boldsymbol{x}_1, \ldots, \boldsymbol{x}_N)^\top$ and $\boldsymbol{f} = f(\boldsymbol{X})$, then for any point $\boldsymbol{x}^* \in \mathbb{R}^d$ we have

$$f(\boldsymbol{x}^*)|\mathcal{D} \sim \mathcal{N}(\tilde{\mu}(\boldsymbol{x}^*), \tilde{\boldsymbol{K}}), \tag{2.20}$$

where

$$\tilde{\mu}(\boldsymbol{x}^*) = \mu(\boldsymbol{x}^*) + k(\boldsymbol{x}^*, \boldsymbol{X})k(\boldsymbol{X}, \boldsymbol{X})^{-1}(\boldsymbol{f} - \mu(\boldsymbol{X})), \tag{2.21}$$

$$\text{and} \qquad \tilde{\boldsymbol{K}} = k(\boldsymbol{x}^*, \boldsymbol{x}^*) - k(\boldsymbol{x}^*, \boldsymbol{X})k(\boldsymbol{X}, \boldsymbol{X})^{-1}k(\boldsymbol{x}^*, \boldsymbol{X})^\top, \tag{2.22}$$

where $k(\boldsymbol{a}, \boldsymbol{b})$ denotes and matrix of dimension numbers of rows in $\boldsymbol{a}$ times number of rows in $\boldsymbol{b}$. The entries are given $k(\boldsymbol{a}, \boldsymbol{b})_{ij} = k(\boldsymbol{a}_i, \boldsymbol{b}_j)$, i.e. the respective kernel evaluations.

As noted in the previous chapter, we need to specify a likelihood to specify a full statistical model. This complicates matters in obtaining the posterior, but there is one likelihood for which the Gaussian process is a conjugate prior, meaning the posterior belongs to the same family as the prior, i.e. the Gaussian family. It just so happens, that this likelihood is also the Gaussian, specifically given by

$$y_i \sim \mathcal{N}(f(\boldsymbol{x}_i), \epsilon), \quad \epsilon > 0. \tag{2.23}$$

This means our datasets is now $\mathcal{D} = \{\boldsymbol{x}_i, y_i\}_{i=1}^N$, and the statistical assumption is that y_i is obtained from $f(\boldsymbol{x_i})$ and some added Gaussian noise of size ϵ. Further, it is assumed that condition on f, the observation become independent. This means the likelihood function is

$$\mathcal{L}(\boldsymbol{y}|f, \epsilon) = \prod_{i=1}^{N} \mathcal{N}(y_i|f(\boldsymbol{x}_i), \epsilon). \tag{2.24}$$

As stated, under this likelihood the posterior of f is known, and the posterior mean and covariance, at some point $\boldsymbol{x}^*$, is given

$$\tilde{\mu}(\boldsymbol{x}^*) = \mu(\boldsymbol{x}^*) + k(\boldsymbol{x}^*, \boldsymbol{X})\big(k(\boldsymbol{X}, \boldsymbol{X}) + \epsilon \boldsymbol{I}_N\big)^{-1}(\boldsymbol{f} - \mu(\boldsymbol{X})), \tag{2.25}$$

$$\text{and} \qquad \tilde{\boldsymbol{K}} = k(\boldsymbol{x}^*, \boldsymbol{x}^*) - k(\boldsymbol{x}^*, \boldsymbol{X})\big(k(\boldsymbol{X}, \boldsymbol{X}) + \epsilon \boldsymbol{I}_N\big)^{-1} k(\boldsymbol{x}^*, \boldsymbol{X})^\top, \tag{2.26}$$

which is analytically tractable.

When the likelihood is not of the form (2.24) the posterior of f does not exist in closed form, and hence approximate Bayesian inference is necessary as described in Section 1.2.1. This is not the only issue with GP models: the computations in (2.21)-(2.22) have to deal with the inversion of $k(\boldsymbol{X}, \boldsymbol{X})$, which is of computational cost $\mathcal{O}(N^3)$.Computationally, this becomes intractable when N is large. In Section 2.1.3 we describe a framework that is computable also for large N. First, we will see how we define GPs when the state space is high dimensional.

2.1.2 Multi-output processes

This section investigates Gaussian processes with state space $\mathbb{R}^D$, where $D > 1$.

Definition 2.4 f is a multi-output Gaussian process, if $\text{vec}(f)$ is a Gaussian process.

Here $\text{vec}(\cdot)$ denotes the operation that takes a matrix $\boldsymbol{A}$ of size $N \times D$ and returns a column vector of size ND, by sequentially stacking the columns of $\boldsymbol{A}$. So if $\boldsymbol{F} =$

$(\boldsymbol{f}_1, \boldsymbol{f}_2, \ldots, \boldsymbol{f}_D)$, where $\boldsymbol{f}_d$ is a N-dimensional column vector for $1 \leq k \leq D$, then

$$\operatorname{vec}(\boldsymbol{F}) = \begin{pmatrix} \boldsymbol{f}_1 \\ \boldsymbol{f}_2 \\ \vdots \\ \boldsymbol{f}_D \end{pmatrix} \sim \mathcal{N}\left(\begin{pmatrix} \boldsymbol{\mu}_1 \\ \boldsymbol{\mu}_2 \\ \vdots \\ \boldsymbol{\mu}_D \end{pmatrix}, \boldsymbol{K} \right), \tag{2.27}$$

where $\boldsymbol{K}$ is a $ND \times ND$ symmetric and positive semi-definite matrix. In particular, all the marginals $\boldsymbol{f}_d$ are Gaussian processes.

The challenge with multi-output GPs is that it is not straightforward to have covariance functions that reflect inter-dimensional dependencies between the outputs. Hence, often $\boldsymbol{K}$ takes the trivial form

$$\boldsymbol{K} = \begin{pmatrix} \boldsymbol{K}_{11} & \boldsymbol{0} & \ldots & \boldsymbol{0} \\ \boldsymbol{0} & \boldsymbol{K}_{22} & & \vdots \\ \vdots & & \ddots & \vdots \\ \boldsymbol{0} & \ldots & \ldots & \boldsymbol{K}_{DD} \end{pmatrix}, \tag{2.28}$$

which makes it easy to choose covariance function on each of the marginals $\boldsymbol{f}_d$, but simultaneously this leaves the multi-output framework redundant as all outputs are independent. The task is to encode these dependencies through a prior on the multi-output GP f.

Coregionalization. Had we assumed all the marginals shared the same covariance function in (2.28), we could have written it simply as

$$\boldsymbol{K} = \boldsymbol{I}_D \otimes \boldsymbol{K}_{dd}, \tag{2.29}$$

where $\otimes$ denotes the *Kronecker product*. Now the inter-dimensional independence is immediately clear from the identity matrix $\boldsymbol{I}_D$. One nice feature of Kronecker products is, that the Kronecker product of two symmetric positive semi-definite matrices is again a symmetric positive semi-definite matrix. Thus, if we swap $\boldsymbol{I}_D$ in (2.29) with another symmetric positive semi-definite matrix $\boldsymbol{A}$, then $\boldsymbol{K}$ is still a covariance matrix. This is known as the *intrinsic coregionalization model* (ICM) [Goovaerts et al., 1997] and we write

$$\boldsymbol{K} = \boldsymbol{A} \otimes k(\boldsymbol{X}, \boldsymbol{X}), \tag{2.30}$$

where we switched back to the notation for kernel matrices. We can see the covariance of any two dimensions d and d^* and any two points $\boldsymbol{x}$ and $\boldsymbol{x}^*$

$$\operatorname{Cov}\big(f_d(\boldsymbol{x}), f_{d^*}(\boldsymbol{x}^*)\big) := a_{d,d^*} k(\boldsymbol{x}, \boldsymbol{x}^*), \tag{2.31}$$

where a_{d,d^*} is the (d, d^*)-th entry in $\boldsymbol{A}$. When the dimensionality D is large this is not a particularly flexible model and commonly we add some flexibility by summing kernels — recall that the sum of kernels is a kernel itself. A well-known model appears

if we let all the corresponding $D \times D$-matrices be of rank 1. In particular, we define

$$\boldsymbol{K} := \sum_{v=1}^{\nu} \boldsymbol{b}_v \boldsymbol{b}_v^\top \otimes k_v(\boldsymbol{X}, \boldsymbol{X}) = \sum_{v=1}^{\nu} (\boldsymbol{b}_v \boldsymbol{b}_v^\top \otimes \boldsymbol{I}_N) k_v(\boldsymbol{X}, \boldsymbol{X}) \tag{2.32}$$

$$= \sum_{v=1}^{\nu} (\boldsymbol{b}_v \otimes \boldsymbol{I}_N) k_v(\boldsymbol{X}, \boldsymbol{X}) (\boldsymbol{b}_v^\top \otimes \boldsymbol{I}_N) \tag{2.33}$$

$$= (\boldsymbol{B} \otimes \boldsymbol{I}_N) \tilde{\boldsymbol{K}} (\boldsymbol{B}^\top \otimes \boldsymbol{I}_N), \tag{2.34}$$

where $\boldsymbol{b}_v$ are the columns of $\boldsymbol{B}$ which is of size $D \times \nu$. $\tilde{\boldsymbol{K}}$ is the $N\nu \times N\nu$-block diagonal matrix with ν block entries $k_1(\boldsymbol{X}, \boldsymbol{X}), \ldots, k_\nu(\boldsymbol{X}, \boldsymbol{X})$. This is the covariance matrix from the *Semiparametric Latent Factor Model* (SLFM) [Seeger et al., 2005], which is the model defined

$$\begin{pmatrix} \boldsymbol{f}_1 \\ \boldsymbol{f}_2 \\ \vdots \\ \boldsymbol{f}_D \end{pmatrix} = \boldsymbol{B} \begin{pmatrix} \boldsymbol{u}_1 \\ \vdots \\ \boldsymbol{u}_\nu \end{pmatrix}, \quad \text{where} \quad \begin{pmatrix} \boldsymbol{u}_1 \\ \vdots \\ \boldsymbol{u}_\nu \end{pmatrix} \sim \mathcal{N}\left(\begin{pmatrix} \boldsymbol{\mu}_1 \\ \vdots \\ \boldsymbol{\mu}_\nu \end{pmatrix}, \tilde{\boldsymbol{K}} \right). \tag{2.35}$$

The latent factors are then $\boldsymbol{u}_1, \ldots, \boldsymbol{u}_\nu$ which are *independent* GPs with kernels $k_1, \ldots, k_\nu$. The semi-parametricity originates from the matrix $\boldsymbol{B}$, which makes the D GP outputs *dependent*.

2.1.3 Low-rank Variational Approximations

This section returns to the case where the output dimension is $D = 1$. We noted in (2.21)-(2.22) that computing the posterior mean and covariance involves inverting the kernel matrix $k(\boldsymbol{X}, \boldsymbol{X})$, which is of size $N \times N$. This operation has complexity $\mathcal{O}(N^3)$. As a single computation, this is doable for moderate sized datasets, say 5000-10000 datapoints, on standard machinery. However, often we wish to compute derivatives to optimise the hyperparameters of the kernel, thus we need to do this operation *many* times until convergence of the gradient optimiser, and the overhead of $\mathcal{O}(N^3)$ is computationally intractable.

Inducing points. Since exact inference is unattainable, we must turn to approximate methods. A key idea in this regard is *inducing* or *auxiliary* points. Let $\boldsymbol{f} = f(\boldsymbol{X})$ as usual, but consider M inducing points $\boldsymbol{u} = (u_1, u_2, \ldots, u_M)^\top$ taking values in $\mathbb{R}$, like the GP f. Along with $\boldsymbol{u}$ are the inducing *locations*, which we will denote $\boldsymbol{Z} = (\boldsymbol{z}_1, \boldsymbol{z}_2, \ldots, \boldsymbol{z}_M)^\top$ that takes their values in the same space as $\boldsymbol{X}$, say $\mathbb{R}^d$.

One of the first ideas in this regime was the *Deterministic Training Conditional approximation* (DTC) [Seeger et al., 2003], which use an approximate training conditional

$$q(\boldsymbol{f}|\boldsymbol{u}) = \mathcal{N}(\boldsymbol{\alpha}(\boldsymbol{X})^\top \boldsymbol{u}, 0), \tag{2.36}$$

where $\boldsymbol{\alpha}(\cdot)^\top = k(\cdot, \boldsymbol{Z}) k(\boldsymbol{Z}, \boldsymbol{Z})^{-1}$. This of course, as the name also suggests, that everything becomes deterministic when conditioned on $\boldsymbol{u}$. From the computational

viewpoint, we note that now we only need to invert $k(\boldsymbol{Z}, \boldsymbol{Z})$ which reduces the operation to $\mathcal{O}(M^3)$, where $M << N$. We can then obtain the posterior over $\boldsymbol{u}$ — often necessary to assume $\boldsymbol{u}$ is a subset of the original training set $\boldsymbol{f}$ — and at test points $\boldsymbol{f}^*$ compute the GP

$$q(\boldsymbol{f}^*|\boldsymbol{u}) = \mathcal{N}\Big(\boldsymbol{\alpha}(\boldsymbol{x}^*)^\top \boldsymbol{u}, k(\boldsymbol{x}^*, \boldsymbol{x}^*) - \boldsymbol{\alpha}(\boldsymbol{x}^*)^\top k(\boldsymbol{Z}, \boldsymbol{Z})\boldsymbol{\alpha}(\boldsymbol{x}^*)\Big). \tag{2.37}$$

We can even marginalise the conditional $\boldsymbol{u}$ if the approximate posterior $q(\boldsymbol{u}) = \mathcal{N}(\boldsymbol{\mu}, \boldsymbol{\Sigma})$ (the prior on $\boldsymbol{u}$ is $\mathcal{N}(\boldsymbol{0}, k(\boldsymbol{Z}, \boldsymbol{Z}))$ as

$$q(\boldsymbol{f}^*) = \mathcal{N}\Big(\boldsymbol{\alpha}(\boldsymbol{x}^*)^\top \boldsymbol{\mu}, k(\boldsymbol{x}^*, \boldsymbol{x}^*) - \boldsymbol{\alpha}(\boldsymbol{x}^*)^\top (k(\boldsymbol{Z}, \boldsymbol{Z}) - \boldsymbol{\Sigma})\boldsymbol{\alpha}(\boldsymbol{x}^*)\Big). \tag{2.38}$$

Snelson and Ghahramani [2006] build upon this by removing the deterministic condition, but maintaining the conditioned on $\boldsymbol{u}$, the marginals of $\boldsymbol{f}$ are independent. They match the marginal variance $\mathrm{Var}(f_i|\boldsymbol{u})$ with the exact marginal variances $k(\boldsymbol{x}_i, \boldsymbol{x}_i) - \boldsymbol{\alpha}(\boldsymbol{x}_i)^\top k(\boldsymbol{Z}, \boldsymbol{Z})\boldsymbol{\alpha}(\boldsymbol{x}_i)$. Thus, the approximate posterior is

$$q(\boldsymbol{f}|\boldsymbol{u}) = \prod_{i=1}^{N} \mathcal{N}\Big(\boldsymbol{\alpha}(\boldsymbol{x}_i)^\top \boldsymbol{u}, k(\boldsymbol{x}_i, \boldsymbol{x}_i) - \boldsymbol{\alpha}(\boldsymbol{x}_i)^\top k(\boldsymbol{Z}, \boldsymbol{Z})\boldsymbol{\alpha}(\boldsymbol{x}_i)\Big). \tag{2.39}$$

and this is convenient since now $\boldsymbol{u}$ does not have to be a subset of $\boldsymbol{f}$. This can be obtained by performing *exact* inference in the modified GP with the kernel

$$\tilde{k}(\boldsymbol{x}_i, \boldsymbol{x}_j) = \boldsymbol{\alpha}(\boldsymbol{x}_i)^\top k(\boldsymbol{Z}, \boldsymbol{Z})\boldsymbol{\alpha}(\boldsymbol{x}_j) + \delta_{ij}\Big(k(\boldsymbol{x}_i, \boldsymbol{x}_j) - \boldsymbol{\alpha}(\boldsymbol{x}_i)^\top k(\boldsymbol{Z}, \boldsymbol{Z})\boldsymbol{\alpha}(\boldsymbol{x}_j)\Big), \tag{2.40}$$

where δ_{ij} is Kronecker's delta, i.e. $\delta_{ij} = 1$, if $i = j$, and 0 else.

The last option for approximating the true posterior with low-rank kernel matrices is to tackle the problem in the framework of variational inference (introduced in Section 1.2.1). In the context of GPs this was introduced by Titsias [2009a]. Thus we approximate the true posterior $p(\boldsymbol{f}, \boldsymbol{u}|\boldsymbol{y})$ with an approximate distribution $q(\boldsymbol{f}, \boldsymbol{u})$. Here $\boldsymbol{y}$ denotes observations. The reason we never considered $\boldsymbol{y}$ in the two previous methods is they only work when the likelihood is Gaussian, i.e. $y_i \sim \mathcal{N}(f_i, \epsilon)$. The variational approach works for *any* choice of likelihood function. Recall from Section 1.2.1 that minimising KL between $q(\boldsymbol{f}, \boldsymbol{u})$ and $p(\boldsymbol{f}, \boldsymbol{u}|\boldsymbol{y})$ is equivalent to maximising the evidence lower bound

$$\mathcal{L}(q) = \mathbb{E}_q[\log p(\boldsymbol{y}|\boldsymbol{f}, \boldsymbol{u})] - \mathrm{KL}(q(\boldsymbol{f}, \boldsymbol{u})\|p(\boldsymbol{f}, \boldsymbol{u})). \tag{2.41}$$

If we choose the approximate posterior to take the form

$$q(\boldsymbol{f}, \boldsymbol{u}) = p(\boldsymbol{f}|\boldsymbol{u})q(\boldsymbol{u}), \tag{2.42}$$

where $q(\boldsymbol{u}) = \mathcal{N}(\boldsymbol{m}, \boldsymbol{S})$, $\boldsymbol{m}$ is a free-form parametric vector of length M, and $\boldsymbol{S}$ is a symmetric and positive semi-definite matrix of size $M \times M$ — also parametric.

In this setup, the evidence lower bound reduces to

$$\mathcal{L}(q) = \mathbb{E}_q(\boldsymbol{f})[\log p(\boldsymbol{y}|\boldsymbol{f})] - \mathrm{KL}(q(\boldsymbol{u})\|p(\boldsymbol{u})). \tag{2.43}$$

Moreover, the approximate posterior where we marginalise the inducing points $\boldsymbol{u}$ takes the form

$$q(\boldsymbol{f}^*) = \mathcal{N}\Big(\boldsymbol{\alpha}(\boldsymbol{x}^*)^\top \boldsymbol{m}, k(\boldsymbol{x}^*, \boldsymbol{x}^*) - \boldsymbol{\alpha}(\boldsymbol{x}^*)^\top (k(\boldsymbol{Z}, \boldsymbol{Z}) - \boldsymbol{S})\boldsymbol{\alpha}(\boldsymbol{x}^*)\Big), \tag{2.44}$$

for $\boldsymbol{f}^* = f(\boldsymbol{x}^*)$.

2.2 Wishart Processes

The *Wishart distribution* generalises the univariate χ^2-distribution to symmetric, positive semi-definite matrices. We can recall the χ^2 is built from squared unit Gaussians

$$\sigma^2 := \sum_{v=1}^{\nu} U_v^2, \tag{2.45}$$

where $U_v \sim \mathcal{N}(0, 1)$ and all independent. Then we say $\sigma^2 \sim \chi^2(\nu)$ — in words, σ^2 is χ^2-distributed with ν degrees of freedom. The extension to matrices relies on the same principles, we change the univariate Gaussians to multivariate, that is

$$\boldsymbol{\Sigma} := \sum_{v=1}^{\nu} \boldsymbol{U}_v \boldsymbol{U}_v^\top, \tag{2.46}$$

where now $\boldsymbol{U}_v$ are independent and distributed as $\mathcal{N}(\boldsymbol{0}, \boldsymbol{I}_D)$. We say that $\boldsymbol{\Sigma}$ is Wishart distribution with ν degrees of freedom and scale matrix $\boldsymbol{I}_D$. The scale matrix is also the expectation, i.e. $\mathbb{E}[\boldsymbol{\Sigma}] = \boldsymbol{I}_D$. In this view we can introduce some parametric matrices to change this expectation, we can write

$$\boldsymbol{\Sigma} := \sum_{v=1}^{\nu} \boldsymbol{L}\boldsymbol{U}_v \boldsymbol{U}_v^\top \boldsymbol{L}^T, \tag{2.47}$$

where $\boldsymbol{L}\boldsymbol{L}^\top$ is a symmetric positive semi-definite $D \times D$ matrix. Now the scale matrix is $\boldsymbol{L}\boldsymbol{L}^\top$, and for short hand notation we write $\boldsymbol{\Sigma} \sim \mathcal{W}_D(\boldsymbol{L}\boldsymbol{L}^\top, \nu)$. $\boldsymbol{\Sigma}$ is degenerate if $\nu < D$.

It is on this foundation that we define Wishart processes. This is near-identical to the definition by Wilson and Ghahramani [2010].

Definition 2.5 Let $\boldsymbol{L}$ be a $D \times D$ matrix, such that $\boldsymbol{L}\boldsymbol{L}^\top$ is positive semi-definite and $f_{d,v} \sim \mathcal{GP}\big(0, k_{d,v}(\boldsymbol{x}, \boldsymbol{x}')\big)$ independently for every $d = 1, \ldots, D$ and $v = 1 \ldots, \nu$, where $\nu \geq D$. Then if

$$\Sigma(\boldsymbol{x}) = \boldsymbol{L}\left(\sum_{v=1}^{\nu} \boldsymbol{f}_v(\boldsymbol{x})\boldsymbol{f}_v^\top(\boldsymbol{x})\right)\boldsymbol{L}^\top \tag{2.48}$$

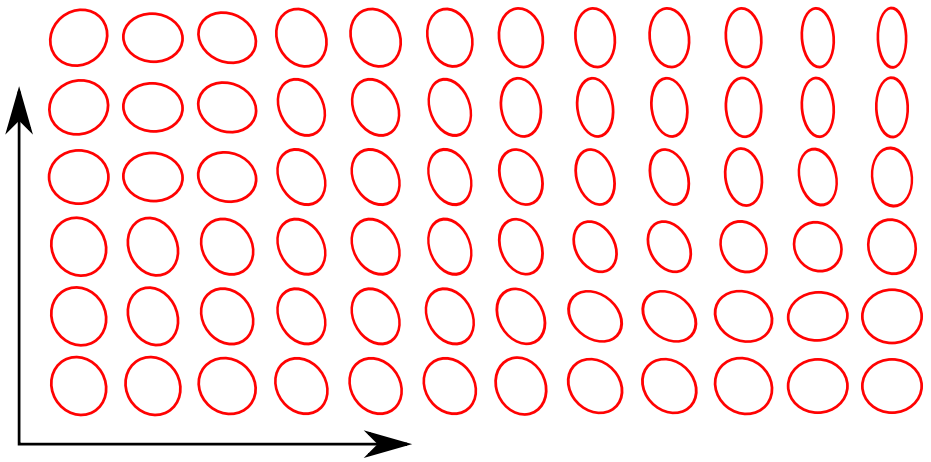


Figure 2.1: Visualisation of the Wishart process. The input dimensionality, d, is 2, and varies along the coordinate system indicated by the black arrows. The output is a 2×2 symmetric matrix, which we can visualise as ellipses. The diagonal elements are represented by the ellipses length along the coordinates system. The off-diagonal element is the rotation of the ellipse. We can see how these change over the coordinate system.

is Wishart distributed for any marginal $\boldsymbol{x}$, and if for any finite collection of points $\boldsymbol{X} = \{\boldsymbol{x}_i\}_{i=1}^N$ the joint distribution $\Sigma(\boldsymbol{X})$ is determined through the covariance functions $k_{d,v}$, then $\Sigma(\cdot)$ is a Wishart process. We will write

$$\Sigma \sim \mathcal{WP}_D(\boldsymbol{L}\boldsymbol{L}^\top, \nu, \kappa), \tag{2.49}$$

where κ is the collection of covariance functions $\{k_{d,v}\}$.

Wilson and Ghahramani [2010] call this the *Generalised Wishart Process* as it is defined for general kernel function as opposed to the construction from Bru [1991].

Like the *coreginalization* from Section 2.1.2 this construction is semi-parametric — at least when the scale matrix is not the identity. Often, we change the notation and write $\boldsymbol{\Sigma}(\boldsymbol{x}) = \boldsymbol{L}\boldsymbol{F}\boldsymbol{F}^\top\boldsymbol{L}^\top$, where $\boldsymbol{F} = (\boldsymbol{f}_1, \boldsymbol{f}_2, \ldots, \boldsymbol{f}_\nu)$, i.e. the matrix with all independent entries $f_{d,v}$. It is this construction that makes it easy to perform (approximate) Bayesian inference, as it reduces to performing the inference in the underlying GPs. Of course, $\boldsymbol{L}$ also needs to be inferred, but we will not consider priors over it.

In view of this, there is a lingual caveat in this framework. If we say that Σ has a Wishart process prior as determined by Definition 2.5, then the *posterior* process need not be marginally Wishart. Notice the condition that the underlying GPs have

constant mean function 0, is not fulfilled by the posterior GPs. Retrospectively, a more suitable name for this family of processes would have been *non-central* Wishart processes. We remark however that we still refer to posterior processes as Wishart processes — a Wishart process is a process whose prior is a Wishart process.

Figure 2.1 visualise how we can think of Wishart processes over the index set $\mathbb{R}^2$. The outputs are 2×2-matrices which we can illustrate by ellipses. The rotation of the ellipses corresponds to the correlation (i.e. Σ_{12}), and the marginal variances — the diagonal of $\boldsymbol{\Sigma}$ — are visualised as the 'diameter' of the ellipse along the D coordinate axes. We see how the ellipses can change for different inputs.

We end this section of by introducing a little trick from Wishart distributions, that also applies to Wishart processes. Let $\boldsymbol{\Sigma}$ follow a ρ-Wishart distribution with ν degrees of freedom and scale matrix $\boldsymbol{L}\boldsymbol{L}^\top$, and let $\boldsymbol{B}$ be a $D \times \rho$-matrix of full rank. Then $\boldsymbol{B}\boldsymbol{\Sigma}\boldsymbol{B}^\top \sim \mathcal{W}_D(\boldsymbol{B}\boldsymbol{L}\boldsymbol{L}^\top\boldsymbol{B}^\top, \nu)$. We will use this feature to scale Wishart processes to high dimensions, thus we will let $\rho < D$. We remark here the resemblance to Semi-parametric Latent Factor Model as introduced in (2.35). It is a 'squared' SLFM, since

$$\boldsymbol{\Sigma}(\boldsymbol{x}) = \boldsymbol{B}\boldsymbol{L}\boldsymbol{F}\boldsymbol{F}^\top\boldsymbol{L}^\top\boldsymbol{B}^\top, \tag{2.50}$$

where $\boldsymbol{F}$ is the $\rho\nu$ latent factors. This is computationally useful, as we only have to perform inference[1] in ρ^2 GPs, as opposed to D^2. The cost of this is that the resulting Wishart process is degenerate.

2.3 Stochastic Differential Equations

Definition 2.6 (Brownian Motion) Let $\sigma^2 > 0$. A (univariate) stochastic process, $B : [0, \infty) \to \mathbb{R}$, is called a Brownian motion if it satisfies

(i) For any $0 \leq s_1 < t_1 \leq s_2 < t_2$, we have $B(t_1 - s_1)$ is independent of $B(t_2 - s_2)$.
(ii) $B(t - s) \sim \mathcal{N}\big(0, \sigma^2(t - s)\big)$, for any $0 \leq s < t$.
(iii) $t \mapsto B(t)$ is almost surely continuous.

Remark. The Brownian motion (BM) is a Gaussian process with constant mean function 0 and covariance function $k(s, t) = \sigma^2 \min(s, t)$. Further, as a consequence of (ii) and (iii) we always have $B(0) = 0$ almost surely. In the literature, the Brownian motion is often also called the Wiener process. The BM is nowhere differentiable.

The stochastic differential equations (SDE) [Särkkä and Solin, 2019, Kloeden and Platen, 2013] in this section are of the form

$$x(t) = x(0) + \int_0^t a(x(s))ds + \int_0^t \sqrt{b(x(s))}dB(s), \tag{2.51}$$

[1] We here always assume the degrees of freedom is equal to the 'new' output dimension ρ.

where $a : \mathbb{R} \to \mathbb{R}$ is called the *drift* function, $b : \mathbb{R} \to \mathbb{R}$ is called the *diffusion*. B is a Brownian motion. In the literature, the parameter σ^2 of the BM is sometimes referred to as the diffusion, but we remark this can be accounted for in b. A stochastic process that satisfies (2.51) is said to be an *Itô process*, named after Kiyoshi Itô — a fundamental figure in the development of stochastic calculus.

They are a continuous time version of *state-space* models (remember e.g. the random walk from Example 2.1), that are generated like[2]

$$x_0 = x, \qquad \text{for some } x \in \mathbb{R}, \tag{2.52}$$
$$x_t = x_{t-1} + a(x_{t-1}) + b(x_{t-1})\epsilon_t, \qquad \text{for } t \in \mathbb{N} \tag{2.53}$$
$$y_t = g(x_t), \qquad \text{for } t \in \mathbb{N} \tag{2.54}$$

where ϵ_t is some random noise — we will assume it unit Gaussian. Here y_t denotes temporal observations and x_t is a latent state. The dynamics of x_t are as given in (2.53). When g is the identity mapping we often call it an auto-regressive model.

We are interested in these types of model for high-dimensional state spaces. We can define a random walk type model on a (multi-output) Gaussian field $f \sim \mathcal{GP}(\mu, \Sigma)$

$$\boldsymbol{x}_{t+\Delta} = \boldsymbol{x}_t + \mu(\boldsymbol{x}_t)\Delta + \sqrt{\boldsymbol{\Sigma}(\boldsymbol{x}_t)\Delta}\boldsymbol{N}, \quad \boldsymbol{N} \sim \mathcal{N}(\boldsymbol{0}, \boldsymbol{I}_D), \tag{2.55}$$

for any $t > 0$ and any $\Delta > 0$. Then the state $\boldsymbol{x}_T$ is given like

$$\boldsymbol{x}_T = \boldsymbol{x}_0 + \sum\nolimits_{t=0}^{\zeta-1} \left(\mu(\boldsymbol{x}_t)\Delta + \sqrt{\boldsymbol{\Sigma}(\boldsymbol{x}_t)\Delta}\boldsymbol{N}\right). \tag{2.56}$$

where $\Delta := T/\zeta$. If we let $\zeta \to \infty$ we obtain the SDE

$$\boldsymbol{x}_T - \boldsymbol{x}_0 = \int_0^T \mu(\boldsymbol{x}_t)dt + \int_0^T \sqrt{\boldsymbol{\Sigma}(\boldsymbol{x}_t)}dB_t, \tag{2.57}$$

where B_t is the multivariate BM with increments $B(t-s) \sim \mathcal{N}(\boldsymbol{0}, (t-s)\boldsymbol{I}_D)$.

Now the task is to infer the posterior field f based on its prior and observations, potentially multivariate, under the model assumption (2.54).

2.4 Continuous-time models in Machine Learning

Differential equations, in general, have recently attracted vast interest in machine learning [Haber and Ruthotto, 2017, E, 2017, Chen et al., 2018]. The connection to time series modelling is evident, however the recent interest stems from their connection to deep neural networks. *ResNets* [He et al., 2016] is a family of network

[2]We now write $x_t = x(t)$ for easier notation

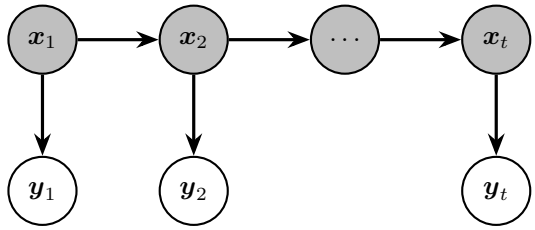


Figure 2.2: The standard state-space model. The temporal dependence is modelled as a Markov chain in the *latent* state space $\boldsymbol{x}_t$, and the observed values $\boldsymbol{y}_t$ are independent condtioned on $\boldsymbol{x}_t$.

architectures that allow stable training of *very* deep networks. In words, the central idea is not to overwrite what previous layers have learned, by having the layers in the network form *residual blocks*. This means that layers $\boldsymbol{l}_t$ of the network take the form

$$\boldsymbol{l}_t = \boldsymbol{l}_{t-1} + f(\boldsymbol{l}_t, \theta_t), \tag{2.58}$$

where f is some transformation parametrized with θ_t. These building blocks are more robust to the vanishing gradient problem, as we could 'turn off' the transformations $f \approx \mathbf{0}$ and maintaining the information from earlier layers.

These blocks in (2.58) resembles the structure in (2.55) — there we just added some random noise in each transformation. One could then ask: what if we took infinitely many infinitely small steps or transformations? In other words, we are envisioning (2.58) as a coarse Euler-discretization [Kloeden and Platen, 2013] of an ODE

$$\frac{\partial \boldsymbol{l}_t}{\partial t} = f(\boldsymbol{l}_t, \theta_t, t). \tag{2.59}$$

Chen et al. [2018] introduced a method to optimise this ODE without backpropagating through the discretization.

For regression tasks, the generative model can then be written, for non-temporal data $\mathcal{D} = \{\boldsymbol{x}_i, y_i\}_{i=1}^N$, like

$$\boldsymbol{x}_0 = \boldsymbol{x}_i, \tag{2.60}$$

$$\boldsymbol{x}_T = \boldsymbol{x}_0 + \int_0^T f(\boldsymbol{l}_t, \theta_t, t)dt, \tag{2.61}$$

$$y_i \sim \mathcal{N}(g(\boldsymbol{x}_T), \epsilon), \tag{2.62}$$

for some predefined $T > 0$ and any $i = 1, \ldots, N$. This implies, that what for usual neural networks is referred to as depth, the discrete number of hidden layers, is now a continuous quantity T.

We will focus on dynamics with noise in them, i.e. SDEs. The solver we shall use is Euler-Maruyama's method, the SDE-eqivalent to Euler's discretization [Kloeden and

Platen, 2013]. This method finely discretizes the interval $[0, T]$ in a mesh $0 = t_0 < t_1 < \ldots < t_l = T$, and *pushes* $\boldsymbol{x}_{t_i}$ along the vector field

$$\boldsymbol{x}_{t_{i+1}} = \boldsymbol{x}_{t_i} + \boldsymbol{\mu}(\boldsymbol{x}_{t_i})\Delta_i + \sqrt{\boldsymbol{\Sigma}(\boldsymbol{x}_{t_i})\Delta_i}\boldsymbol{N}, \tag{2.63}$$

where $\boldsymbol{N} \sim \mathcal{N}(\boldsymbol{0}, \boldsymbol{I}_D)$ and $\Delta_i = t_{i+1} - t_i$. We 'generalise' the Neural ODE model to SDEs; this has been studied in Twomey et al. [2019], Tzen and Raginsky [2019], Liu et al. [2019], Li et al. [2020], and the work by Andreas and Kandemir [2019], who model the drift and diffusion of an SDE with Bayesian neural networks.

Differential Gaussian Process Flows. The deep learning equivalent in the GP literature is *deep Gaussian processes*. This notion covers multiple constructions [Dunlop et al., 2018], but here a deep Gaussian process (DGP) is a sequence of conditional GPs such that

$$f_l | f_{l-1} \sim \mathcal{GP}\big(\mu_l(f_{l-1}), k_l(f_{l-1}, f_{l-1})\big), \tag{2.64}$$

where l_0 is the input data. In words, the outputs of one GP are the inputs of the next. This construction has been studied by Damianou and Lawrence [2013], Cutajar et al. [2017], Dai et al. [2015] and efficient inference was given by Salimbeni and Deisenroth [2017] to scale these models to large datasets.

While neural nets have vanishing gradient problems, these GP compositions have a more fundamental problem. For most kernels the intermediate GPs are not bijective, which means two different inputs is likely to collapse to the same input for very deep models. To overcome this issue Salimbeni and Deisenroth [2017] propose to have the prior mean of each layer be the identity — in contrast to the usual constant mean. Another solution was proposed by Duvenaud et al. [2014], where each layer is a GP with input from the previous layer and l_0 concatenated.

Hegde et al. [2019] proposed the 'infinite' limit of DGPs that views the sequence of GPs as a coarse Euler-Maruyama discretization. They let the SDE in (2.63) be parametrized with the mean function an covariance matrix of *one* Gaussian field. For completion we present the graphical and generative model here, but most detail are covered in the next section; the graphical model is given in Figure 2.3 and generative model reads as

$$\boldsymbol{x}_0 = \boldsymbol{x}_i, \tag{2.65}$$

$$\boldsymbol{x}_T = \boldsymbol{x}_0 + \int_0^T \mu(\boldsymbol{x}_t)dt + \int_0^T \sqrt{\boldsymbol{K}(\boldsymbol{x}_t)}dB_t, \tag{2.66}$$

$$y_i \sim \mathcal{N}(g(\boldsymbol{x}_T), \epsilon), \tag{2.67}$$

where $\boldsymbol{K}(\boldsymbol{x})$ is the $ND \times ND$ covariance matrix of the posterior of the Gaussian field f. μ is the posterior mean function. The same type of model was studied by Ustyuzhaninov et al. [2020].

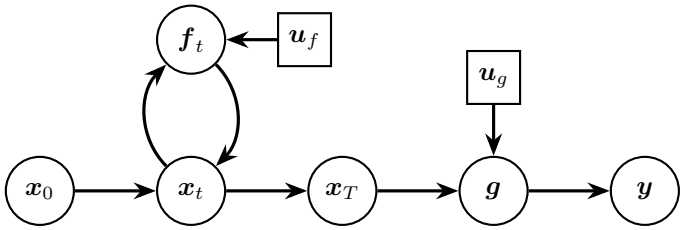


Figure 2.3: The graphical model of Hegde et al. [2019].

2.5 Diffusions with a Wishart prior

In this section, we will introduce the model presented by Jørgensen, Deisenroth, and Salimbeni [2020]. The model is inspired by the model in Hegde et al. [2019], which was briefly summarised in the previous section. A caveat in the model there is the diffusion term. In their experiments they let the $K(\boldsymbol{x}_t)$ in (2.66) be a $ND \times ND$ diagonal matrix — this means that all points move independently also among output dimensions. In this way, (2.66) could have been written as ND independent SDEs. From a regression perspective this is not a big issue, but it is not intuitive for a dynamical system. In temporal processes, often two (or many) variables *evolve* in a correlated manner, and it is this kind of behaviour we aim to capture in this section. In temporal modelling these models have been studied and used intensively in many fields. Among them is the award-winning ARCH model [Engle, 1982], and it's successors, which learn the diffusion in an auto-regressive way.

Hegde et al. [2019] only focus on regression and classification, and achieve positive results compared to other DGP models. It is still interesting to try to understand if the 'uncertainty' propagated through an SDE is comparable to the uncertainty of posterior GPs, i.e. $k(\boldsymbol{x}_i, \boldsymbol{x}_i)$. From a regression viewpoint, the diffusion can be seen as a regulariser, and it is interesting if this can also be learned in a Bayesian manner.

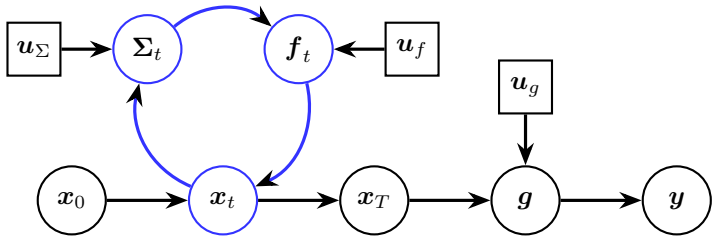
The model we present consists of a Gaussian field $f : \mathbb{R}^D \times [0, T] \to \mathbb{R}^D$ and a GP $g : \mathbb{R}^D \to \mathbb{R}^\eta$, whose priors are

$$f \sim \mathcal{GP}(0, k_f(\cdot, \cdot) \otimes \boldsymbol{I}_D), \quad g \sim \mathcal{GP}(0, k_g(\cdot, \cdot) \otimes \boldsymbol{I}_\eta). \tag{2.68}$$

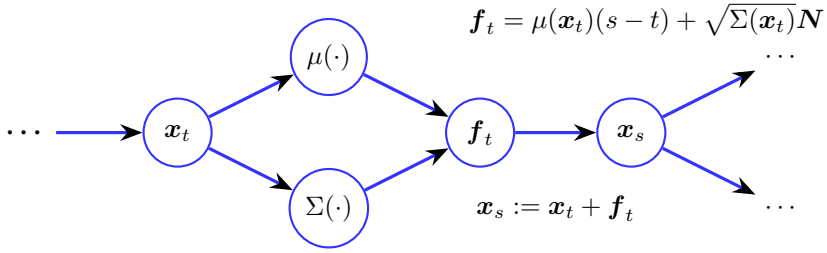
We also consider a Wishart process $\Sigma : \mathbb{R}^D \times [0, T] \to \mathcal{G}$, where $\mathcal{G}$ denotes the set of symmetric, positive semi-definite $D \times D$ matrices.

Remark. For easier notation we will write $k^D(\boldsymbol{a}, \boldsymbol{b}) := k(\boldsymbol{a}, \boldsymbol{b}) \otimes \boldsymbol{I}_D$. That is, $k(\boldsymbol{a}, \boldsymbol{b})$ returns a kernel matrix of dimension number of rows in $\boldsymbol{a}$ times the number of rows in $\boldsymbol{b}$. This corresponds to the assumption of independence between output dimensions.

Before detailing the model, we present the overview here with a graphical model (see



(a) Graphical model based on Eq. (2.73)



(b) Cycle from (a) and how it moves along the time-axis.

Figure 2.4: (a) Graphical model based on the factorisation in Eq. (2.73); (b) The cycle from (a), which represents the *field* f, and how it moves along the time-axis. Here $\boldsymbol{N} \sim \mathcal{N}(\boldsymbol{0}, (s-t)\boldsymbol{I})$. Blue represents the flow/SDE, square nodes are variational variables.

Figure 2.4) and a generative model that reads

$$\boldsymbol{x}_0 = \boldsymbol{x}_i, \tag{2.69}$$

$$\Sigma \sim \mathcal{WP}_D(\boldsymbol{L}\boldsymbol{L}^\top, \nu, \kappa), \quad f(\cdot)|\Sigma \sim \mathcal{GP}\big(\mu(\cdot), \boldsymbol{\Sigma}(\cdot)\big), \tag{2.70}$$

$$\boldsymbol{x}_T = \boldsymbol{x}_0 + \int_0^T \mu(\boldsymbol{x}_t)dt + \int_0^T \sqrt{\boldsymbol{\Sigma}(\boldsymbol{x}_t)}dB_t, \tag{2.71}$$

$$\boldsymbol{y}_i \sim \mathcal{N}(g(\boldsymbol{x}_T), \boldsymbol{A}\boldsymbol{\Sigma}(\boldsymbol{x}_t)\boldsymbol{A}^\top + \boldsymbol{\Lambda}), \tag{2.72}$$

where the fields in (2.70) denotes the *prior* fields. Next, we will show how to marginalise these. Here $\boldsymbol{A}$ is a $\eta \times D$ parametric matrix and $\boldsymbol{\Lambda}$ is a diagonal $\eta \times \eta$ parametric matrix. If $\eta = 1$, then usually $\boldsymbol{A} = \boldsymbol{0}$ and the model reduces to $y_i \sim \mathcal{N}(g(\boldsymbol{x}_T), \epsilon)$ for some $\epsilon > 0$.

This construction is a continuous-time deep learning model capable of propagating noise in high-dimensions. In words, the diffusion coefficient $\boldsymbol{\Sigma}(\boldsymbol{x}_t)$ of the SDE has Wishart process prior. The model we present factorises as

$$p(\boldsymbol{y}, \Theta) = p(\boldsymbol{y}|\boldsymbol{g})p(\boldsymbol{g}|\boldsymbol{x}_T, \boldsymbol{u}_g)p(\boldsymbol{u}_g)p(\boldsymbol{x}_T|\boldsymbol{f})p(\boldsymbol{f}|\boldsymbol{\Sigma}, \boldsymbol{u}_f)p(\boldsymbol{u}_{\boldsymbol{f}})p(\boldsymbol{\Sigma}|\boldsymbol{u}_\Sigma)p(\boldsymbol{u}_\Sigma), \tag{2.73}$$

where $\Theta := \{\boldsymbol{g}, \boldsymbol{u}_g, \boldsymbol{x}_T, \boldsymbol{f}, \boldsymbol{u}_f, \boldsymbol{\Sigma}, \boldsymbol{u}_\Sigma\}$ denotes the variables to be marginalised. We

assume data $\mathcal{D} = \{(\boldsymbol{x}_i, \boldsymbol{y}_i)\}_{i=1}^N$ is i.i.d. given the process, such that

$$p(\boldsymbol{y}|\boldsymbol{g}) = \prod_{i=1}^N p(\boldsymbol{y}_i|\boldsymbol{g}_i). \tag{2.74}$$

We approximate the posterior of g with the variational distribution

$$q(\boldsymbol{g}_i) = \int p(\boldsymbol{g}_i|\boldsymbol{u}_g)q(\boldsymbol{u}_g)d\boldsymbol{u}_g = \mathcal{N}(\tilde{\mu}_g(\boldsymbol{x}_i), \tilde{k}_g(\boldsymbol{x}_i, \boldsymbol{x}_i)), \tag{2.75}$$

where

$$\tilde{\mu}_g(\boldsymbol{x}_i) = \boldsymbol{\alpha}_g^\top(\boldsymbol{x}_i)\mathrm{vec}(\boldsymbol{m}_g), \tag{2.76}$$

$$\tilde{k}_g(\boldsymbol{x}_i, \boldsymbol{x}_i) = k_g^\eta(\boldsymbol{x}_i, \boldsymbol{x}_i) - \boldsymbol{\alpha}_g^\top(\boldsymbol{x}_i)\big(k_g^\eta(\boldsymbol{Z}_g, \boldsymbol{Z}_g) - \boldsymbol{S}_g\big)\boldsymbol{\alpha}_g(\boldsymbol{x}_i), \tag{2.77}$$

where $\boldsymbol{\alpha}_g(\boldsymbol{x}_i) := k_g^\eta(\boldsymbol{x}_i, \boldsymbol{Z}_g)k_g^\eta(\boldsymbol{Z}_g, \boldsymbol{Z}_g)^{-1}$. Here $\boldsymbol{m}_g$ is an $M \times \eta$ matrix, and $\boldsymbol{S}_g$ is an $M\eta \times M\eta$-matrix, constructed as η different $M \times M$-matrices $\boldsymbol{S}_g = \{\boldsymbol{S}_j\}_j^\eta$.

The inputs to g are the state distribution $\boldsymbol{x}_T$ of an SDE at a fixed time point $T \geq 0$. We construct this SDE from the viewpoint of a random field. The SDE is

$$\boldsymbol{x}_T - \boldsymbol{x}_0 = \int_0^T \mu(\boldsymbol{x}_t)dt + \int_0^T \sqrt{\boldsymbol{\Sigma}(\boldsymbol{x}_t)}dB_t, \tag{2.78}$$

where B is a Brownian motion. Thus $\boldsymbol{x}_t$ is an Itô process, which we numerically can solve by the Euler-Maruyama method. We will see that by a particular choice of variational distribution that $\boldsymbol{\Sigma}(\boldsymbol{x}_t)$ will be the realisation of a Wishart process. But first we consider no prior over the diffusion, as in Hegde et al. [2019].

The coefficients in (2.78) are determined as the mean and covariance of a Gaussian field f. The posterior of f is approximated with a Gaussian $q(\boldsymbol{f}_i) = \mathcal{N}(\tilde{\mu}_f(\boldsymbol{x}_i), \tilde{k}_f(\boldsymbol{x}_i, \boldsymbol{x}_i))$, where

$$\tilde{\mu}_f(\boldsymbol{x}_i) = \boldsymbol{\alpha}_f^\top(\boldsymbol{x}_i)\mathrm{vec}(\boldsymbol{m}_f), \tag{2.79}$$

$$\tilde{k}_f(\boldsymbol{x}_i, \boldsymbol{x}_i) = k_f^D(\boldsymbol{x}_i, \boldsymbol{x}_i) - \boldsymbol{\alpha}_f^\top(\boldsymbol{x}_i)\big(k_f^D(\boldsymbol{Z}_f, \boldsymbol{Z}_f) - \boldsymbol{S}_f\big)\boldsymbol{\alpha}_f(\boldsymbol{x}_i), \tag{2.80}$$

and $\boldsymbol{\alpha}_f(\cdot) = k_f^D(\cdot, \boldsymbol{Z}_f)k_f^D(\boldsymbol{Z}_f, \boldsymbol{Z}_f)^{-1}$.

Summarising, we have a dynamic propagating a data point $\boldsymbol{x}_0$ through the SDE (2.78) to $\boldsymbol{x}_T$, and further through the GP g, to make a prediction. However, each coordinate of $\boldsymbol{x}$ move independently, by construction of the prior f. By introducing Wishart processes, this assumption is eliminated.

Wishart Diffusions. Still considering the Gaussian field f, whose posterior is approximated by the variational distribution $q(\boldsymbol{f})$. Remaining within the Bayesian variational framework, we define a hierarchical model:

$$p(\boldsymbol{f}) = \int p(\boldsymbol{f}|\boldsymbol{u}_f, \boldsymbol{\Sigma})p(\boldsymbol{u}_f)p(\boldsymbol{\Sigma}|\boldsymbol{u}_\Sigma)p(\boldsymbol{u}_\Sigma)d\{\boldsymbol{\Sigma}, \boldsymbol{u}_f, \boldsymbol{u}_\Sigma\}, \tag{2.81}$$

where Σ is a Wishart process. Specifically, its prior is

$$\boldsymbol{\Sigma} \sim \mathcal{WP}_D(\boldsymbol{L}\boldsymbol{L}^\top, \nu, k_f). \tag{2.82}$$

Thus, any marginal $\boldsymbol{\Sigma}(\boldsymbol{x}_t) = \boldsymbol{L}\boldsymbol{J}\boldsymbol{J}^\top\boldsymbol{L}^\top$ is Wishart, and we let $\boldsymbol{J}$ be the $D \times \nu$-matrix with all independent entries $j_{d,v}(\boldsymbol{x}_t)$ drawn from GPs that share the same prior $j_{d,v}(\cdot) \sim \mathcal{GP}(0, k_f(\cdot,\cdot))$. To approximate the posterior of the Wishart process we choose the variational distribution

$$q(\boldsymbol{J}, \boldsymbol{u}_\Sigma) = q(\boldsymbol{J}|\boldsymbol{u}_\Sigma)q(\boldsymbol{u}_\Sigma) := p(\boldsymbol{J}|\boldsymbol{u}_\Sigma)q(\boldsymbol{u}_\Sigma), \tag{2.83}$$

where $q(\boldsymbol{u}_\Sigma) = \prod_{d=1}^{D}\prod_{v=1}^{\nu}\mathcal{N}(\boldsymbol{m}_{d,v}^\Sigma, \boldsymbol{S}_{d,v}^\Sigma)$. Here, $\boldsymbol{m}_{d,v}^\Sigma$ is $M\times 1$ and $\boldsymbol{S}_{d,v}^\Sigma$ is $M\times M$ for each pair $\{d, v\}$.

The same kernel is used for the Wishart process as is used for the random field f, that is: only one kernel *controls* the vector field f. The posterior of $\boldsymbol{\Sigma}$ is defined through the posterior of $\boldsymbol{J}$. Given our choice of kernel, this approximate posterior is identical to Eqs. (2.79)-(2.80), only changing the variational parameters to $\boldsymbol{m}_\Sigma$ and $\boldsymbol{S}_\Sigma$, and D changes to $D\nu$.

Lastly, we define $p(\boldsymbol{f}|\boldsymbol{\Sigma}, \boldsymbol{u}_f)$. Since $\boldsymbol{\Sigma}(\boldsymbol{x}_t)$ is a $D\times D$-matrix we can write

$$p\big(\boldsymbol{f}|\{\boldsymbol{\Sigma}(\boldsymbol{x}_i)\}_{i=1}^N, \boldsymbol{u}_f\big) = \mathcal{N}\big(\tilde{\mu}(\boldsymbol{X}), \tilde{k}_f^\Sigma(\boldsymbol{X},\boldsymbol{X})\big), \tag{2.84}$$

$$\tilde{\mu}(\boldsymbol{x}_i) = \boldsymbol{\alpha}_f^\top(\boldsymbol{x}_i)\mathrm{vec}(\boldsymbol{u}_f), \tag{2.85}$$

$$\tilde{k}_f^\Sigma(\boldsymbol{x}_i, \boldsymbol{x}_j) = \big(\boldsymbol{\Sigma}(\boldsymbol{x}_i) - \boldsymbol{h}_{ij}\big)\delta_{ij}, \tag{2.86}$$

where $\boldsymbol{h}_{ij} = \boldsymbol{\alpha}_f(\boldsymbol{x}_i)^\top k_f^D(\boldsymbol{Z}_f, \boldsymbol{Z}_f)\boldsymbol{\alpha}_f(\boldsymbol{x}_j)$ and δ_{ij} is Kronecker's delta. Notice this, conditioned on the Wishart process, constitutes a FITC-type model [Snelson and Ghahramani, 2006, Quiñonero Candela and Rasmussen, 2005].

This goes beyond the assumption of independent output dimensions, and instead makes the model capture the inter-dimensional dependence structure through the Wishart process Σ. This structure shall also be optimised in the variational inference setup. The posterior of conditional $\boldsymbol{f}$ is approximated by

$$\begin{aligned} q(\boldsymbol{f}, \boldsymbol{u}_f|\{\boldsymbol{\Sigma}(\boldsymbol{x}_i)\}_{i=1}^N) &= q(\boldsymbol{f}|\{\boldsymbol{\Sigma}(\boldsymbol{x}_i)\}_{i=1}^N, \boldsymbol{u}_f)q(\boldsymbol{u}_f) \\ &= p(\boldsymbol{f}|\{\boldsymbol{\Sigma}(\boldsymbol{x}_i)\}_{i=1}^N, \boldsymbol{u}_f)q(\boldsymbol{u}_f), \end{aligned} \tag{2.87}$$

where $q(\boldsymbol{u}_f) := \mathcal{N}(\boldsymbol{m}_f, k_f^D(\boldsymbol{Z}_f, \boldsymbol{Z}_f))$. At first, this might seem restrictive, but covariance estimation is already in Σ and the variational approximation is the simple expression

$$q(\boldsymbol{f}|\{\boldsymbol{\Sigma}(\boldsymbol{x}_i)\}_{i=1}^N) = \prod_{i=1}^{N}\mathcal{N}\big(\boldsymbol{\alpha}_f^\top(\boldsymbol{x}_i)\boldsymbol{m}_f, \boldsymbol{\Sigma}(\boldsymbol{x}_i)\big). \tag{2.88}$$

The marginalisation can then be computed with Jensen's inequality

$$\begin{aligned}\log p(\boldsymbol{y}) = \log \int p(\boldsymbol{y}, \Theta) d\Theta &\geq \int \log \Big(\frac{p(\boldsymbol{y}, \Theta)}{q(\Theta)} \Big) q(\Theta) d\Theta \\ &= \int \log p(\boldsymbol{y}|\boldsymbol{g}) q\big(\boldsymbol{g}|\Theta \setminus \{\boldsymbol{g}\}\big) d\Theta - \mathrm{KL}\big(q(\boldsymbol{u}_g) \| p(\boldsymbol{u}_g)\big) \\ &\quad - \mathrm{KL}\big(q(\boldsymbol{u}_f) \| p(\boldsymbol{u}_f)\big) - \mathrm{KL}\big(q(\boldsymbol{u}_\Sigma) \| p(\boldsymbol{u}_\Sigma)\big) \\ &= \mathbb{E}_{q(g)}[\log p(\boldsymbol{y}|\boldsymbol{g})] - \mathrm{KL}\big(q(\boldsymbol{u}_g) \| p(\boldsymbol{u}_g)\big) \\ &\quad - \mathrm{KL}\big(q(\boldsymbol{u}_f) \| p(\boldsymbol{u}_f)\big) - \mathrm{KL}\big(q(\boldsymbol{u}_\Sigma) \| p(\boldsymbol{u}_\Sigma)\big).\end{aligned} \tag{2.89}$$

The right-hand side in (2.89) is the evidence lower bound. The first term, the expectation, is analytically intractable, due to $q(g)$ being non-conjugate to the likelihood. Therefore, we approximate it with Monte Carlo methods or Gauss-Hermite quadrature [Hensman et al., 2015]. Using Monte Carlo, a few samples often suffice for reliable inference [Salimans and Knowles, 2013].

The KL-terms in (2.89) can be computed analytically as they all involve multivariate Gaussians. One of them is special — the one regarding $\boldsymbol{u}_f$. Since both $q(\boldsymbol{u}_f)$ and $p(\boldsymbol{u}_f)$ have the same covariance matrix it reduces to

$$\mathrm{KL}\big(q(\boldsymbol{u}_f) \| p(\boldsymbol{u}_f)\big) = \frac{1}{2} \sum\nolimits_{d=1}^{D} \boldsymbol{m}_{f_d}^\top k_f^D(\boldsymbol{Z}_f, \boldsymbol{Z}_f)^{-1} \boldsymbol{m}_{f_d}, \tag{2.90}$$

since the kernel and inducing locations are shared.

Outlining the model, we have initial inputs $\boldsymbol{x}_0 := \boldsymbol{x}$ that are warped through an SDE with drift function μ and diffusion Σ. The value of this SDE, at some given time T, is then used as input to a GP g, i.e. $g(\boldsymbol{x}_T)$ predicts targets $y(\boldsymbol{x})$. We note again that y can be both temporal and non-temporal. The model is inferred by maximising the ELBO (2.89).

Scaling to high-dimensional data. In Jørgensen, Deisenroth, and Salimbeni [2020] a method to overcome the computational burden if D is large is presented: a low-rank approximation on the dimensionality-axis. Recall, if $\boldsymbol{\Sigma}_\rho \sim \mathcal{WP}_\rho(\boldsymbol{I}, \nu, \kappa)$, then $\boldsymbol{\Sigma}_D := \boldsymbol{L}\boldsymbol{\Sigma}_\rho \boldsymbol{L}^\top \sim \mathcal{WP}_D(\boldsymbol{L}\boldsymbol{L}^\top, \nu, \kappa)$. These matrices are of rank $\rho \ll D$. The computational overhead is reduced to $\mathcal{O}(\rho^2 NM^2 + D\rho^2)$ if $\nu = \rho$. This is compared with $\mathcal{O}(D\nu NM^2 + D\nu D)$ without the approximation. This same structure was introduced by Heaukulani and van der Wilk [2019] for time-series modelling; and it reminisces the structure of Semi-parametric Latent Factor Models (SLFM) [Seeger et al., 2005], as also discussed in Section 2.1.2.

In this approximation, we need only to sample $\sqrt{\boldsymbol{\Sigma}_D} = \boldsymbol{L}\boldsymbol{J}$, where $\boldsymbol{J}$ is a $\rho \times \nu$ matrix, with GP values according to the approximate posterior $q(\boldsymbol{J})$, with D replaced by ρ.

2.6 Evaluation

This section evaluates what the Wishart diffusions bring to the table. In this respect, we will revisit the dataset studied in Jørgensen, Deisenroth, and Salimbeni [2020]. The dataset measures Air Quality in Beijing, China [Zhang et al., 2017]. To be precise it measures 10 features: the concentration of PM2.5, PM10, SO2, NO2, CO, O3. Further, it observes the temperature and dew point temperature, air pressure and amount of precipitation. These measurements are made hourly for a period of three years (2014-2016). The measurements were done at three locations around the city of Beijing: Tiantan, Dongsi and Shunyi.

The hypothesis in such a dataset would be that there is (at least) correlations to be found in the measurements across the different locations, as the temperature is with high probability the same at these geographically close locations. To this end, we would speculate the Wishart diffusion might pick up on these correlations.

In Jørgensen, Deisenroth, and Salimbeni [2020] three models were compared, but we shall focus on the two most interesting here. Naturally, the first is the Wishart diffusion SDE presented in the previous sections. For comparison, we compare to an SDE model where the diffusion is *diagonal*, but the drift is parametrized like it is for the model above. Thus, the 'control' model has dynamics

$$\boldsymbol{x}_t = \boldsymbol{x}_s + \mu(\boldsymbol{x}_s)(t-s) + \sqrt{\boldsymbol{\Lambda}(t-s)}\boldsymbol{\epsilon}_t, \quad \boldsymbol{\epsilon} \sim \mathcal{N}(\mathbf{0}, \boldsymbol{I}). \tag{2.91}$$

In the paper it was verified that the Wishart model forecasts significantly better than its diagonal counterpart. Here we will verify what is learned in the dynamical model. To this end, we have visualised the dynamics of some features (CO, NO2 and temperature) in Figure 2.5. The lines here indicate how the forecasting evolves for the measurements at two different locations. At first, we may note how the lines for the Wishart diffusions are aligned along the diagonal indicating that when, say CO, rises in Shunyi it is likely to do as well in Dongsi. This aligns with the initial hypothesis, and the signal is especially clear for the temperature measurements, which likely is less affected by local factors such as traffic. Note that there are no axes in the plots, as all measurements have in standardised — for this evaluation we are interested in the dynamics only.

For experiments on regression tasks we refer to [Jørgensen, Deisenroth, and Salimbeni, 2020], which is also appended (Paper B) this thesis.

Future directions

The contributing method we displayed in this chapter is sample intensive, by which we mean that sampling is the computational bottleneck of the approach. Recently, Wilson

et al. [2020] introduced an approach to which such systems can be sampled in linear time (in T), which is a significant speed-up compared to the many recomputations we did for the code in the paper. This computational speed-up comes with only a negligible approximation error.

More fundamentally, it would be interesting to bridge the gap between neural-ODEs and SDEs, such as the Bayesian one presented here. We hypothesise one direction this can be achieved: replacing the Brownian motion with a *fractional* Brownian motion can generate sample paths that are smoother contrasted to the rugged Brownian motion. On the contrary, they can potentially be even more rugged; which could happen if more regularisation is needed. The 'ruggedness' of the sample paths would be decided by the Hurst parameter [Mandelbrot and Ness, 1968]. We know that the fractional Brownian motion is the Gaussian process with covariance function given by

$$k(t,s) = \frac{1}{2}(|t|^{2H} + |s|^{2H} - |t-s|^{2H}), \tag{2.92}$$

where H is the Hurst parameter. We notice that $H = 1/2$ would yield the Brownian motion as described in Definition 2.6. Further, we note that substituting the Brownian motion with a fractional Brownian motion would not alter with the existence of the Ito integrals.

As the last point, we will mention the influence of T, when the intention is regression or classification. It was pointed out in Hegde et al. [2019] that large T generally yields better results. However, with a proper choice of prior distribution over T, we conjecture it would be possible to treat T is a 'regular' parameter, sooner than a hyperparameter. Postulated examples of such priors could be the exponential or Weibull distributions.

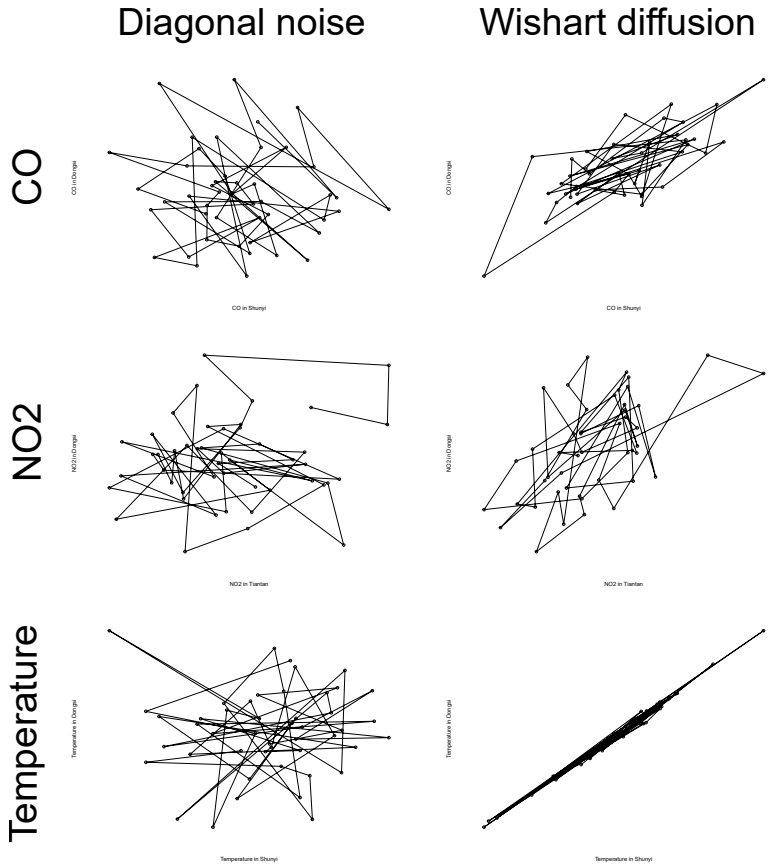


Figure 2.5: Visualisations of the dynamics of both Wishart diffusions and the diagonal setting. We observe Wishart dynamics are more focused around the diagonal, indicating correlated variables.

Chapter 3

Random Manifolds and Latent Variables

At the end of this chapter, I will have presented a method to learn a *random* Riemannian manifold and its associated metric. This will be done by inferring a Gaussian field, that should mimic some metric space we are given. This differs from previous methods by being both generative and based on dissimilarity data, i.e. we do not need observations to lie in a defined coordinate space. The contributions of this chapter is based on the article *Isometric Gaussian Process Latent Variable Model for Dissimilarity Data* [Jørgensen and Hauberg, 2020].

To this end, I will begin by introducing the fundamentals of Riemannian geometry. *Manifolds*, in general, are abstract spaces, and I will go over how the machine learning literature have dealt with learning them. This will provide intuition of how our proposed method is cherry-picking some of the best ideas in this field and combining them in a novel way. We begin by covering standard topology and define exactly the types of manifold we will eventually use. We further present a beginner's guide to persistent homology, which is a data-centric approach to topology.

Then we will argue that uncertainty serves as a 'shadow'-topology, in the sense that it can inform us where to construct our manifolds. This paves the way back into Gaussian processes and the Gaussian process latent variable model (GPLVM) [Lawrence, 2005]. To study geometry on GPs we cover some notions on GP arc lengths. The final contribution includes all these aspects. But first — topology.

3.1 A Primer on Topology and Geometry

In this section, we will cover the mathematical foundation on topology and differential geometry. We begin by defining a topological space, which is the bedrock of all that is to come.

Definition 3.1 A topological space $(\mathcal{X}, \tau)$ is a set $\mathcal{X}$ and τ is a family of subsets satisfying:

(i) $\emptyset \in \tau$,
(ii) $\mathcal{X} \in \tau$,
(iii) For any *finite* collection $U_1, \ldots, U_N$ of sets from τ, then $\bigcap_{i=1}^{N} U_i \in \tau$,
(iv) For any collection U_α, $\alpha \in \mathcal{I}$, then $\bigcup_{\alpha \in \mathcal{I}} U_\alpha \in \tau$.

If τ satisfies (i)-(iv), we call it a *topology*.

Remark. The topology of $\mathbb{R}^D$ is the family of open sets.

Colloquially speaking, a topological space is one that is invariant under simple continuous transformations of the space. This is why topologists are said to be incapable of distinguishing coffee mugs from donuts. These transformations are called *homeomorphisms*.

Definition 3.2 We say that a function $f : \mathcal{X} \to \mathcal{Y}$ between two topological spaces $(\mathcal{X}, \tau_X)$ and $(\mathcal{Y}, \tau_Y)$ is a homeomorphism if it satisfies:

(i) f is a bijection (one-to-one mapping),
(ii) f is continuous,
(iii) f^{-1} is continuous.

If f is a homeomorphism, we say that $\mathcal{X}$ and $\mathcal{Y}$ are *homeomorphic*.

A pathological example of a homeomorphism is given in Figure 3.1, where the sphere can continuously 'morph' into the box, but is not able to split itself into two boxes. Equivalently, the sphere and the box are homeomorphic, but the sphere and two disjoint boxes are not.

With this understanding we can define what a *manifold* is.

Definition 3.3 A (d-)manifold $\mathcal{M}$ is a topological space, such that for any point $\boldsymbol{x} \in \mathcal{M}$, there exist a neighbourhood $U_{\boldsymbol{x}} \in \mathcal{M}$ which is homeomorphic to an open subset of $\mathbb{R}^d$.

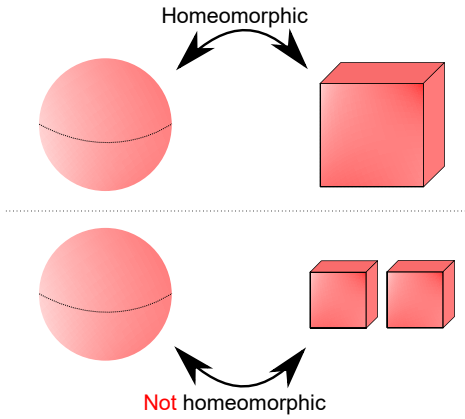


Figure 3.1: A simple example of two sets that are homeomorphic and not homeomorphic. We can continuously warp the sphere into a box and reversely, but we can not warp a sphere into two boxes without tearing.

Remark. A set $U_{\boldsymbol{x}} \subset \mathcal{M}$ is called a neighbourhood of $\boldsymbol{x}$, if there exist $O \in \tau$ such that $\boldsymbol{x} \in O \subset \mathcal{M}$.

So everywhere, very locally, the manifold is topologically equivalent to Euclidean space. Concretezing this, the earth appears, from our viewpoint, flat and Euclidean, but once we zoom out we see the curvature and ultimately spherical manifold structure.

For our purpose, we will also care about metrics on manifolds. In general, when we have a metric space $(\mathcal{X}, d)$, where d denotes the metric, then we have a canonical topology on $\mathcal{X}$.

Example 3.1 *Consider the Euclidean metric $\|\cdot\|$ on $\mathbb{R}^D$. Define the family $\mathcal{A}$ of all open balls*

$$\mathcal{A} := \{B_r(\boldsymbol{x}) \big| \quad \boldsymbol{x} \in \mathbb{R}^D \text{ and } r > 0\} \quad \text{where} \quad B_r(\boldsymbol{x}) = \{\boldsymbol{y} \in \mathbb{R}^D \big| \quad \|\boldsymbol{y} - \boldsymbol{x}\| < r\}, \tag{3.1}$$

then the topology generated with $\mathcal{A}$ as an open basis is the topology on $\mathbb{R}^D$. A family $\mathcal{A}$ is an open basis of a topology τ, if any open set can be written as a union of sets in $\mathcal{A}$. This is exactly the case of $\mathcal{A}$ defined above and the topology τ on $\mathbb{R}^D$. We say that τ was generated by the metric d.

A topological space $\mathcal{X}$ with τ generated from a metric, as above, is *metrizable*. This implies that we can always generate a topology from a metric, but the other way is not ensured. Urysohn's metrization theorem provides sufficient conditions on the topological space to be metrizable.

We avoid this existential question by considering topological spaces generated from metrics. In particular, we consider Riemannian manifolds that have additional constraints on the manifold, but supplies us with a well-defined metric.

Definition 3.4 A Riemannian manifold $\mathcal{M}$ is a smooth q-manifold equipped with an inner product

$$\langle \cdot, \cdot \rangle_{\boldsymbol{x}} : \mathcal{T}_{\boldsymbol{x}}\mathcal{M} \times \mathcal{T}_{\boldsymbol{x}}\mathcal{M} \to \mathbb{R}, \qquad \boldsymbol{x} \in \mathcal{M}, \tag{3.2}$$

that is smooth in $\boldsymbol{x}$. Here $\mathcal{T}_{\boldsymbol{x}}\mathcal{M}$ denotes the tangent space of $\mathcal{M}$ evaluated at $\boldsymbol{x}$.

The length of a curve. The inner product construction of the Riemannian manifold allows us to compute curve-lengths on it. Let $\boldsymbol{c} : [0, 1] \to \mathcal{M}$ be a smooth curve on the manifold. Then the curve-length s is given by

$$s = \int_0^1 \|\dot{\boldsymbol{c}}(t)\|_{\boldsymbol{c}(t)} dt, \quad \text{where } \|\cdot\|_{\boldsymbol{x}} = \sqrt{\langle \cdot, \cdot \rangle_{\boldsymbol{x}}}. \tag{3.3}$$

Having this tool at hand we can define a *metric* on $\mathcal{M}$ by

$$d_{\mathcal{M}}(\boldsymbol{x}_i, \boldsymbol{x}_j) = \inf \left\{ s \middle| \boldsymbol{c} \in C^1([0, 1], \mathcal{M}) \text{ and } \boldsymbol{c}(0) = \boldsymbol{x}_i \text{ and } \boldsymbol{c}(1) = \boldsymbol{x}_j \right\}, \tag{3.4}$$

and we call the curve $\boldsymbol{c}$ a *geodesic* between $\boldsymbol{x}_i$ and $\boldsymbol{x}_j$, if $d_{\mathcal{M}}(\boldsymbol{x}_i, \boldsymbol{x}_j) = \int_0^1 \|\dot{\boldsymbol{c}}(t)\| dt$.

3.2 Manifold and Metric Learning

In this section, we will look at topology, manifolds and metrics from a data-centric viewpoint. The machine learning community has some of its most fundamental ideas — nearest-neighbour search, linkage clustering and dimensionality reduction etc. — rooted in abstract ideas of topology and geometry.

Some of the most popular methods for non-linear dimensionality reduction are deterministic in their nature. This is not an issue if the end goal is visualisation of high-dimensional datasets. However, if the model — or representation — is part of a decision-making system it can be crucial to have calibrated uncertainty quantification. This section also initiates a discussion on the relationship between uncertainty and geometry.

3.2.1 Data manifolds

In machine learning, our information is usually restricted to N observations $\{\boldsymbol{x}_i\}_{i=1}^N$ from an unknown distribution $p(x)$. These observations are residing in some space – usually $\mathbb{R}^D$. The *data manifold* is the restricted 'topological space' where the data lives. This aligns with a common hypothesis in the machine learning community.

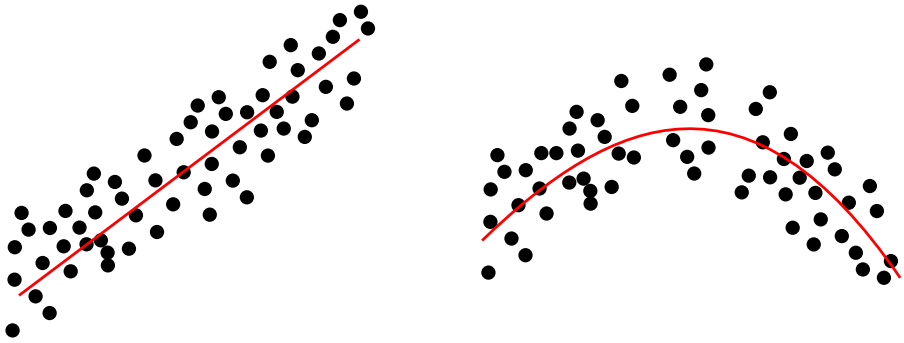


Figure 3.2: The manifold assumption visualised: there exist a manifold (red), that lies close to the datapoints (black). This manifold is of smaller dimension, than the space the data is embedded in. Here the data is in 2 dimensions, while the manifold is 1-dimensional.

The *manifold assumption* hypothesizes that data residing in high-dimensional spaces tend to lie on, or close to, a manifold of lower dimensionality. This implies that the data manifold is locally homeomorphic to a low-dimensional Euclidean space. In Figure 3.2 this phenomenon is visualised with black dots as the observation in $\mathbb{R}^2$. We see, in both cases, that observations lie near a 1-manifold represented by the red line.

If the manifold assumption applies to high dimensions, what is the dimensionality of the data manifold? In general, there is no answer to this question. Johnson-Lindenstrauss' lemma, however, provide some formalism on the manifold assumption [Dasgupta and Gupta, 2003]. They state for some $\epsilon \in (0, 1)$ and N points in $\mathbb{R}^D$, there exists a *linear* mapping $f : \mathbb{R}^D \to \mathbb{R}^q$ such that

$$(1-\epsilon)\|\boldsymbol{x}_i - \boldsymbol{x}_j\|^2 \leq \|f(\boldsymbol{x}_i) - f(\boldsymbol{x}_j)\|^2 \leq (1+\epsilon)\|\boldsymbol{x}_i - \boldsymbol{x}_j\|^2 \qquad \forall i, j \leq N, \tag{3.5}$$

where $q > 8\log(N)/\epsilon^2$. Intuitively, this means that in very high dimensional spaces we can preserve Euclidean distances by randomly projecting to a significantly smaller Euclidean subspace. However, if the mapping is not linear, consider for example the right-hand side of Figure 3.2, things are not as straightforward.

From an alternative viewpoint, when we move around on the data manifold, we will always be *close* to the data. This illustrates, why the red line has to curve on right-hand side of Figure 3.2. If the two endpoints of the red line had been linearly interpolated, then we would have moved through a region *not close* to the data, hence we moved *off* the data manifold.

Data Topology. As a first sanity check, one would be interested in topological invariants of the data manifolds, such as connectedness and holes. This gives some information about where we can not move on the data manifold, as we should not

move between components — in other words, *clusters* — or through holes. We wish to study the homological features of the space $\mathcal{X}$, through samples $\{\boldsymbol{x}_i\}_{i=1}^N$ that are *near* $\mathcal{X}$.

Definition 3.5 (Vietoris-Rips complex) Let $(\mathcal{X}, d)$ be a metric space. $VR(\mathcal{X}, \epsilon)$ is a Vietoris-Rips complex with vertex set $\mathcal{X}$ if any finite set of points $\{x_0, x_1, \ldots, x_k\}$ from the vertex set satisfies

$$\{x_0, x_1, \ldots, x_k\} \text{ is a } k\text{-simplex} \iff d(x_i, x_j) \leq \epsilon \quad \forall i, j \in \{1, \ldots, k\}. \tag{3.6}$$

A *k-simplex* can be thought of as a higher-dimensional graph, i.e. a 1-simplex is graph. In general, a k-simplex is the convex hull of $k+1$ vertices. An *abstract simplicial complex* $\mathcal{V}$ is a finite set of simplices, that is closed under taking subsets. That is, for any $V \subset \mathcal{V}$ and any $W \subset V$, then $W \subset \mathcal{V}$.

The reason for introducing simplicial complexes is because their *homology* is easily computed. For this thesis we will not show how this is done, but focus on the *why*. For more on the exact computation we refer to Carlsson [2009].

3.2.2 Persistent Homology

Homology is an algebraic topology concept describing topological invariants. In this sense, it is an algebraic treatment of whether two topological space are homeomorphic or not. The key idea is that homology studies 'dimensional' holes of the topological spaces. A 0-dimensional hole is a 'gap' between two vertices, thus if there are no 0-dimensional holes the vertex is connected. A 1-dimensional hole, or a circular hole, can be thought of a loop on the simplex; while 2-dimensional holes are 'void'-like — think the inside of a sphere. This generalizes to any dimensions, but are quite abstract quantities. For a complicated $\mathcal{V}$ the numbers $\beta_0(\mathcal{V}), \beta_1(\mathcal{V}), \ldots$, known as the Betti numbers, count the number of holes in each dimension. This sequence of numbers describe topological invariants.

Figure 3.3 shows how the Vietoris-Rips complex is build from the closed balls with radii ϵ. From the point set on the left, we cover it with closed balls of some radii. If two points lie within each other's coverings, we connect them to a 1-simplex. Had three points shared coverings in a similar way, they would have connected to a 2-simplex, and so forth. On the right in Figure 3.3 we see the resulting complex, which has a different topology than 'trivial' topology of the point cloud on the left. Naturally, this complex is sensitive to the threshold parameter ϵ.

The most reliable homological information is obtained by not analysing a fixed value of ϵ. Instead, we analyse a range of different complexes to review the homological information for multiple values of ϵ. It is from this idea, that we talk about *persistent* homology — we wish to obtain a *signal* in the topology and leave out the noise, that stems from infrequent or noisy sampling of the point set.

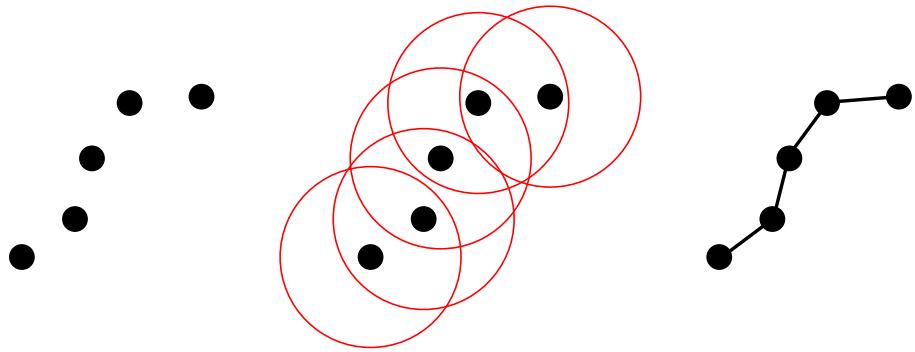


Figure 3.3: Trivial example of a Vietoris-Rips construction. On left we see data, we cast a sphere of some radius ϵ over each observation and make simplices of everything that is contained within the same sphere. The final complex is visulalised on the right; here, the complex is solely made of 1-simplices.

Example 3.2 *Throughout this chapter we will focus on an easily understood 3-dimensional dataset known as the* Open Box. *The data is shown in Figure 3.4 and is best thought of as a cardboard box with the 'lid' open, meaning the lid is only connected to one of the 'faces' of the open box. It consists of 3000 data points and each face is a square with side lengths* 1. *It should be clear that it is homeomorphic to a* 2*-manifold and the ground truth manifold (as indicated by the black lines in Figure 3.4) consists of one connected component and no holes of any dimension.*

Initially, we could try out $\epsilon \approx 0$ *for our Rips-complex. This yields the trivial topology of every point being their own connected component — or cluster. Sequentially, we can increase* ϵ *and see how quick some components merge into one; we say one component is 'dying'. For simplicial complex this increase in* ϵ *generates a filtration that is visualised in Figure 3.5, and this is called a Rips diagram. This plot was generated using the Open Box data from Figure 3.4.*

Interpreting the output, we find that long lifetimes indicate a 'persistence' for a wide range of ϵ *values. Short lifespans indicate holes that only appear because of the noisy nature of a finite sample size. In Figure 3.5 we illustrated (shaded area) that choosing* $\epsilon = 0.15$ *yields one connected component and two* 1*-dimensional holes, both of which appear to be quite persistent. They are persistent because of their significantly longer lifetime than the majority of the holes generated in this filtration.*

This example illustrates the key idea of persistent homology: considering a *filtration* of ϵ-values, can obtain good indications of the homology information, and ultimately build our models on the most accurate topological space. Although the example indicated two holes in the manifold, which differs from the ground truth, there are results that

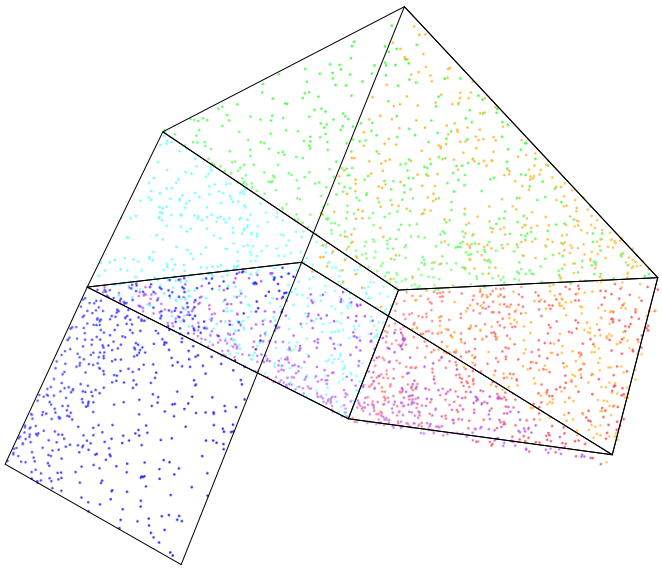


Figure 3.4: The 3-dimensional 'Open Box' dataset. Although, the perception cheats, all faces are squares of area 1.

indicate persistent homology is robust, when samples are dense on the manifold [Chazal and Michel, 2017]. Here we just outlined the ideas of persistent homology, and showed how to 'eyeball' a robust value of ϵ.

3.2.3 Data Geometry

Geometry, in general terms, is a toolbox for dealing with lengths, angles and volumes. The word originates from ancient Greek and translates to *earth measurement*. Often, we find people talking about some space having *a* geometry. In this terminology, they often mean some kind of structure, allowing them to consider lengths, angles or volumes, built on top of some topological space. The simplest is the Euclidean geometry, where every length between two points is the length of the linear interpolation between them. If we allow the line to curve, or the surfaces to have curvature we can describe this structure using Riemannian geometry. Here curves are the generalization of lines, and manifolds the generalization of surfaces.

In non-linear dimensionality reduction [Lee and Verleysen, 2007] the goal is often to capture this geometric structure induced by a dataset in $\mathbb{R}^D$ and represent the dataset — with its geometric structure — in $\mathbb{R}^q$, where $q << D$. To make algorithms from

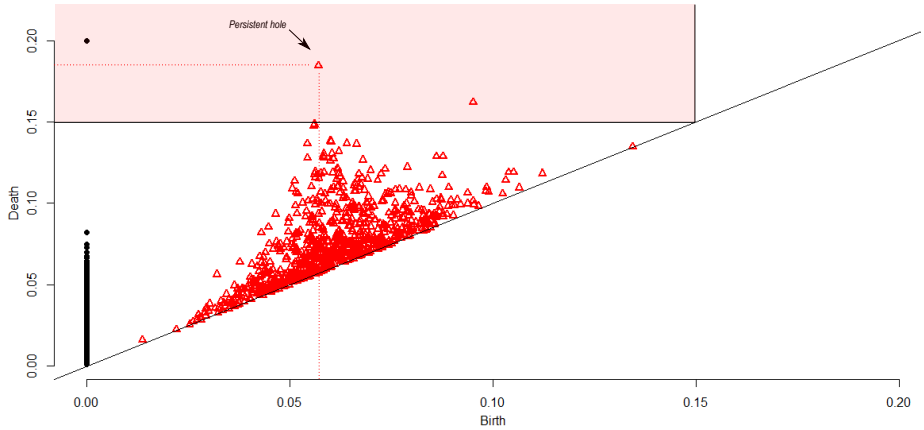


Figure 3.5: Rips Persistence Diagram. Black dots indicate connected components that are all born at $\epsilon = 0$, but die at later times. We see that there is one *persistent* component. The red triangles are the 1-dimensional holes. They are born at different time-points, but most of them die shortly after; this indicates the hole was caused by noise or infrequent sampling.

this, one common criterion is to preserve the distances of observations in $\mathbb{R}^D$ in the latent representation $\mathbb{R}^q$.

Distance Preservation. The most fundamental algorithm that builds its foundation on preservation of pairwise distances is *Multi-Dimensional Scaling* (MDS). In fact, this algorithm is an umbrella term for a range of methods [Mead, 1992]. Classical metric MDS considers the case where distances between points in $\mathbb{R}^D$ are all the Euclidean distance

$$d(\boldsymbol{x}_i, \boldsymbol{x}_j) = \sqrt{\sum_{k=1}^{D} (x_{ik} - x_{jk})^2}, \tag{3.7}$$

and it then tries to find a latent presentation $\mathcal{Z}$ with the Euclidean metric such that it minimises pairwise distances with the *stress* defined as

$$\text{stress} := \sum_{i,j=1}^{N} (b_{i,j} - \langle \boldsymbol{z}_i, \boldsymbol{z}_j \rangle)^2, \quad \text{where} \quad b_{i,j} = \frac{\langle \boldsymbol{x}_i, \boldsymbol{x}_i \rangle + \langle \boldsymbol{x}_j, \boldsymbol{x}_j \rangle - d(\boldsymbol{x}_i, \boldsymbol{x}_j)^2}{2}, \tag{3.8}$$

which one can note does not directly minimise pairwise distances, but rather inner products (which, of course, is closely related). The advantage of this formulation is that there exists a closed form algebraic solution, and in the Euclidean case this is the PCA solution.

A generalisation would be to consider the more straightforward definition of stress.

That is, directly comparing pairwise distances

$$\text{stress} := \sum_{i,j=1}^{N} \left(d(\boldsymbol{x}_i, \boldsymbol{x}_j) - d(\boldsymbol{z}_i, \boldsymbol{z}_j)\right)^2, \tag{3.9}$$

which is often optimised with gradient descent or stress majorisation. In this formalisation, we do not require that the metric on the data space can be written as some inner product. In fact, it is not a requirement that in fact is a metric, but we can fill in any positive number to substitute $d(\boldsymbol{x}_i, \boldsymbol{x}_j)$. There exists further generalisations, such as *non-metric* MDS, but we will not expand on those here.

3.2.4 IsoMap

This section is dedicated to in depth describing the method introduced by Tenenbaum et al. [2000]. The method is called *IsoMap*, and the aim is to consider the distances between points induced by geodesics on some manifold instead of Euclidean distances. We present the pseuso-code here, for easy reference, and expand on each step after.

Initialise Determine initial values $\mathcal{Z} = (\boldsymbol{z}_1, \boldsymbol{z}_2, \ldots, \boldsymbol{z}_N)$ in $\mathbb{R}^q$. Go to **NN-Graph.**

NN-Graph Using either k-nearest neighbours or every neighbour within distance ϵ, compute a neighbourhood graph for data-points $\mathcal{X} = (\boldsymbol{x}_1, \ldots, \boldsymbol{x}_N)$. Go to **Dijkstra**.

Dijkstra For any two pairs, $\boldsymbol{x}_i$ and $\boldsymbol{x}_j$, compute the shortest path length connecting them. Set $d(\boldsymbol{x}_i, \boldsymbol{x}_j)$ to this value. Go to **metric-MDS**.

metric-MDS Optimise the stress (3.9), with respect to latent points $\mathcal{Z}$, until convergence.

The **NN-Graph** and the **Dijkstra** step is what really define the IsoMap algorithm. The graph embedding is kind of a discrete manifold — the linear distance is a good approximator, as long as the distance is small. It is also this step that introduces the only hyperparameter of the algorithm, either k for a k-NN embedded graph, or ϵ for a graph similar to the Vietoris-Rips complex (Definition 3.5) restricted to at most contain 1-simiplices.

Dijkstra's algorithm [Dijkstra et al., 1959] is an efficient computation of the shortest path length between two nodes on a graph. The geometric interpretation here is that if the neighbourhood graph is a good approximation of the manifold, then the length between two nodes computed via Dijkstra is a representation of a geodesic on the manifold. For dense matrices, it can be more efficient to use the Floyd-Warshall algorithm [Floyd, 1962].

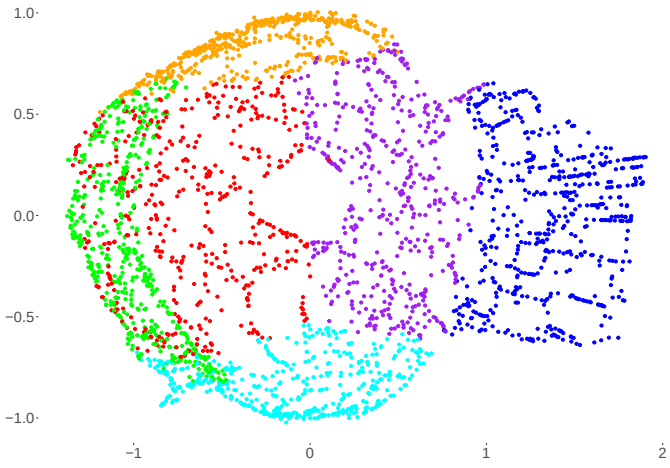


Figure 3.6: IsoMap representation of the Open Box Dataset. Each colour represent a *face* of the box, and can be compared with the colours of Figure 3.4.

The final step in the IsoMap algorithm is to perform **metric-MDS**, in order to preserve these 'geodesic' distances in our latent representation. That is, we optimise

$$\text{stress} := \sum_{i,j=1}^{N} \left(d_{\text{Dijkstra}}(\boldsymbol{x}_i, \boldsymbol{x}_j) - d(\boldsymbol{z}_i, \boldsymbol{z}_j)\right)^2, \tag{3.10}$$

where d_{Dijkstra} is the shortest path on the neighbourhood graph and d is the Euclidean distance.

Example 3.3 *We revisit the Open Box dataset from Example 3.2. Our aim is to represent the dataset in two dimensions — clearly, this is possible as there exists a 2-manifold. The dataset contains 3000 observations, so we choose to build a k-NN graph using* $k = 5$.

Figure 3.6 shows the latent representations. In this visualisation the Euclidean distance should represent geodesics between points. We can observe that the faces *of the boxes seem to be well separated, except from the green and red in particular. Beyond this, we see that the green, orange and cyan faces have a 'rounding' effect, and as such it is not clear that they represent a square with side lengths 1.*

Figure 3.6 further illustrates another common issue with IsoMap — it has tendencies to create 'holes' that are not represented in the data. In other words, *non-persistent* holes appear in the latent representation. This is likely explained with only considering 1-simplices naturally generates holes in the sense of Section 3.2.2. This is not the

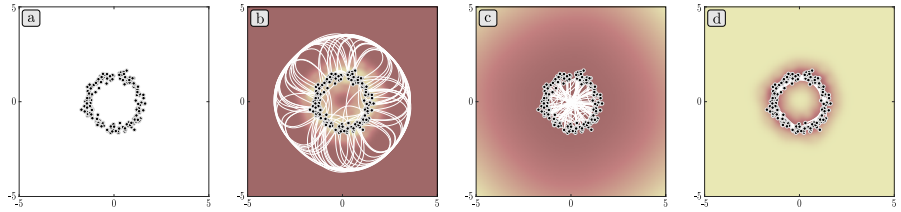
only topological concern with IsoMap; we are necessarily required to assume that the manifold is path-connected, else we can not compute the distances between all points. This issue can be averted by either removing outliers or considering two (or more) clusters separately. The latter suffers from having no *push* effect between clusters, unless an artificial one is created — i.e. separate clusters can theoretically intersect in the latent space. The next section considers how we can keep the 'path-connected' assumption, but represent holes (also 0-dimensional holes) with uncertainty.

3.2.5 What is the role of Uncertainty in Geometry?

This section opens a discussion of the role of uncertainty within geometry. As discussed already, geometry deals with computation of lengths, volumes etc. Revisiting the manifold assumption, we assume that we can move along some manifold of dimension q and always remain *close* to the data points embedded in $\mathbb{R}^D$. Mirroring this logic: when we move away from the data points, we move *off* the low-dimensional manifold — we may even say that we move into a hole. Recall here that a 0-dimensional hole is simply the 'gap' between non-connected manifolds. For now, we will interpret a hole as space a curve can not move through — this is coherent with the homotopy group of the underlying topological space, but we stick with the interpretation for now. This implies that all curves stay on the manifold (thus near data) and of course this is also true for the geodesic. In this interpretation, holes are regions of the space with no (or little) data support.

Hauberg [2018] studied these geodesics under different Riemannian manifolds. Some of the experiments are shown in Figure 3.7. In (a), it shows the latent embedding of a dataset which resides in $\mathbb{R}^{1000}$, and for (b)-(d) it shows the geodesics from three different manifolds. We will go into depth on how the geodesics are computed in the next section, when we discuss *pull-back metrics*. For now, we will consider the background colour of (b)-(d) which indicates the determinant of the *expected metric tensor*. In our setting the metric tensor is a 2×2 matrix (in general, $q \times q$). Light yellow indicates high values, and dark red lower values. We see that geodesics tend to avert areas where this measure is high. The thing to notice here is that GPLVM (also, yet to be introduced) is the only representation to obtain the geometry that the manifold assumptions hypothesise. The GP uncertainty (with proper choice of kernel) is high away from data, and we can see that the determinant of the metric tensor acts similar — i.e. it is high away from data. Thus we can hypothesise that uncertainty is correlated with the metric tensor.

This quick example brings us back to Chapter 1 and the idea that it is possible to *extrapolate* uncertainty estimates; and we can as such cover holes with high uncertainty, forcing geodesics to stay on the manifold. It also appears to be a good reason to explore probabilistic (read Bayesian) generative models within manifold learning. We will cover many of the concepts introduced here in the next section.



(a) The embedding of data in $\mathbb{R}^2$. (b) RBF kernel. (c) RBF + linear kernel. (d) GPLVM.

Figure 3.7: Geodesics (white curves) computed from different manifolds in the latent space visualised. Only the 'uncertain' GPLVM recover the geometry that is coherent with the manifold assumption. This figure originally appears in Hauberg [2018].

3.3 Generative Models

In statistical modelling, we can often separate *generative* and *discriminative* modelling; given a dataset of explanatory variables $\boldsymbol{X}$ and target variable $\boldsymbol{y}$ we separate between modelling

$$p(\boldsymbol{X}, \boldsymbol{y}) \qquad \text{and} \qquad p(\boldsymbol{y}|\boldsymbol{X}). \tag{3.11}$$

In the first setup, the generative one, if our statistical model is successful, we can generate new samples $\{X^*, y^*\}$. In the discriminative, we would have to know $X^* = x^*$ before generating a (perhaps) suitable y^*.

In this view, all generative modelling is density estimation. In this chapter, we are interested in unsupervised learning, so we will be modelling datasets $\mathcal{D} = \{\boldsymbol{x}_i\}_{i=1}^N$ with no target variable. In machine learning, we often introduce *latent variables* $\boldsymbol{z}_i$ to assist us in modelling. In the previous sections we already encountered these as latent representations of the data. For full disclosure, neither MDS or IsoMap are dealing with *statistical* modelling, as we at no point treat either $\boldsymbol{X}$ or $\boldsymbol{Z}$ as *stochastic* variables. This naturally also limit their uses to mostly visualisation purposes.

In the following, we will consider models which generating process reads

$$\boldsymbol{x}_i = f(\boldsymbol{z}_i) + \boldsymbol{\epsilon}_i, \tag{3.12}$$

where $\boldsymbol{z}_i$ is an unobserved (or latent) variable and $\boldsymbol{\epsilon}$ follows some noise distribution. In other words, we can optimise some likelihood defined $p(\boldsymbol{x}|f(\boldsymbol{z}))$ with respect to f and $\boldsymbol{z}$. If $\boldsymbol{z}$ is deterministic, this would be a discriminative model; if we assume a distribution over $\boldsymbol{z}$, we could marginalise it and optimise the marginal likelihood $p(\boldsymbol{x})$, and we would characterise it as a generative model. Models like (3.12), introducing latent variables $\boldsymbol{z}$ are usually called *latent variable models*.

If $\boldsymbol{z}$ follows some distribution q, it is trivial to generate new samples from $p(\boldsymbol{x})$, by

first sampling $\boldsymbol{z}^* \sim q(\boldsymbol{z})$, and then $\boldsymbol{x}^* = f(\boldsymbol{z}^*) + \boldsymbol{\epsilon}$ is a sample from $p(\boldsymbol{x})$.

Pull-back metric. Latent variable models of the type (3.12) provide a framework for studying the geometry of the latent space using Riemannian manifolds. Assume we have observed points $\boldsymbol{X} = (\boldsymbol{x}_1, \boldsymbol{x}_2, \ldots, \boldsymbol{x}_N)^\top \subset \mathcal{M}_X$ and latent points $\boldsymbol{Z} = (\boldsymbol{z}_1, \boldsymbol{z}_2, \ldots, \boldsymbol{z}_N)^\top \subset \mathcal{M}_Z$, where $\mathcal{M}_X$ and $\mathcal{M}_Z$ are Riemannian manifolds. Let $f : \mathcal{M}_Z \to \mathcal{M}_X$ be a differentiable function.

Let $\boldsymbol{c}$ be a curve on $\mathcal{M}_Z$, and let us consider the length of the curve $f(\boldsymbol{c})$ on $\mathcal{M}_X$. To ease the analysis, we will assume $\mathcal{M}_X$ is the Euclidean space $\mathbb{R}^D$ with the usual inner product. Following (3.3) the length of $f(\boldsymbol{c})$ can be computed

$$\int_0^1 \left\| \frac{\partial f(\boldsymbol{c}(t))}{\partial t} \right\| dt = \int_0^1 \sqrt{\langle \dot{f}(\boldsymbol{c}(t))\dot{\boldsymbol{c}}(t), \dot{f}(\boldsymbol{c}(t))\dot{\boldsymbol{c}}(t)\rangle} dt \tag{3.13}$$

$$= \int_0^1 \sqrt{\langle \boldsymbol{J}_{\boldsymbol{c}(t)}\dot{\boldsymbol{c}}(t), \boldsymbol{J}_{\boldsymbol{c}(t)}\dot{\boldsymbol{c}}(t)\rangle} dt \tag{3.14}$$

$$= \int_0^1 \sqrt{\langle \boldsymbol{J}_{\boldsymbol{c}(t)}\dot{\boldsymbol{c}}(t), \boldsymbol{J}_{\boldsymbol{c}(t)}\dot{\boldsymbol{c}}(t)\rangle} dt \tag{3.15}$$

$$= \int_0^1 \sqrt{\langle \dot{\boldsymbol{c}}(t), \dot{\boldsymbol{c}}(t)\rangle_{\mathcal{M}_Z^f}} dt, \tag{3.16}$$

where we defined $\langle \dot{\boldsymbol{c}}(t), \dot{\boldsymbol{c}}(t)\rangle_{\mathcal{M}_Z^f} = \langle \dot{\boldsymbol{c}}(t), \boldsymbol{J}_{\boldsymbol{c}(t)}^\top \boldsymbol{J}_{\boldsymbol{c}(t)}\dot{\boldsymbol{c}}(t)\rangle$.

Thus, $\mathcal{M}_Z^f$ is a Riemannian manifold with smooth varying inner product and the associated metric tensor $\boldsymbol{M}_Z(\boldsymbol{z}) = \boldsymbol{J}_{\boldsymbol{z}}^\top \boldsymbol{J}_{\boldsymbol{z}}$ for $\boldsymbol{z} \in \mathcal{M}_Z^f$. We call this the *pull-back metric* (tensor), since we have pulled it from the Euclidean metric in $\mathbb{R}^D$ through f to our manifold of interest.

This is convenient for learning manifolds and metrics through observations in some space, which usually for machine learning tasks is Euclidean, in the lack of better. Now the function f allows us to study geometric properties of the data in latent spaces. We remark that the pull-back metric exists for general metrics, i.e. we can pull any Riemannian metric down, not just the Euclidean.

Example 3.4 *We revisit Figure 3.7, and discuss why the geodesics have unwanted behaviour. We recall the observed data resides in* $\mathbb{R}^{1000}$ *and that subfigure (a) is the* true *latent representation*[1]*. Subfigure (b)+(c) use* kernel ridge regression *[Nadaraya, 1965] for the map* $f : \mathbb{R}^2 \to \mathbb{R}^{1000}$*, i.e. a deterministic function. That is,*

$$f(\boldsymbol{z}^*) = k(\boldsymbol{z}^*, \boldsymbol{Z})\big(k(\boldsymbol{Z}, \boldsymbol{Z}) + \sigma^2 \boldsymbol{I}_N\big)^{-1} \boldsymbol{X}. \tag{3.17}$$

In (b), the kernel used is the RBF kernel, so why are the geodesics moving away from the representation? By inspecting the metric tensor in regions away from data we observe that the Jacobian $\boldsymbol{J}$ *becomes* $\mathbf{0}$*, when we move away from* $\boldsymbol{Z}$*, because* f *becomes constant when we consider the RBF kernel. Thus, in the pull-back metric the*

[1] Data was non-linearly embedded in $\mathbb{R}^{1000}$ from this representation.

curves can move in these regions at no extra cost, hence geodesics would move to these areas.

For (c) the kernel is a RBF kernel added to a linear kernel. Again, the RBF kernel would tend to move geodesics away from data, while the linear components will emphasize geodesics as linear interpolations.

The background color is what Bishop et al. [1997] call the Magnification factor*. It is defined as* $\sqrt{det(\boldsymbol{J}^\top \boldsymbol{J})}$*. Dark red colors indicate small values of it, which again correlates with* $\boldsymbol{J} \approx \boldsymbol{0}$*. In the next section, we will investigate what happens if* f *is a GP.*

3.3.1 Gaussian Process Latent Variable Model

The Gaussian Process Latent Variable Model (GPLVM) [Lawrence, 2005, Titsias and Lawrence, 2010] is a latent variable model where the generating process is a GP. The model reads

$$\boldsymbol{x}_i = f(\boldsymbol{z}_i) + \boldsymbol{\epsilon}_i, \qquad i = 1, \dots, N, \tag{3.18}$$

where $\boldsymbol{\epsilon} \sim \mathcal{N}(\boldsymbol{0}, \sigma^2 \boldsymbol{I}_D)$ and $f : \mathbb{R}^q \to \mathbb{R}^D$ is a GP. Here, we assume the measurement noise (i.e. the likelihood) is Gaussian, but it could be any distribution.

There exists formulations of the GPLVM, where the latents are treated as parameters (or point estimates) to be optimised [Lawrence, 2005], and where they are marginalised [Titsias and Lawrence, 2010]. The latter uses variational inference in GPs (Section 2.1.3) and maximise the ELBO given as

$$\mathcal{L} = \mathbb{E}_{q(f,\boldsymbol{z})}[\log p(\boldsymbol{x}|f(\boldsymbol{z})] - \mathrm{KL}(q(\boldsymbol{u})\|p(\boldsymbol{u})) - \mathrm{KL}(q(\boldsymbol{z})\|p(\boldsymbol{z})), \tag{3.19}$$

where $\boldsymbol{u}$ denotes the inducing points.

Derivative of a Gaussian Process. GPs are differentiable when their covariance functions are differentiable [Rasmussen and Williams, 2006]. The linear nature of the differential operation implies that the derivative of a GP is a GP itself. Let $\boldsymbol{J}$ denote the Jacobian of a GP $f : \mathbb{R}^q \to \mathbb{R}^D$, with covariance function k for each output dimension. Thus, $\boldsymbol{J}$ is a $D \times q$-matrix. We can then write the joint Gaussian of $\boldsymbol{f} = f(\boldsymbol{X})$ and the derivative $\boldsymbol{J}$ at some locations $\boldsymbol{x}^* \in \mathbb{R}^q$ as

$$\begin{pmatrix} \boldsymbol{f} \\ \boldsymbol{J}^\top \end{pmatrix} \sim \mathcal{N}\left(\begin{pmatrix} \boldsymbol{0} \\ \boldsymbol{0} \end{pmatrix}, \begin{pmatrix} k(\boldsymbol{X}, \boldsymbol{X}) & \partial k(\boldsymbol{X}, \boldsymbol{x}^*) \\ \partial k(\boldsymbol{X}, \boldsymbol{x}^*)^\top & \partial^2 k(\boldsymbol{x}^*, \boldsymbol{x}^*) \end{pmatrix} \right), \tag{3.20}$$

where

$$\partial k(\boldsymbol{X}, \boldsymbol{x}^*)_{i,d} = \frac{\partial k(\boldsymbol{x}_i, \boldsymbol{x}^*)}{\partial x^*_{(d)}}, \quad \text{and} \quad \partial^2 k(\boldsymbol{x}^*, \boldsymbol{x}^*)_{d,d'} = \frac{\partial^2 k(\boldsymbol{x}^*, \boldsymbol{x}^*)}{\partial x^*_{(d)} \partial x^*_{(d')}}, \tag{3.21}$$

where $i = 1, \ldots, N$ and $d, d' = 1, \ldots, q$. Here $x_{(d)}$ denotes the d-th entry in $\boldsymbol{x} \in \mathbb{R}^q$. Notice the above is a multi-output GP, and we have for ease assumed a constant mean prior, but it could be generalized.

We see that $\boldsymbol{J}^\top \boldsymbol{J}$ is a Wishart process (see Definition 2.5). This is also the pull-back metric tensor of the GPLVM.

Riemannian metric from the gplvm. Tosi et al. [2014] were the first to study the Riemannian structure induced by the GPLVM. They make the observation that the $\boldsymbol{J}$ is a GP and $\boldsymbol{J}^\top \boldsymbol{J}$ is (non-central) Wishart distributed. The stochasticity of the metric tensor — and thus randomness of the Riemannian manifold itself — prohibit the use of the usual toolbox from differential geometry to compute geodesics. For a *deterministic* manifold geodesics can be computed by solving the second order ODE

$$\boldsymbol{c}'' = -\frac{1}{2}\boldsymbol{M}^{-1}\left(2(\boldsymbol{I}_d \otimes \boldsymbol{c}'^\top)\frac{\partial \mathrm{vec}(\boldsymbol{M})}{\partial \boldsymbol{c}}\boldsymbol{c}' - \left(\frac{\partial \mathrm{vec}(\boldsymbol{M})}{\partial \boldsymbol{c}}\right)^\top (\boldsymbol{c}' \otimes \boldsymbol{c}')\right), \tag{3.22}$$

where $\boldsymbol{M} = \boldsymbol{J}^\top \boldsymbol{J}$. There exists other methods to compute geodesics, but this seems to be the approach scaling most graciously. In fact, the problem with geodesics on *random manifolds* is more fundamental than computations — there exists no established definition or notion of what it means to be a geodesic on a random manifold.

Tosi et al. [2014] avert this issue by considering the *mean* metric tensor. This is given by

$$\mathbb{E}[\boldsymbol{M}] = \mathbb{E}[\boldsymbol{J}^\top \boldsymbol{J}] = \mathbb{E}[\boldsymbol{J}^\top]\mathbb{E}[\boldsymbol{J}] + D \cdot \mathrm{Cov}(\boldsymbol{J}^\top). \tag{3.23}$$

This way they can apply the rules from deterministic Riemannian geometry to analyse our manifold. By inspecting the mean metric tensor, we observe that the uncertainty in the GP is represented through $\mathrm{Cov}(\boldsymbol{J}^\top)$. Here, it is crucial to realise that the mean manifold is *not* the same as the manifold 'pulled-back' from the mean of f.

We can now revisit Figure 3.7 for the last time. The background colour in (d) is the magnification factor from the mean metric tensor, i.e.

$$\sqrt{\det(\mathbb{E}[\boldsymbol{M}])}, \tag{3.24}$$

and it is clear that the covariance term is dominant, when D gets large. For the RBF kernel (as used in the figure, [Hauberg, 2018]) the extrapolation of variance for f follows through to the Jacobian (up to a scaling). This explains why we see the magnification factor is large away from data; and as a consequence geodesics always lie close to the data region. The manifold pulled from the mean of f, would look like Figure 3.7(b). Thus, the *uncertainty* is crucial for meaningful geodesics.

Eklund and Hauberg [2019a] study how the length of the curve on the mean manifold approximates the mean length of the curve on the random manifold. They show that as D gets large, the approximation is tight.

The method we introduce in the next section is not the first that wish to encode geometry and topology into the GPLVM. Other constraints have been tried out by Urtasun et al. [2008] and Lawrence and Quiñonero Candela [2006].

3.4 Isometric GPLVM

In this section, we will cover the approach presented in Jørgensen and Hauberg [2020]. At the fundamental level, the model presented there base its ideas on the Bayesian GPLVM and IsoMap. The main difference from the GPLVM is that its input is not given as tabular data, but rather as proximity data — also known as dissimilarity data. Hence, the model is coordinate-free, but only considers pairwise-distances. The similarity with IsoMap is then, that we will try to maintain the geodesics along some neighbourhood graph. This is based on the manifold assertion, i.e. we can anywhere locally consider the space to be Euclidean. The consequence of this is that all short pairwise distances have geodesics that are well-approximated by linear interpolation. This is one of the central ideas of IsoMap as well.

We overcome some of IsoMap's caveats, i.e. not being forced to have a fully connected manifold, thus we are able to model datasets which contain persistent clusters. On top of this, we inherent good uncertainty estimation in the estimated manifold. This stems from the GPLVM part of the model, and we can assess the uncertainty at any point in the continuum of the latent space. Unlike IsoMap the metric in the latent space is not Euclidean; we may think of it as being an IsoMap with curvature.

Since we do not consider coordinates, we can not consider the GPLVM in the usual sense. Instead, we model the Jacobian $\boldsymbol{J}$ directly to keep the implicit mapping f isometric, i.e. distances in data space are equal to those in latent space. For trustworthy geodesics, we introduce the concept of censoring — a tool known mainly from survival analysis to handle 'informed' missing data. In the presented model, we use it to keep large distances in data space large in latent space. Unlike most models that deal with proximity data, we specify a full statistical model, in which we can marginalise the underlying mapping and the latent representation too. A key step to this is motivating a likelihood function based on Gaussian process theory. We will here present the pseudo-code for the approach, to provide an overview, before detailing each step.

Initialise Determine initial values $\mathcal{Z} = (\boldsymbol{z}_1, \boldsymbol{z}_2, \ldots, \boldsymbol{z}_N)$ in $\mathbb{R}^q$. Go to **Connectivity.**

Connectivity Choose $\epsilon > 0$. Construct the Vietoris-Rips complex $VR(\mathcal{X}, \epsilon)$ for data-points $\mathcal{X} = (\boldsymbol{x}_1, \ldots, \boldsymbol{x}_N)$. Go to **Censoring**.

Censoring Right-censor all pairs that are not connected in the complex with the value ϵ. Go to **Optimise**.

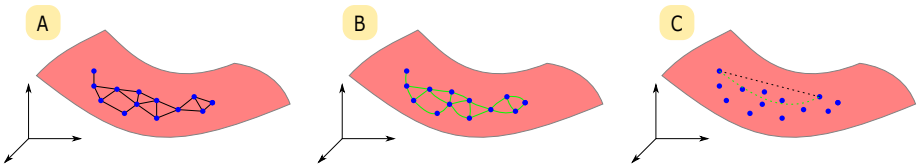


Figure 3.8: A small dataset embedded in $\mathbb{R}^3$ and a 2-dimensional manifold (red). Black lines are linear interpolations in $\mathbb{R}^2$, green lines are curves on the manifold. Solid lines are *short*, and the manifold distances should have approximately same lengths. Dashed lines are not short, and manifold distances should not either.

Optimise Optimise the Nakagami-based ELBO (3.43), with respect to latent points $\mathcal{Z}$ and $\boldsymbol{J}$, until convergence.

We will begin the detailing of the algorithm from back to start. Thus, first we will introduce the *Nakagami* motivated likelihood function. Remember, that our data consists of $N(N-1)/2$ observations of pairwise distances, we will denote them $\{e_{ij}\}_{i<j\leq N}$. We are interested in finding a manifold that matches the shortest distances. To do this we are considering the manifold distances

$$s_{ij} = \int_0^1 \|\boldsymbol{J}(\boldsymbol{c}(t))\dot{\boldsymbol{c}}(t)\|dt, \tag{3.25}$$

where $\boldsymbol{c}(0) = \boldsymbol{z}_i$ and $\boldsymbol{c}(1) = \boldsymbol{z}_j$. The curves we will consider lie on an implicit q-dimensional manifold, and should approximate linear interpolations in $\mathbb{R}^D$, only when curves are short. This motivation is visualised in Figure 3.8 where the manifold is visualised by the red surface. The black lines are the linear interpolations and the green are the manifold distances. We see when the black lines are short, they are well-approximated by the green corresponding manifold distances. However, in (C) we observe that the manifold distance is not approximating the black line well, since the manifold curves.

Our data is observed Euclidean distances[2] e_{ij} for any two points. We can compute the same pairwise distance on the manifold s_{ij}. Now the question to ask could be whether e_{ij} could originate from the distribution of s_{ij}. To do so we need to study the distribution of Gaussian process arc lengths.

Let us first list a few of desiderata for our approach:

- A manifold is locally Euclidean, so *short* curves are approximately linear.
- Curves that are not short, are most likely non-Euclidean. Alternatively, long Euclidean distances, should not be short on the manifold.

[2]This is not a strict assumption, it can be non-Euclidean, but we will consider this for now.

In fact, we could say that any curve is longer on the manifold than in Euclidean distance. First, we will consider the distribution of s_{ij}, when the manifold is a GP. Then we will use censoring to enforce the topology.

3.4.1 Gaussian Process Arc Lengths

This section will present an approximation to the distribution of arc lengths of Gaussian processes. Naturally, this implies that we need to assume the underlying kernel to be smooth, as we know the Brownian motion (non-smooth kernel) has infinite arc lengths. The arc lengths of smooth GPs were studied by Bewsher et al. [2017] who found that these quantities could be approximated with the Nakagami distribution. We are interested in the distribution of the arc length (or curve length)

$$s = \int_0^1 \|\boldsymbol{J}(\boldsymbol{c}(t))\dot{\boldsymbol{c}}(t)\| dt, \tag{3.26}$$

where J is a Gaussian process[3]. Bewsher et al. [2017] start by studying the distribution of the integrand

$$\|\boldsymbol{J}(\boldsymbol{c}(t))\dot{\boldsymbol{c}}(t)\| = \sqrt{\sum_{d=1}^{D} \boldsymbol{J}_d(\boldsymbol{c}(t))\dot{\boldsymbol{c}}(t)\dot{\boldsymbol{c}}(t)^\top \boldsymbol{J}_d^\top(\boldsymbol{c}(t))}, \tag{3.27}$$

where $\boldsymbol{J}_d$ is the d-th row of the Jacobian matrix. For the remainder of this we will assume $\dot{\boldsymbol{c}}(t)$ to be constant — this is assuming the $\boldsymbol{c}(t)$ linearly interpolates between points in latent space. This will make analysis easier. This means the inner product

$$\boldsymbol{J}_d(\boldsymbol{c}(t)) \begin{pmatrix} 1 \\ 1 \end{pmatrix} \sim \mathcal{N}(\mu, S), \tag{3.28}$$

for some μ and S. One can then note that

$$\sum_{d=1}^{D} \boldsymbol{J}_d(\boldsymbol{c}(t))\dot{\boldsymbol{c}}(t)\dot{\boldsymbol{c}}(t)^\top \boldsymbol{J}_d^\top(\boldsymbol{c}(t)) = \boldsymbol{u}^\top \boldsymbol{u}, \tag{3.29}$$

for a Gaussian $D \times 1$ vector $\boldsymbol{u}$. Since it is a sum of squared Gaussian variables, Bewsher et al. [2017] suggest approximating it with a single Gamma variable.

On this basis, the *integrand* is *Nakagami* distributed, since it is the distribution of square root of Gamma variables. But what about the distribution of s? We can approximate the integral by a sum of *highly* correlated Nakagamis, where the correlation stems from the continuity of the GP. There is some work on this by Zlatanov et al. [2010]; they present a good approximation — especially for the very correlated scenario. We choose to approximate s with a Nakagami based on this, as the expression of Zlatanov et al. [2010] is not far from this.

[3]We assume it is the Jacobian of an implicit Gaussian process f

This provides no accurate form of the distribution of s, but it provides a solid motivation as to why we can use it for a likelihood function, when our data consists of distances and our aim is to model these distances with GPs. Now, we will detail the Nakagami and present the likelihood function.

Nakagami distribution. The Nakagami distribution [Nakagami, 1960] describes the length of an isotropic Gaussian vector. The density function is

$$g(s) = \frac{2m^m}{\Gamma(m)\Omega^m} s^{2m-1} \exp\Big(-\frac{m}{\Omega}s^2\Big), \qquad s \geq 0, \tag{3.30}$$

and it is parametrised by $m \geq \frac{1}{2}$ and $\Omega > 0$. Γ denotes the Gamma function. The parameters are interpretable by the equations

$$\Omega = \mathbb{E}[s^2] \quad \text{and} \quad m = \frac{\Omega^2}{\mathrm{Var}(s^2)}, \tag{3.31}$$

which can be used to infer the parameters through samples, although it does involve a fourth moment, yielding that it might not be sample-efficient.

Push-pull terms. We use (3.30) as our likelihood function, and if we feed in only Euclidean distances we would expect the manifold to recover these. Consequently, we would recover a probabilistic metric MDS, as we would *match* all the linear distances. We are interested in *non-linear* dimensionality reduction, and as listed in the desiderata above, we do not trust long Euclidean interpolations to reside on the manifold. However the long distances still provide some global information about the manifold, that we do not want to dismiss.

Carreira-Perpiñan [2010] noted that many modern non-linear dimensionality reduction methods consists of *push* and *pull* terms in their objective function. Pull terms to find the local patterns, and push terms to provide the global structure. In the next section, we will introduce censoring, which will assist us in preserving global structure. This will provide push terms to our method, while staying in the probabilistic framework.

3.4.2 Censoring

In this section, we will cover censoring and its role in manifold learning. *Censoring* is a statistical concept emerging from the field of survival analysis [Lee and Wang, 2003]. At a low level it is an informed variant of missing data: we do not observe a value, but we have information of where the observation could *not* be.

Assume we have observations $\{x_i\}_{i=1}^N$, originating from a distribution with cumulative distribution function G and density function g. It could be that some observations are not directly observed, but we observe their value is greater than some observed $\mathcal{C}_i$. We can augment the observation space in a way to encode this information: consider now

observations $\{(x_i, \delta_i)\}_{i=1}^N$, where

$$\delta_i = \begin{cases} 1 & \text{if } x_i \geq \mathcal{C}_i, \\ 0 & \text{if } x_i < \mathcal{C}_i, \end{cases} \tag{3.32}$$

i.e. an indicator function representing whether x_i was censored to the right by $\mathcal{C}_i$. Here, I will only consider *right-censoring*, but note that left-censoring occurs when the inequalities in (3.32) are mirrored. To make it clear, when $\delta_i = 1$, then our *observation* is the censoring value: $x_i = \mathcal{C}_i$.

Now we will formulate the likelihood in terms on the observations $\{(x_i, \delta_i)\}_{i=1}^N$. The marginal δ is binomial with probability parameter $p_\delta := \mathbb{P}(\delta_i = 1) = 1 - G(\mathcal{C}_i)$. We formulate the likelihood as

$$\mathcal{L} = \prod_{i=1}^N p\big((x_i, \delta_i)\big) = \prod_{i=1}^N p(x_i|\delta_i)p(\delta_i), \tag{3.33}$$

where we can split it up

$$\prod_{i=1}^N p(x_i|\delta_i)p(\delta_i) = \prod_{i:\delta_i=1} p(x_i|\delta_i = 1)\Big(1 - G(\mathcal{C}_i)\Big) \prod_{i:\delta_i=0} p(x_i|\delta_i = 0)G(\mathcal{C}_i). \tag{3.34}$$

The remaining conditional probabilities are defined as

$$p(x_i|\delta_i = 1) = 1 \qquad \text{and} \qquad p(x_i|\delta_i = 0) = \frac{g(x_i)}{G(\mathcal{C}_i)}, \tag{3.35}$$

because in the first case $x_i = \mathcal{C}_i$ by definition and in the second it is the definition of conditional density. This implies that (3.33) simplifies to

$$\mathcal{L} = \prod_{i:\delta_i=0} g(x_i) \prod_{i:\delta_i=1} \big(1 - G(\mathcal{C}_i)\big). \tag{3.36}$$

A likelihood function for isometric GPLVM. So far, we have argued the Nakagami distribution is well-suited to capture manifold distances and that censoring is a likelihood-based way of introducing *push* terms to keep global structure and avoid *matching* large distance exactly. We introduce the final likelihood for the isometric GPLVM here. Remember that e_{ij} denotes the observed (Euclidean) distance. Then for some $\epsilon > 0$ we define

$$L(\big\{\{e_{ij}\}_{i<j}\big\}_{i=1}^{N-1}|\theta, \epsilon) = \prod_{e_{ij}<\epsilon} g_\theta(e_{ij}) \prod_{e_{ij}\geq\epsilon} (1 - G_\theta(\epsilon)), \tag{3.37}$$

where g and G are the pdf and cdf, respectively, of the Nakagami distribution. $\theta = \{m, \Omega\}$ are the parameters of the Nakagami (3.31). These parameters are determined by the latent points $(\boldsymbol{z}_i, \boldsymbol{z}_j)$ and the GP $\boldsymbol{J}$ in a way yet to be described.

For completion we write the actual log-likelihood function up here

$$l\left(\left\{\{e_{ij}\}_{i<j}\right\}_{i=1}^{N-1}\middle|\theta,\epsilon\right) = -\sum_{e_{ij}<\epsilon}\Bigg(\log\Gamma\left(m_{ij}\right)+m_{ij}\log\left(\frac{\Omega_{ij}}{m_{ij}}\right) - \left(2m_{ij}-1\right)\log\left(e_{ij}\right)+\frac{m_{ij}e_{ij}^2}{\Omega_{ij}}\Bigg) \tag{3.38}$$
$$-\sum_{e_{ij}\geq\epsilon}\left(\log\Gamma\left(m_{ij}\right)-\log\left(\Gamma\left(m_{ij}\right)-\gamma(m_{ij},\frac{m_{ij}}{\Omega_{ij}}e_{ij}^2)\right)\right),$$

where Γ and γ denotes the Gamma function and lower incomplete gamma function, respectively. The remaining questions are: *how do we infer* θ_{ij} and *how do we set* ϵ?

3.4.3 Putting the model together

We infer the parameters of the Nakagami by introducing a latent Gaussian field J and a latent representation $\boldsymbol{z}$. The Nakagami was motivated as being the distribution of curve lengths on Gaussian fields, thus we may write

$$p(\theta|\boldsymbol{J},\boldsymbol{z}) := \int p(\theta|s)p(s|\boldsymbol{J},\boldsymbol{z})\mathrm{d}s, \quad \text{and} \quad p(\theta|s) = \begin{cases}\delta_{\mathbb{E}s^2}(\Omega)\\ \delta_{\frac{\Omega}{\mathrm{Var}(s^2)}}(m),\end{cases} \tag{3.39}$$

where δ denoted the Dirac probability measure and $p(s|\boldsymbol{J},\boldsymbol{z})$ is Nakagami (see (3.26)).

In practice, we will finely discretize the curve[4] $\boldsymbol{c}$, and approximate s as a sum. This is how we justified the Nakagami distribution. By sampling multiple $\boldsymbol{J}$'s we get multiple samples from s, and from this we can estimate the parameters $\theta = \{m,\Omega\}$ through second and fourth moment (see (3.31)).

We will infer everything with variational inference [Blei et al., 2017]. Hence, we choose a variational distribution over the variables to be marginalised. Let $\mathcal{E} := \{e_{ij}\}_{i<j\leq N}$ denote our observed distances. We approximate the posterior $p(\theta,\boldsymbol{J},\boldsymbol{z},\boldsymbol{u}|\mathcal{E})$ with

$$q(\theta,\boldsymbol{J},\boldsymbol{z},\boldsymbol{u}) := q(\theta|\boldsymbol{J},\boldsymbol{z})q(\boldsymbol{J},\boldsymbol{u})q(\boldsymbol{z}), \tag{3.40}$$

where $\boldsymbol{u}$ is an inducing variable [Titsias, 2009b], and

$$q(\theta|\boldsymbol{J},\boldsymbol{z}) = p(\theta|\boldsymbol{J},\boldsymbol{z}), \quad q(\boldsymbol{J},\boldsymbol{u}) = p(\boldsymbol{J}|\boldsymbol{u})q(\boldsymbol{u}) \quad \text{and} \quad q(\boldsymbol{z}) = \mathcal{N}(\boldsymbol{\mu}_z,\boldsymbol{A}_z), \tag{3.41}$$

where $\boldsymbol{\mu}_z$ is a vector of size N and $\boldsymbol{A}_z$ is a diagonal $N\times N$-matrix. Further $q(\boldsymbol{u}) = \mathcal{N}(\boldsymbol{\mu}_u,\boldsymbol{S})$ is a full M-dimensional Gaussian, with $M << N$.

From this we can compute the evidence lower bound to be

$$\log p(\mathcal{E}) = \log\int\frac{p(\mathcal{E},\theta,\boldsymbol{J},\boldsymbol{z},\boldsymbol{u})}{q(\theta,\boldsymbol{J},\boldsymbol{z},\boldsymbol{u})}q(\theta,\boldsymbol{J},\boldsymbol{z},\boldsymbol{u})\mathrm{d}\theta\mathrm{d}\boldsymbol{J}\mathrm{d}\boldsymbol{u}\mathrm{d}\boldsymbol{z} \tag{3.42}$$
$$\geq \mathbb{E}_\theta[l(\mathcal{E}|\theta)] - \mathrm{KL}\big(q(\boldsymbol{u})||p(\boldsymbol{u})\big) - \mathrm{KL}\big(q(\boldsymbol{z})||p(\boldsymbol{z})\big), \tag{3.43}$$

[4]For practical concerns, we only consider linear curves in latent space.

where (3.43) is the ELBO to be maximised.

The hyperparameter ϵ. The ϵ is the hyperparameter that determines the connectivity and thus also controls the non-linearity of the model. Its interpretation is much like that of IsoMap's neighbourhood graph, but we do not require that our underlying graph must be connected. We suggest using persistent homology (Section 3.2.2) to find a good value of ϵ, that captures the correct topology of the dataset.

Generative model. Jørgensen and Hauberg [2020] formulates a way to generate new data points even though the generative mapping f is implicit. However, this would require a *isometric* regression to immerse the learned manifold into $\mathbb{R}^D$. We found no efficient implementation of this, but some work in isometric regression has been done recently [Atzmon et al., 2020].

3.5 Empirical evaluation

We will inspect the presented model on the Open-Box dataset (Figure 3.4). The latent representation is visualised in Figure 3.9, where the points are the *means* of the latents $\boldsymbol{z}$, which have a Gaussian prior. In comparison with the IsoMap representation (Figure 3.6), the 'rounding' effect is not as prevailing here. Green and red seem more separated, but there are still some difficulties — especially where they meet the orange and cyan faces. The scaling of the axes are also different and should be noted. We see that the areas, that each face make up, are more comparable here, but interestingly in the case here, we can not trust our eyes too much when speaking of such geometric quantities.

Where IsoMap forces the latent space to be Euclidean, our method naturally learns the Riemannian structure of the latent space from the ambient space. Thus geodesics can not be trusted to be linear interpolators. We consider this behaviour on a higher dimensional, but well-known dataset.

The MNIST dataset contains 60000 greyscale images of handwritten digits. We consider a subset of 5000 of these (mostly for visualisation purposes), but every digit is represented in the subset. The dataset is 784-dimensional, one for each pixel in the images, our aim will be to represent it using two dimensions. To further point out key differences to the 'standard' GPLVM, we choose to use a different metric than the Euclidean for the ambient space $\mathbb{R}^{784}$. We consider a *lexicographic* metric [Rodriguez-Velazquez, 2018]

$$d_{\text{LEX}}(\boldsymbol{x}_i, \boldsymbol{x}_j) = \begin{cases} r, & \text{if } y_i \neq y_j, \\ \min\{2r,\ d(\boldsymbol{x}_i, \boldsymbol{x}_j)\}, & \text{if } y_i = y_j. \end{cases} \tag{3.44}$$

Notice, that this metric uses the *label* information at training time, but this trick can be used for other categorical variables to encode topology.

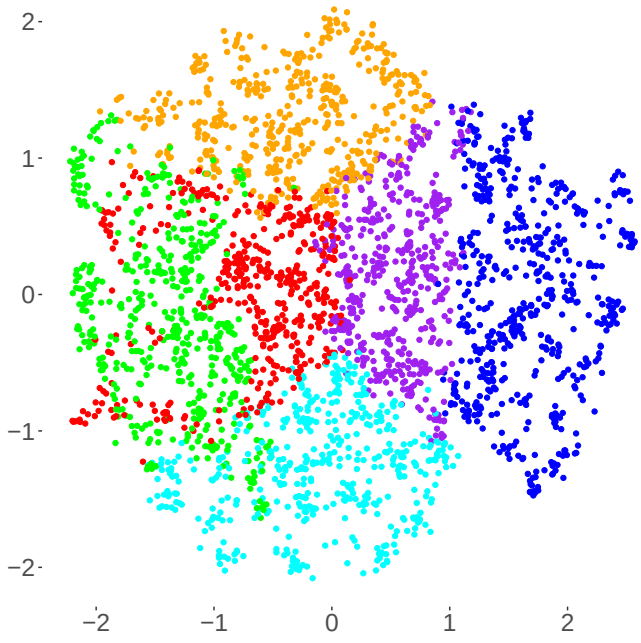


Figure 3.9: The latent representation of the Open Box dataset using the Iso-GPLVM. The points are the mean of the latents $\boldsymbol{z}$.

Figure 3.10 visualises the latent representation. The coloured dots are datapoints, colour coded by the integer in each image. To this end we see that the integers are accurately separated by the integers. The coloured lines are geodesics between points in latent space. Here, we observe that geodesics are in regions of high data density. The green geodesics, in particular, have comforting interpretations aligning with the desiderata of the approach. The 'interpolation' from images of 9's to 2's could have linear interpolated through 8's, but instead traverse them and stay in the 9's averting areas of high uncertainty.

On this note, the background colour is a good indicator of uncertain regions. It indicates the magnification factor (Eq. (3.24)), which we argued is a 'cost' of moving through some point. Further, the interpretation is coherent with the desiderata, that this magnification factor is small in the data regions and high in extrapolated area, encouraging geodesics to stay on the data manifold. Comparisons with other metrics and further baselines can be found in Jørgensen and Hauberg [2020].

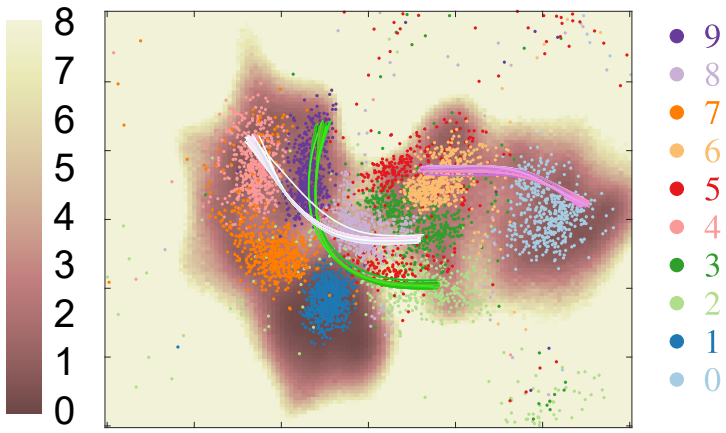


Figure 3.10: 2-dimensional representation of MNIST data. The coloured lines are geodesics computed using the magnification factor (background colour) of the *mean* manifold.

Future directions

One major bottleneck, also mentioned previously, is the actual optimisation for GPLVMs in general. They tend to be highly sensitive to initialisation. One idea to overcome this would be to use natural gradients. However, going forward we would replace the optimisation of the latents $\boldsymbol{z}$ with *amortised* inference. In a few words, amortised inference is a way of optimising over something else than the quantity we are conceptually optimising. For our needs, this would be exemplified by introducing a parametrized neural network, which would act as an encoder $\boldsymbol{x} -> \boldsymbol{z}$ and we would optimise the parameters of this network. The hope here would be that this approach has more flexibility. Usual methods, such as MDS, are known to be sensitive to initialisation too. We also recognise a need to establish a notion of *random geodesics*. In all experiments above, a geodesic is defined on *deterministic* manifolds — we simply compute them on the mean manifold. This has been shown to be a good approximation [Eklund and Hauberg, 2019b], under certain conditions which are perhaps a bit strict for many real-world settings. At this point, we are not settled on whether such a notion should be a distributional approach, by which we mean that a geodesic would actually be a *distribution* over curves in latent space. The alternative to this would be a single (deterministic) curve in the latent space, which would then satisfy something for the distribution of GP arcs in the ambient space $\mathbb{R}^D$.

On this note, the GP arc lengths were loosely *assumed* Nakagami distributed. More accurately, we chose the Nakagami likelihood as a 'best-fit' solution. This assumption, or choice, was based on theoretical findings in Bewsher et al. [2017]. However, it is not clear how tight the approximations here are. These findings are mostly dealing with very short curves, as they are dealing with the 'integrand distribution'. Empirically,

they briefly show that the Nakagami overshoots the variance of GP arc lengths, which is why we envision there can be attained tighter approximations, which potentially would benefit our model.

Lastly, the actual embedding of the manifold into ambient space, such that we are able to effortlessly generate new data points is non-trivial. We repeat from earlier, that we are looking for an isometric embedding, such that $f(\boldsymbol{z}) \approx \boldsymbol{x}$. Finding such an f, would make the model generative in the usual sense.

Chapter 4

Non-parametric Causal Discovery

So far in this thesis we have considered statistical models. In this chapter, we will change to *causal* models. How are they different? Classical statistical modelling cares about estimating probability distributions and learning the dependencies between random variables. In causal modelling, we are interested in modelling the data generating process. In more technical terms, say we have three random variables X, Y and Z, then statistical learning or modelling would try to infer the distribution $p(X, Y, Z)$. Contrarily, causal models encodes a hierarchy between the random variables, e.g. if X changes then Y changes, but the reverse does not necessarily hold. It could further be that if Y changes then Z changes, but again not the opposite. This constitutes the causal graph

$$X \to Y \to Z,$$

and we would say that X is a *direct* cause of Y, and an indirect cause of Z.

Causal graphs are usually encoded as Directed Acyclical Graphs (DAGs). In this framework, we will discuss causal relationship, both direct and indirect, and conditional independencies, which are fundamental when dealing with purely *observational* data. This allows us to create sparse graphs, that ultimately gives us information of the data generating process. We will briefly discuss structural equation models — a model class where each variable is a function of its parents in the DAG and some noise. For certain function classes, the underlying graph is said to be identifiable — which means we can recover the true causal relationship through

data — under this assumption that the functions belong to some particular model class.

A common phrase uttered in the communities involved with statistics is '*correlation does not imply causation*', and there exists a multitude of examples confirming this. However, sometimes there is a causal explanation; this triggers the question: can we *infer* what is the cause and what is the effect? In general, this is a hard question, and we will see that in fact in the bivariate case the answer is no. This will however not hold scientists back from trying anyway.

The contribution in this chapter is in this bivariate regime and is based on the paper *Reparametrization Invariance for non-parametric Causal Discovery* [Jørgensen and Hauberg, 2020]. It tries to make informed decisions to the bivariate causal query: is X the cause or the effect of Y? It takes base in the principle, that if X causes Y, then $f(X)$ causes $g(Y)$, where f and g are bijections.

4.1 Causality

To understand causality it is important to understand randomisation or experimental control. Say we are interested in two random variables X and Y and their causal relationship. Suppose the ground truth is that X causes Y, but this is unknown to the experimenter, whom we will call E. To investigate a first strategy could be to *observe* X and Y. This will provide us with data to estimate $p(X, Y)$, and the conditional distributions $p(X|Y)$ and $p(Y|X)$ and lastly also the marginals $p(X)$ and $p(Y)$. This situation is depicted in Figure 4.1A. The experimenter is here able to see that X and Y are dependent, but is unable to determine what is cause and effect. To infer this relationship, the experimenter has to *intervene*.

An *intervention* is when the experimenter is actively involved in the experiment. Say, the experimenter is able to fiddle with the experiment to ensure that $X = x$. Pearl [2009] denotes this $do(x)$. This situation is the one visualised in Figure 4.1B, where the experimenter intervenes on X. The data that is collected now is called *interventional* data, as opposed to *observational* data. This gives us data to estimate the distributions $p(Y|do(x))$. The experimenter will in this setting notice that $p(Y|do(x)) \neq p(Y|do(x'))$, which means that the behaviour in Y changes when we intervene on X, for $x \neq x'$.

Figure 4.1C depicts the situation where the experimenter intervenes on Y, i.e. $do(y)$. This intervention means that anything else that could have been the cause of Y, is no longer the cause of Y, since the experimenter is the cause $Y = y$ through the intervention. Now, the experimenter will observe that $p(X|do(y)) = p(X)$, i.e. that the interventional distribution $p(X|do(y))$ is no different than the marginal $p(X)$. Thus, our intervention on Y has no *effect* on X, hence Y can not be the cause of X. With these consideration, we can formalise the definition of cause and effect.

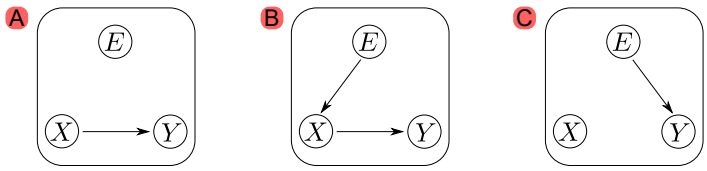


Figure 4.1: An experimenter E can choose to observe (situation A), intervene on X (situation B) or intervene on Y (situation C).

Definition 4.1 If for some $x \neq x'$, we have that $\mathbb{P}(Y|\text{do}(x)) \neq \mathbb{P}(Y|\text{do}(x'))$, then X is a cause of Y.

The observation also notes the following; if X causes Y then

(i) $p(X|do(y)) = p(X) \neq p(X|Y = y)$,
(ii) $p(Y|do(x)) = p(Y|X = x) \neq p(Y)$.

In practice, interventions are often impossible, expensive or ethically wrong. This raises the question, can we recover the causal relationship between variables based on only observational data? There is plenty of active research to this end, and the answer seems to be: *sometimes*. One thing, we can note from Definition 4.1 is that if X causes Y, this does not exclude that Y also causes X.

Direct causes and confounders. A *causal graph* is a graph that has random variables as its vertices and a directed edge between two vertices, say from X to Y, if and only if X is a *direct* cause of Y. Without formalism, X is a direct cause of Y if the relationship is not mediated through any other variable. In other words, if we fix all other variables (except X and Y) and intervene on X, this would still be detected in Y.

In fact, the building blocks in causal graphs can be phrased in triplets of variables (X, Y, Z). They are visualised in Figure 4.2. Figure 4.2A visualises what is often referred to as a *chain*, where the causal effect from X to Y is mediated by Z. Here Z is a direct cause of Y, while X is an indirect cause of Y, but a direct cause of Z. In 4.2B we see what is commonly referred to as a *fork*. Here, Z is known to be a *confounder* of X and Y. This situation is of particular importance, as in the real-world there will often exist unobserved or hidden confounders. Reichenbach and Reichenbach [1991] formalised the principle of common causes, that states if two variables X and Y correlates, then one is the cause of the other *or* there exists a third variable Z, which causes both of them. This paints a fuller picture than the tiresome expression 'correlation does not imply causation'.

Figure 4.2C pictures the last option, where Z is a so-called *collider*. Here both X and Y are causes of Z, but X and Y are at the same time independent of each other.

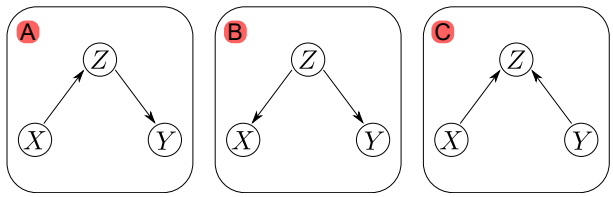


Figure 4.2: The 'building blocks' of causal graphs. A is a chain, B is a fork, and C is a collider.

Interestingly, they are not *conditionally* independent.

Conditional Independence. Two random variables X and Y are independent conditioned on a set of variables $\boldsymbol{Z}$ if and only if

$$p(X, Y|\boldsymbol{Z}) = p(X|\boldsymbol{Z})p(Y|\boldsymbol{Z}), \tag{4.1}$$

and notation-wise we write $X \perp\!\!\!\perp Y|\boldsymbol{Z}$. We will also consider d-separations, but we need to introduce some notions on graphs. Let G be a graph with vertices X_i, for $i = 1, \ldots, p$, and X_i are also random variables. Let E denote the edge set of the graph.

(i) A path in G is a sequence of distinct vertices $X_{i_1}, \ldots, X_{i_n}$ such that (i_j, i_{j+1}) or (i_{j+1}, i_j) is in E for all $1 \leq j < n$. The path is called directed if $(i_j, i_{j+1}) \in E$, for all $1 \leq j < n$.

(ii) A node X_i is a child of X_j if $(j, i) \in E$. X_j is called a parent of X_i. X_i is a descendant of X_j if X_i is a child of a descendant of X_j. Note the recursive definition.

(iii) A path between two nodes X_i and X_j is blocked by a disjoint subset of vertices $\boldsymbol{Z}$ if there is a node S in the path which satisfies one of

- $S \in \boldsymbol{Z}$ and S appears as a fork *or* a chain in the path (see above).
- S is a collider in the path, and none of its descendant, or S itself, is in $\boldsymbol{Z}$.

If $\boldsymbol{Z}$ blocks *all* paths between X_i and X_j, then $\boldsymbol{Z}$ is said to d-separate X_i and X_j.

(iv) G is a DAG if there is no pair of nodes, for which there exists directed paths 'in both directions'.

(v) The joint distribution $p(X_1, \ldots, X_p)$ is Markov with respect to the DAG G if, for disjoint subsets $\boldsymbol{X}, \boldsymbol{Y}, \boldsymbol{Z}$ of the vertices, satisfies

$$\boldsymbol{X} \text{ and } \boldsymbol{Y} \text{ are } d\text{-separated by } \boldsymbol{Z} \implies \boldsymbol{X} \perp\!\!\!\perp \boldsymbol{Y}|\boldsymbol{Z}.$$

(vi) The distribution is said to be faithful if the reverse implication is satisfied.

Faithfulness is an assumption we have to make and generally can not be tested or verified from the data directly [Zhang and Spirtes, 2008]. If faithfulness is not satisfied then identifiability through testing conditional independencies is not guaranteed.

PC Algorithm. The PC algorithm assumes the distribution is faithful to the underlying DAG. This yields the Markov equivalence class identifiable, and thus any query of d-separation can by inferred by corresponding conditional independence tests.

Spirtes et al. [2001] initialise the PC algorithm with a fully connected undirected graph and increase the size of the conditioning set $\boldsymbol{Z}$ at each iteration, initialising with the empty set. At iteration j, it considers conditional sets $\boldsymbol{Z}$ consisting of j variables. To see whether X and Y can be d-separated, one only needs to consider sets $\boldsymbol{Z}$ that are subsets either of the neighbours of X or of the neighbours of Y. This is computationally very appealing, especially for sparse graphs.

The output of the PC algorithm is not a DAG, since not all edges can have their direction determined. However, it can infer a lot a causal information in the arrows that are determined. The usefulness of DAGs is that sometimes it is possible to estimate *causal effects* through only observational data and conditional probabilities, as opposed to interventional distributions. We will not detail how this is done, as we will focus on the simpler setting of only two variables.

4.2 The bivariate causal discovery problem

In this section, we will consider the simplest of all DAGs - the bivariate DAG (the univariate is not too interesting). We observe that the joint distribution $p(X, Y)$ has two factorisations $p(X|Y)p(Y)$ and $p(Y|X)p(X)$, which corresponds, respectively, to the graphical models $Y \to X$ and $X \to Y$. From a statistical viewpoint, both of these are valid, but *at most* one of them can be causal, so how can we determine this? Naturally, we need interventional distributions to decide this (which introduces a third variable, the experimenter, see Figure 4.1). We need to approach the problem differently. Remember, that if $p(X|do(y)) = p(X)$, then Y is not the cause of X, and if $p(X|do(y)) = p(X|Y = y)$, it is.

We need some kind of way to break the symmetry of the joint distribution and its two factorisations. One way is to say, that X is a cause of Y if we can write the Y as a function f of X and some noise N_Y

$$Y := f(X, N_Y). \tag{4.2}$$

The issue is that we can *always* write this model — in both directions — and we can always find such a f. If we look to tighter definitions of the model class which f belongs to, we may break this symmetry. This is the approach of Structural Equation Models (SEMs), where we restrict f to have a certain structure. Under certain model

assumptions, we can show results on *identifiability*. That is, we can infer the causal direction from purely observational data. In this chapter, we will consider additive noise models (ANMs). Of course, this is a limitation that the model class where this is true is too narrow, and what happens if the data does not support such models? We note that identifiability within ANMs can be achieved, but this does not imply a *causal* identifiability.

The one thing we need to keep in mind for bivariate causal discovery is that *any* inference scheme is based on some interpretation of Occam's razor. We can not from observational data identify causal directions, thus we are forced to consider the *most likely* or *simplest* explanation based on some principles or experiences. It is these principles that break the symmetry of correlation.

Additive noise model. The hypothesis suggested by Hoyer et al. [2009] is that if the joint distribution $p(X, Y)$ satisfy an additive noise model from X to Y, then it is very likely that X is the cause of Y [Mooij et al., 2016]. An additive noise model from X to Y is defined as

$$Y = f(X) + N_Y, \quad \text{where } X \perp\!\!\!\perp N_Y. \tag{4.3}$$

We may without loss of generality assume $\mathbb{E}[N_Y] = 0$, but the important message here is that the 'supposed cause' X is independent of the noise N_Y.

Hoyer et al. [2009] operationalize this observation by performing Gaussian process regression on a training set, i.e. f is a GP and $Y = f(X) + N_Y$. Then they test for independence of X and N_Y using Hilbert-Schmidt Independence Criterion (HSIC) [Gretton et al., 2005] on the test set. This scheme is performed in both directions, $X \to Y$ and $Y \to X$, and they infer the causal direction as the one with the best HSIC score.

Next we will consider an approach, that makes the strong assumption of no noise.

Information-Geometric Causal Inference. Janzing et al. [2012] base their approach on the assumption the the marginal $p(X)$ contains no information about the conditional $p(Y|X)$, if X causes Y. Their method, *Information-Geometric Causal Inference* (IGCI), is based on the restriction that X and Y are related, in a *deterministic* way, by a bijection f, i.e. $Y = f(X)$ and $X = f^{-1}(Y)$.

To formalise what the 'no information' criterion is they consider the covariance of $\log f'(X)$ and $p(X)$, where it helps to think of these as random variables. More precisely, the covariance with respect to the uniform distribution

$$\mathrm{Cov}(\log f'(X), p(X)) = \int_0^1 \log f'(x)p(x)dx - \int_0^1 \log f'(x)dx \int_0^1 p(x)dx. \tag{4.4}$$

If this is 0, indicating no shared information of f (which is $p(Y|X)$ in the deterministic case) and $p(X)$, then the opposite direction is only 0 if f is linear. Hence, it breaks the symmetry of correlation.

Since $\int_0^1 \log f'(x)dx \leq \log \int_0^1 f'(x)dx = 0$ (wlog. for now assume f is strictly increasing on $[0,1]$), then the above covariance can only be 0 if

$$C_{X\to Y} := \int_0^1 \log f'(x)p(x)dx \leq 0. \tag{4.5}$$

Hence, to make an decision-making algorithm of this Janzing et al. [2012] estimates $C_{X\to Y}$ with

$$\hat{C}_{X\to Y} = \frac{1}{N-1}\sum_{i=1}^{N-1} \log \frac{|y_{i+1} - y - i|}{x_{i+1} - x_i}, \tag{4.6}$$

where x have been sorted such that $x_i < x_{i+1}$ for $i = 1, \ldots, N-1$. Further, both X and Y have been transformed such that their extrema lies on 0 and 1. Then, they infer $X \to Y$ if $\hat{C}_{X\to Y} < \hat{C}_{Y\to X}$.

The non-parametricity of this approach is appealing to our approach, but the assumption of no noise is too restricting for practical purposes. Although, IGCI have performed surprisingly good on the real-world dataset CEP [Mooij et al., 2016], a possible explanation for this is provided in Jørgensen and Hauberg [2020]. Our contribution, relies more on the next causal inference scheme.

Regression-Error based Causal Inference. Blöbaum et al. [2018] propose an inference scheme which measures the regression errors in an additive model

$$Y = f(X) + N_Y, \tag{4.7}$$

but they have no restriction on N_Y being independent of X, i.e. there can be heteroskedastic noise. They show, based on one key assumption which we will discuss shortly, that

$$\mathbb{E}[\mathrm{Var}(X|Y)] \geq \mathbb{E}[\mathrm{Var}(Y|X)], \tag{4.8}$$

under this assumption.

The assumption is the condition that

$$\mathrm{Cov}(f'(X), \mathbb{E}[\mathrm{Var}(Y|X)]p(X)) = 0. \tag{4.9}$$

Say X causes Y, then informally, this entails a form of independence between the causal marginal distribution X and the conditional $p(Y|X)$. Consider the case of constant noise $N_Y = \mathbb{E}[\mathrm{Var}(Y|X)]$, which would yield an ANM. Then the condition reduces to

$$\mathrm{Cov}(f'(X), p(X)) = 0, \tag{4.10}$$

which is very similar to the causal principle considered by IGCI. This is also what is formalised as the principle of *independent mechanisms* in Peters et al. [2017].

Equation (4.8) provides an actionable inference algorithm. Assume the additive noise model in both directions, and compute the regression errors N_Y and N_X. Here, we assume that both X and Y have been scaled for fair comparison (this is also assumed for their theoretical results). Then we can infer the most likely causal direction to be the one that gives the smallest regression error. This regression is done by different

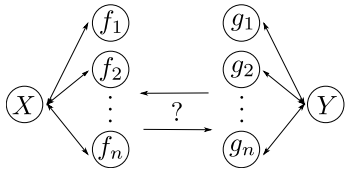


Figure 4.3: The invariance principle. If X is a cause of Y, then *theoretically* arrows from f_i to g_i should point to the right for *all* $i = 1, \ldots, n$. Inconsistency in these arrow, can be interpreted as uncertainty.

parametrized regression-types, e.g. polynomials or neural networks — it is not immediately clear, that this provide fair comparison of the regression error.

Unfortunately further, this inference is mostly stable when the noises are *small*, but it is an improvement over IGCI's 'no-noise assumption'.

4.2.1 Reparametrization Invariance

This section aims to outline the ideas presented in Jørgensen and Hauberg [2020]. The title of the section is informative for the fundamental principle on which the approach is based. We state that principle here.

A deterministic bijective reparametrization of the observed variables does not change the causal direction.

For bivariate data, this principle is also represented by the equivalence: X is the cause of Y if and only if $f(X)$ is the cause of $g(Y)$, where f and g are bijective functions.

The motivation for this principle can be phrased by the relationship of X and $f(X)$. $f(X)$ is a bijection of X, simultaneously X is a bijection of $f(X)$. If we choose to intervene on either, we would change the other — in reality an intervention on either would be an intervention on both. By Definition 4.1 we may then say that X is a cause of $f(X)$ *and* $f(X)$ is a cause of X. This would then imply that, that if X is a cause of Y, then $f(X)$ is a cause of Y too. Theoretically, $f(X)$ would be a indirect cause, but the deterministic relationship of X and $f(X)$ would make it practically direct. A symmetric argument would then yield the equivalence stated above, which is the principle exactly.

How can we operationalize this principle? Figure 4.3 indicate our approach, if for some f_i and g_j, where $i, j = 1, \ldots, n$, we have g_j is a cause of f_i, then X can not be a cause of Y. Likewise, if f_i is a cause of g_j, then Y can not be a cause of X, since X there is a descendant of Y in the 'causal' graph. Thus, the approach is to create n bijections

of both X and Y, and test the causal relationship among all these bijective copies of X and Y. If the decisions here are coherent, the causal link is likely strong. If they are sensitive to these bijections, this interprets as uncertainty in the causal estimator.

Non-parametric error estimator. The approach presented in Jørgensen and Hauberg [2020] aims to take a advantage of the invariant principle by the regression-error based inference. This choice was made as we, at first, can say that structural equation models, such as ANM, are not invariant under reparametrizations — a reparametrized ANM does not necessarily yield another ANM. In fact, this does not hold for any of the presented inference schemes, but we will empirically verify that the approach we present is more robust. The method we use differs from Blöbaum et al. [2018] in one fundamental way, beside the invariance, and that is *non-parametricity*.

Any parametrized form of regression can not guarantee to order the regression errors unanimously, when we take bijections on the marginals. We choose to compute the regression error non-parametrically by a simple sorting. We observe

$$\frac{1}{N-1}\sum_{i=1}^{N-1}(y_{i+1}-y_i)^2 \to 2\mathbb{E}\mathrm{Var}(Y|X), \tag{4.11}$$

for $N \to \infty$ and where we sort X, such that $x_1 \leq x_2 \leq \cdots \leq x_N$.

If we standardise both X and Y to have unit variance we can compute the *comparable* quantities $C_{X\to Y}$ and $C_{Y\to X}$ like

$$C_{X\to Y} := 1 - \frac{1}{2(N-1)}\sum_{i=1}^{N-1}\big(y_{i+1}-y_i\big)^2, \tag{4.12}$$

and based on Blöbaum et al. [2018] we can say that if $C_{X\to Y} < C_{Y\to X}$, then it is likely that X is the cause of Y. This provides a really efficient way to compute this quantity many times, and thus for many pairs of bijections. We present here the pseudo-code for the inference scheme and elaborate after.

Initialise Determine number of bijections n. Go to **Biject.**

Biject Sample two bijective functions f and g. Go to **Regress**.

Regress Perform the implicit regression, and evaluate the regression errors $C_{X\to Y}$ and $C_{Y\to X}$. If less than n iterations done, go to **Biject**; else go to **Confidence**.

Confidence Based on the n computations of $C_{X\to Y}$ and $C_{Y\to X}$, compute the confidence in the causal decisions, respectively.

In the above algorithm, we choose to perform n bijections of X and Y. We omit details on how the bijections are sampled — we refer to Jørgensen and Hauberg [2020] for

the details. We also already discussed the implicit regression and the computation of regression errors in (4.12). To discuss the last point — *confidence* — we introduce uncertainty associated with the causal decisions.

4.2.2 Uncertainty in decisions

In Figure 4.3 we argued how the n bijections can give indications to how unanimous the causal directions between X and Y are. This translates into an uncertainty — if all decisions are aligned, we are confident in the final causal decision. In the algorithm above, we thus associate a confidence in any causal decision. We define this as

$$\text{conf} := |p_x - 0.5|, \tag{4.13}$$

where p_x is the *probability* of $X \to Y$. This probability is computed based on all the values of $C_{X\to Y}$ and $C_{Y\to X}$ computed over all bijections. In Jørgensen and Hauberg [2020] we suggest to compute it as

$$p_x = \frac{1}{n}\sum_{i=1}^{n} \mathbf{1}\{C^{(i)}_{X\to Y} < C^{(i)}_{Y\to X}\}. \tag{4.14}$$

This would approximate the probability of the event $C_{X\to Y} < CY \to X$, where both C's have marginalised out the reparametrizations. We note that p_x has the least confidence when $p_x = 0.5$, which makes sense because $p_x + p_y = 1$. Thus the confidence as defined in (4.13) is associated with the joint distribution and not the causal decision $X \to Y$ — it measures only how confident we can be in the decision.

Evaluation. Mooij et al. [2016] present a simulated dataset, that is supposed to mimic *realistic* datasets, but we have a ground truth causal relationship, i.e. we know the data generating process. On the dataset we investigate here, it is actually generated to be an ANM. We are interested in the robustness of the different presented causal inference schemes when faced with reparametrization of the marginals. The dataset consists of 100 different pairs (X, Y), which can be read as the rows of the 'matrices' shown in Figure 4.4. Each pair consist of 1000 datapoints. The columns are then 20 bijections applied to each pair.Red indicates the inference scheme takes an incorrect decision on this parametrization. Black is a correct one. The figures can be thought of as 100×20-matrices, that we have colored out whether the decision is correct or not. In this setup, robustness would be measured in *full bars*; by which we mean rows that take one colour only, correct or not. On this dataset, MQV (*mean quadratic variation*, refers to the algorithm described above), is the most robust. RECI refers to Blöbaum et al. [2018]. ANM and IGCI are Hoyer et al. [2009] and Janzing et al. [2012], respectively.

On the qualitative performance of each of the methods we refer to Jørgensen and Hauberg [2020]. In a few words, most methods are competitive on the real world dataset, albeit IGCI is subpar on simulated datasets. The scarcity of real world data makes it difficult to accurate assess on the performance. In the next section, we cover

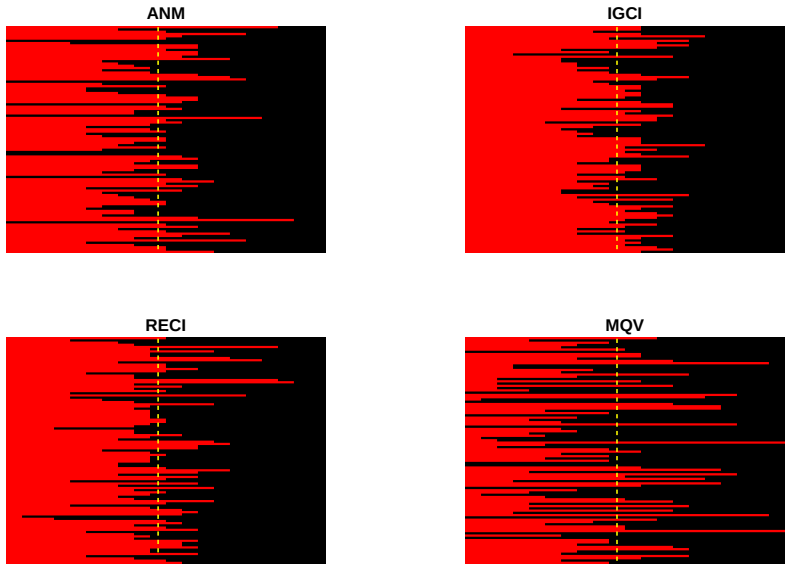


Figure 4.4: On the 100 pairs (rows) from benchmark [Mooij et al., 2016] simulated dataset, we applied 20 random bijections (columns). Above illustrates how the bijections influenced the decision. Red is an incorrect decision. MQV has 'fuller' bars, indicating that decisions are less influenced by bijections.

a simple extension of the non-parametric reparametrization approach to multivariate data.

4.2.3 Multivariate extension

An interesting feature of the reparametrization and regression-error approach is a connection to the usual multivariate DAG estimation. We can read this theorem [Jacod and Protter, 2000] as a reparametrization invariance.

Theorem 4.2 *Two random variables X and Y are independent if and only if*

$$Cov(f(X), g(Y)) = 0, \tag{4.15}$$

for any pair of functions f and g that are bounded and continuous.

Remember from earlier, that DAG estimation is closely linked to conditional independence testing, and we say that two variables X and Y and conditional independent on Z, if their conditional *covariance* is 0 for any reparametrization. The

conditional covariance can be estimation in an analogous way to the way we estimated the regression-error (or conditional variance). We observe

$$\frac{1}{8(N-1)} \sum_{i=1}^{N-1} \left(\left(s_{i+1} - s_i\right)^2 - \left(t_{i+1} - t_i\right)^2 \right) \to \mathbb{E}\mathrm{Cov}(X, Y|Z) \tag{4.16}$$

as $N \to \infty$. Here $s_i = x_i + y_i$ and $t_i = x_i - y_i$, and the indexing is determined by a sorting of Z, i.e. $z_1 \leq z_2 \leq \cdots \leq z_N$.

On this inspection, only a small tweak to the algorithm presented above changes it to a conditional independence test. In Jørgensen and Hauberg [2020] there is a toy example evaluation of this based on simulated data.

Future directions

The conditioning set in this last part is, as presented here, restricted to being one-dimensional because of the sorting. Thus, we can not use it as a conditional independence test in the PC algorithm. It would be an interesting future project to generalise this sorting constraint and perhaps use kernel methods to estimate the same quantity.

A major bottleneck for the bivariate causal estimation problem is the lack of real world data [Mooij et al., 2016]. This cast a large uncertainty over any empirical evaluation on the scarce data that exist.

Bibliography

L. Andreas and M. Kandemir. Differential Bayesian neural nets. *arXiv:1912.00796*, 2019.

S. Asmussen and P. W. Glynn. *Stochastic simulation: algorithms and analysis*, volume 57. Springer Science & Business Media, 2007.

M. Atzmon, A. Gropp, and Y. Lipman. Isometric autoencoders, 2020.

J. Bewsher, A. Tosi, M. Osborne, and S. Roberts. Distribution of gaussian process arc lengths. In *Artificial Intelligence and Statistics*, pages 1412–1420, 2017.

C. M. Bishop, M. Svens' en, and C. K. Williams. Magnification factors for the som and gtm algorithms. In *Proceedings 1997 Workshop on Self-Organizing Maps*, 1997.

D. M. Blei, A. Kucukelbir, and J. D. McAuliffe. Variational inference: A review for statisticians. *Journal of the American statistical Association*, 112(518):859–877, 2017.

P. Blöbaum, D. Janzing, T. Washio, S. Shimizu, and B. Schölkopf. Cause-effect inference by comparing regression errors. In *International Conference on Artificial Intelligence and Statistics*, pages 900–909, 2018.

M.-F. Bru. Wishart processes. *Journal of Theoretical Probability*, 4(4):725–751, 1991.

D. Burt, C. E. Rasmussen, and M. Van Der Wilk. Rates of convergence for sparse variational gaussian process regression. In *International Conference on Machine Learning*, pages 862–871, 2019.

R. Calandra, J. Peters, C. E. Rasmussen, and M. P. Deisenroth. Manifold gaussian processes for regression. In *2016 International Joint Conference on Neural Networks (IJCNN)*, pages 3338–3345. IEEE, 2016.

G. Carlsson. Topology and data. *Bulletin of the American Mathematical Society*, 46 (2):255–308, 2009.

M. Á. Carreira-Perpiñan. The elastic embedding algorithm for dimensionality reduction. In *Proceedings of the 27th International Conference on International Conference on Machine Learning*, pages 167–174, 2010.

F. Chazal and B. Michel. An introduction to topological data analysis: fundamental and practical aspects for data scientists, 2017.

T. Q. Chen, Y. Rubanova, J. Bettencourt, and D. K. Duvenaud. Neural ordinary differential equations. In *Advances in Neural Information Processing Systems*, 2018.

K. Cutajar, E. V. Bonilla, P. Michiardi, and M. Filippone. Random feature expansions for deep gaussian processes. In *International Conference on Machine Learning*, pages 884–893, 2017.

Z. Dai, A. Damianou, J. González, and N. Lawrence. Variational auto-encoded deep gaussian processes. *arXiv preprint arXiv:1511.06455*, 2015.

A. Damianou and N. D. Lawrence. Deep Gaussian processes. In *Artificial Intelligence and Statistics*, 2013.

S. Dasgupta and A. Gupta. An elementary proof of a theorem of johnson and lindenstrauss. *Random Structures & Algorithms*, 22(1):60–65, 2003.

N. S. Detlefsen, M. Jørgensen, and S. Hauberg. Reliable training and estimation of variance networks. In *33rd Conference on Neural Information Processing Systems*, 2019.

E. W. Dijkstra et al. A note on two problems in connexion with graphs. *Numerische mathematik*, 1(1):269–271, 1959.

M. M. Dunlop, M. A. Girolami, A. M. Stuart, and A. L. Teckentrup. How deep are deep gaussian processes? *The Journal of Machine Learning Research*, 19(1):2100–2145, 2018.

D. Duvenaud, O. Rippel, R. Adams, and Z. Ghahramani. Avoiding pathologies in very deep networks. In *Artificial Intelligence and Statistics*, 2014.

D. K. Duvenaud, H. Nickisch, and C. E. Rasmussen. Additive gaussian processes. In *Advances in neural information processing systems*, pages 226–234, 2011.

W. E. A proposal on machine learning via dynamical systems. *Communications in Mathematics and Statistics*, 5(1):1–11, 2017.

B. Efron. Bootstrap methods: Another look at the jackknife. *Ann. Statist.*, 7(1):1–26, 01 1979.

D. Eklund and S. Hauberg. Expected path length on random manifolds. *arXiv preprint arXiv:1908.07377*, 2019a.

D. Eklund and S. Hauberg. Expected path length on random manifolds. *arXiv preprint arXiv:1908.07377*, 2019b.

R. F. Engle. Autoregressive conditional heteroscedasticity with estimates of the variance of United Kingdom inflation. *Econometrica: Journal of the Econometric Society*, pages 987–1007, 1982.

R. W. Floyd. Algorithm 97: Shortest path. *Commun. ACM*, 5(6):345, June 1962.

A. Y. Foong, Y. Li, J. M. Hernández-Lobato, and R. E. Turner. 'in-between'uncertainty in bayesian neural networks. *arXiv preprint arXiv:1906.11537*, 2019.

C. Fu and D. Cai. Efanna : An extremely fast approximate nearest neighbor search algorithm based on knn graph. 09 2016.

Y. Gal and Z. Ghahramani. Dropout as a bayesian approximation: Representing model uncertainty in deep learning. In *International Conference on Machine Learning*, pages 1050–1059, 2016.

T. Gao and V. Jojic. Degrees of freedom in deep neural networks. In *Proceedings of the Thirty-Second Conference on Uncertainty in Artificial Intelligence*, pages 232–241, 2016.

A. Gelman, J. B. Carlin, H. S. Stern, D. B. Dunson, A. Vehtari, and D. B. Rubin. *Bayesian Data Analysis*. CRC Press, 2014.

P. Goovaerts et al. *Geostatistics for natural resources evaluation*. Oxford University Press on Demand, 1997.

A. Gretton, O. Bousquet, A. Smola, and B. Schölkopf. Measuring statistical dependence with hilbert-schmidt norms. *Algorithmic Learning Theory*, pages 63–78, 2005.

E. Haber and L. Ruthotto. Stable architectures for deep neural networks. *Inverse Problems*, 34(1):014004, 2017.

S. Hauberg. Only bayes should learn a manifold (on the estimation of differential geometric structure from data). *arXiv preprint arXiv:1806.04994*, 2018.

K. He, X. Zhang, S. Ren, and J. Sun. Deep residual learning for image recognition. In *Conference on Computer Vision and Pattern Recognition*, 2016.

C. Heaukulani and M. van der Wilk. Scalable Bayesian dynamic covariance modeling with variational Wishart and inverse Wishart processes. In *Advances in Neural Information Processing Systems*, 2019.

P. Hegde, M. Heinonen, H. Lähdesmäki, and S. Kaski. Deep learning with differential Gaussian process flows. In *Artificial Intelligence and Statistics*, 2019.

J. Hensman, A. G. d. G. Matthews, M. Filippone, and Z. Ghahramani. MCMC for variationally sparse Gaussian processes. In *Advances in Neural Information Processing Systems*, 2015.

J. M. Hernández-Lobato and R. Adams. Probabilistic backpropagation for scalable learning of bayesian neural networks. In *International Conference on Machine Learning*, pages 1861–1869, 2015.

T. K. Ho. Random decision forests. In *Proceedings of 3rd international conference on document analysis and recognition*, volume 1, pages 278–282. IEEE, 1995.

S. Hochreiter and J. Schmidhuber. Long short-term memory. *Neural computation*, 9 (8):1735–1780, 1997.

D. G. Horvitz and D. J. Thompson. A generalization of sampling without replacement from a finite universe. *Journal of the American Statistical Association*, 47(260): 663–685, 1952.

P. O. Hoyer, D. Janzing, J. M. Mooij, J. Peters, and B. Schölkopf. Nonlinear causal discovery with additive noise models. In D. Koller, D. Schuurmans, Y. Bengio, and L. Bottou, editors, *Advances in Neural Information Processing Systems 21*, pages 689–696. Curran Associates, Inc., 2009.

J. Jacod and P. Protter. *Probability Essentials*. Springer, New York, 2000.

D. Janzing, J. Mooij, K. Zhang, J. Lemeire, J. Zscheischler, P. Daniusis, B. Steudel, and B. Schölkopf. Information-geometric approach to inferring causal directions. *Artificial Intelligence*, (182-183):1–31, 2012.

M. Jørgensen and S. Hauberg. Isometric gaussian process latent variable model for dissimilarity data, 2020.

M. Jørgensen, M. P. Deisenroth, and H. Salimbeni. Stochastic differential equations with variational wishart diffusions. In *International Conference on Machine Learning*, 2020.

M. Jørgensen and S. Hauberg. Reparametrization invariance in non-parametric causal discovery, 2020.

P. E. Kloeden and E. Platen. *Numerical Solution of Stochastic Differential Equations*, volume 23. Springer Science & Business Media, 2013.

B. Lakshminarayanan, A. Pritzel, and C. Blundell. Simple and scalable predictive uncertainty estimation using deep ensembles. In *Advances in Neural Information Processing Systems*, pages 6402–6413, 2017.

K. L. Lange, R. J. A. Little, and J. M. G. Taylor. Robust statistical modeling using the t distribution. *Journal of the American Statistical Association*, 84(408):881–896, 1989.

N. Lawrence. Probabilistic non-linear principal component analysis with Gaussian process latent variable models. *Journal of machine learning research*, 6(Nov):1783–1816, 2005.

N. D. Lawrence and J. Quiñonero Candela. Local distance preservation in the gp-lvm through back constraints. In *Proceedings of the 23rd International Conference on Machine Learning*, ICML '06, page 513–520, New York, NY, USA, 2006. Association for Computing Machinery.

Y. LeCun and Y. Bengio. Convolutional networks for images, speech, and time series. In *The handbook of brain theory and neural networks*, pages 255–258. MIT Press, 1998.

E. T. Lee and J. Wang. *Statistical Methods for Survival Data Analysis*, volume 476. John Wiley & Sons, 2003.

J. A. Lee and M. Verleysen. *Nonlinear dimensionality reduction*. Springer Science & Business Media, 2007.

X. Li, T.-K. L. Wong, R. T. Q. Chen, and D. Duvenaud. Scalable gradients for stochastic differential equations. In *Artificial Intelligence and Statistics*, 2020.

X. Liu, T. Xiao, S. Si, Q. Cao, S. Kumar, and C.-J. Hsieh. Neural SDE: Stabilizing neural ODE networks with stochastic noise. *arXiv:1906.02355*, 2019.

D. J. MacKay. Bayesian neural networks and density networks. *Nuclear Instruments and Methods in Physics Research Section A: Accelerators, Spectrometers, Detectors and Associated Equipment*, 354(1):73–80, 1995.

D. Madras, J. Atwood, and A. D'Amour. Detecting extrapolation with influence functions. In *ICML 2019 Workshop on Uncertainty & Robustness in Deep Learning*, 2019.

B. B. Mandelbrot and J. W. V. Ness. Fractional brownian motions, fractional noises and applications. *SIAM Review*, 10(4):422–437, 1968.

A. Mead. Review of the development of multidimensional scaling methods. *Journal of the Royal Statistical Society: Series D (The Statistician)*, 41(1):27–39, 1992.

J. M. Mooij, J. Peters, D. Janzing, J. Zscheischler, and B. Schölkopf. Distinguishing cause from effect using observational data: Methods and benchmarks. *Journal of Machine Learning Research*, (17):1–102, 2016.

E. Nadaraya. On non-parametric estimates of density functions and regression curves. *Theory of Probability & Its Applications*, 10(1):186–190, 1965.

M. Nakagami. The m-distribution—a general formula of intensity distribution of rapid fading. In *Statistical Methods in Radio Wave Propagation*, pages 3–36. Elsevier, 1960.

D. Nix and A. Weigend. Estimating the mean and variance of the target probability distribution. In *Proc. 1994 IEEE Int. Conf. Neural Networks*, pages 55–60 vol.1. IEEE, 1994.

P. Orbanz. Lecture notes on bayesian nonparametrics. *Journal of Mathematical Psychology*, 56:1–12, 2012.

J. Pearl. *Causality: models, reasoning, and inference*. Cambridge University Press, 2009.

J. Peters, D. Janzing, and B. Schölkopf. *Elements of Causal Inference*. MIT Press, 2017.

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

C. E. Rasmussen and C. Williams. *Gaussian Processes for Machine Learning*. MIT Press, 2006.

H. Reichenbach and M. Reichenbach. *The Direction of Time*. University of California Press, 1991.

J. A. Rodriguez-Velazquez. Lexicographic metric spaces: Basic properties and the metric dimension, 2018.

T. Salimans and D. A. Knowles. Fixed-form variational posterior approximation through stochastic linear regression. *Bayesian Analysis*, 8(4):837–882, 2013.

H. Salimbeni and M. P. Deisenroth. Doubly stochastic variational inference for deep Gaussian processes. In *Advances in Neural Information Processing Systems*, 2017.

S. Särkkä and A. Solin. *Applied stochastic differential equations*, volume 10. Cambridge University Press, 2019.

M. Seeger, C. Williams, and N. Lawrence. Fast forward selection to speed up sparse gaussian process regression. Technical report, 2003.

M. Seeger, Y.-W. Teh, and M. Jordan. Semiparametric latent factor models. Technical report, 2005.

B. Settles. Active learning literature survey. Technical report, University of Wisconsin-Madison Department of Computer Sciences, 2009.

E. Snelson and Z. Ghahramani. Sparse Gaussian processes using pseudo-inputs. In *Advances in Neural Information Processing Systems*, 2006.

P. Spirtes, C. Glymour, R. Scheines, et al. Causation, prediction, and search. *MIT Press Books*, 1, 2001.

N. Srivastava, G. Hinton, A. Krizhevsky, I. Sutskever, and R. Salakhutdinov. Dropout: a simple way to prevent neural networks from overfitting. *The Journal of Machine Learning Research*, 15(1):1929–1958, 2014.

J. B. Tenenbaum, V. de Silva, and J. C. Langford. A Global Geometric Framework for Nonlinear Dimensionality Reduction. *Science*, 290(5500):2319, 2000.

M. Titsias. Variational learning of inducing variables in sparse Gaussian processes. In *Artificial Intelligence and Statistics*, 2009a.

M. Titsias. Variational learning of inducing variables in sparse gaussian processes. In *Artificial Intelligence and Statistics*, pages 567–574, 2009b.

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

A. Tosi, S. Hauberg, A. Vellido, and N. D. Lawrence. Metrics for Probabilistic Geometries. In *The Conference on Uncertainty in Artificial Intelligence (UAI)*, July 2014.

N. Twomey, M. Kozłowski, and R. Santos-Rodríguez. Neural ODEs with stochastic vector field mixtures. *arXiv:1905.09905*, 2019.

B. Tzen and M. Raginsky. Neural stochastic differential equations: deep latent Gaussian models in the diffusion limit. *arXiv:1905.09883*, 2019.

R. Urtasun, D. J. Fleet, A. Geiger, J. Popović, T. J. Darrell, and N. D. Lawrence. Topologically-constrained latent variable models. In *Proceedings of the 25th International Conference on Machine Learning*, pages 1080–1087, 2008.

I. Ustyuzhaninov, I. Kazlauskaite, C. H. Ek, and N. Campbell. Monotonic gaussian process flows. volume 108 of *Proceedings of Machine Learning Research*, pages 3057–3067, Online, 2020. PMLR.

A. G. Wilson and Z. Ghahramani. Generalised Wishart processes. *arXiv:1101.0240*, 2010.

A. G. Wilson, Z. Hu, R. Salakhutdinov, and E. P. Xing. Deep kernel learning. In *Artificial intelligence and statistics*, pages 370–378, 2016.

J. T. Wilson, V. Borovitskiy, A. Terenin, P. Mostowsky, and M. P. Deisenroth. Efficiently sampling functions from gaussian process posteriors. *arXiv preprint arXiv:2002.09309*, 2020.

J. Zhang and P. Spirtes. Detection of unfaithfulness and robust causal inference. *Minds and Machines*, 18(2):239–271, 2008.

S. Zhang, B. Guo, A. Dong, J. He, Z. Xu, and S. X. Chen. Cautionary tales on air-quality improvement in Beijing. *Proceedings of the Royal Society A: Mathematical, Physical and Engineering Sciences*, 473, 2017.

N. Zlatanov, Z. Hadzi-Velkov, and G. Karagiannidis. An efficient approximation to the correlated nakagami-m sums and its application in equal gain diversity receivers. *IEEE Transactions on Wireless Communications*, 9(1):302–310, Jan 2010.

Paper A

Reliable training and estimation of variance networks

Reliable training and estimation of variance networks

Nicki S. Detlefsen[*] [†]
nsde@dtu.dk

Martin Jørgensen[*] [†]
marjor@dtu.dk

Søren Hauberg [†]
sohau@dtu.dk

Abstract

We propose and investigate new complementary methodologies for estimating predictive variance networks in regression neural networks. We derive a locally aware mini-batching scheme that results in sparse robust gradients, and we show how to make unbiased weight updates to a variance network. Further, we formulate a heuristic for robustly fitting both the mean and variance networks post hoc. Finally, we take inspiration from posterior Gaussian processes and propose a network architecture with similar extrapolation properties to Gaussian processes. The proposed methodologies are complementary, and improve upon baseline methods individually. Experimentally, we investigate the impact of predictive uncertainty on multiple datasets and tasks ranging from regression, active learning and generative modeling. Experiments consistently show significant improvements in predictive uncertainty estimation over state-of-the-art methods across tasks and datasets.

1 Introduction

The quality of *mean* predictions has dramatically increased in the last decade with the rediscovery of neural networks [LeCun et al., 2015]. The predictive *variance*, however, has turned out to be a more elusive target, with established solutions being subpar. The general finding is that neural networks tend to make overconfident predictions [Guo et al., 2017] that can be harmful or offensive [Amodei et al., 2016]. This may be explained by neural networks being general function estimators that does not come with principled uncertainty estimates. Another explanation is that *variance* estimation is a fundamentally different task than *mean* estimation, and that the tools for mean estimation perhaps do not generalize. We focus on the latter hypothesis within regression.

To illustrate the main practical problems in variance estimation, we consider a toy problem where data is generated as $y = x \cdot \sin(x) + 0.3 \cdot \epsilon_1 + 0.3 \cdot x \cdot \epsilon_2$, with $\epsilon_1, \epsilon_2 \sim \mathcal{N}(0, 1)$ and x is uniform on $[0, 10]$ (Fig. 1). As is common, we do maximum likelihood estimation of $\mathcal{N}(\mu(x), \sigma^2(x))$, where μ and σ^2 are neural nets. While μ provides an almost perfect fit to the ground truth, σ^2 shows two problems: σ^2 is significantly underestimated and σ^2 does not increase outside the data support to capture the poor mean predictions.

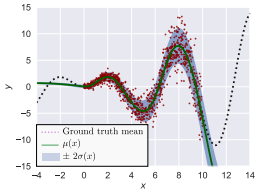


Figure 1: Max. likelihood fit of $\mathcal{N}(\mu(x), \sigma^2(x))$ to data.

These findings are general (Sec. 4), and alleviating them is the main purpose of the present paper. We find that this can be achieved by a combination of methods that *1)* change the usual mini-batching to be location aware; *2)* only optimize variance conditioned on the mean; *3)* for scarce data, we introduce a more robust likelihood function; and *4)* enforce well-behaved interpolation and extrapolation of variances. Points 1 and 2 are achieved through changes to the training algorithm, while 3 and 4 are changes to model specifications. We empirically demonstrate that these new tools significantly improve on state-of-the-art across datasets in tasks ranging from regression to active learning, and generative modeling.

[*] Equal contribution
[†] Section for Cognitive Systems, Technical University of Denmark

33rd Conference on Neural Information Processing Systems (NeurIPS 2019), Vancouver, Canada.

2 Related work

Gaussian processes (GPs) are well-known function approximators with built-in uncertainty estimators [Rasmussen and Williams, 2006]. GPs are robust in settings with a low amount of data, and can model a rich class of functions with few hyperparameters. However, GPs are computationally intractable for large amounts of data and limited by the expressiveness of a chosen kernel. Advances like sparse and deep GPs [Snelson and Ghahramani, 2006, Damianou and Lawrence, 2013] partially alleviate this, but neural nets still tend to have more accurate mean predictions.

Uncertainty aware neural networks model the predictive mean and variance as two separate neural networks, often as multi-layer perceptrons. This originates with the work of Nix and Weigend [1994] and Bishop [1994]; today, the approach is commonly used for making variational approximations [Kingma and Welling, 2013, Rezende et al., 2014], and it is this general approach we investigate.

Bayesian neural networks (BNN) [MacKay, 1992] assume a prior distribution over the network parameters, and approximate the posterior distribution. This gives direct access to the approximate predictive uncertainty. In practice, placing an informative prior over the parameters is non-trivial. Even with advances in stochastic variational inference [Kingma and Welling, 2013, Rezende et al., 2014, Hoffman et al., 2013] and expectation propagation [Hernández-Lobato and Adams, 2015], it is still challenging to perform inference in BNNs.

Ensemble methods represent the current state-of-the-art. *Monte Carlo (MC) Dropout* [Gal and Ghahramani, 2016] measure the uncertainty induced by Dropout layers [Hinton et al., 2012] arguing that this is a good proxy for predictive uncertainty. *Deep Ensembles* [Lakshminarayanan et al., 2017] form an ensemble from multiple neural networks trained with different initializations. Both approaches obtain ensembles of *correlated* networks, and the extent to which this biases the predictive uncertainty is unclear. Alternatives include estimating *confidence intervals* instead of variances [Pearce et al., 2018], and gradient-based Bayesian model averaging [Maddox et al., 2019].

Applications of uncertainty include *reinforcement learning*, *active learning*, and *Bayesian optimization* [Szepesvári, 2010, Huang et al., 2010, Frazier, 2018]. Here, uncertainty is the crucial element that allows for systematically making a trade-off between *exploration* and *exploitation*. It has also been shown that uncertainty is required to learn the topology of data manifolds [Hauberg, 2018].

The main categories of uncertainty are *epistemic* and *aleatoric* uncertainty [Kiureghian and Ditlevsen, 2009, Kendall and Gal, 2017]. Aleatoric uncertainty is induced by unknown or unmeasured features, and, hence, does not vanish in the limit of infinite data. Epistemic uncertainty is often referred to as *model uncertainty*, as it is the uncertainty due to model limitations. It is this type of uncertainty that Bayesian and ensemble methods generally estimate. We focus on the overall *predictive uncertainty*, which reflects both epistemic and aleatoric uncertainty.

3 Methods

The opening remarks (Sec. 1) highlighted two common problems that appear when μ and σ^2 are neural networks. In this section we analyze these problems and propose solutions.

Preliminaries. We assume that datasets $\mathcal{D} = \{\boldsymbol{x}_i, y_i\}_{i=1}^N$ contain i.i.d. observations $y_i \in \mathbb{R}, \boldsymbol{x}_i \in \mathbb{R}^D$. The targets y_i are assumed to be conditionally Gaussian, $p_\theta(y|\boldsymbol{x}) = \mathcal{N}(y|\mu(\boldsymbol{x}), \sigma^2(\boldsymbol{x}))$, where μ and σ^2 are continuous functions parametrized by $\theta = \{\theta_\mu, \theta_{\sigma^2}\}$. The maximum likelihood estimate (MLE) of the variance of i.i.d. observations $\{y_i\}_{i=1}^N$ is $\frac{1}{N-1}\sum_i (y_i - \hat{\mu})^2$, where $\hat{\mu}$ is the sample mean. This MLE does not exist based on a single observation, unless the mean μ is known, i.e. the mean is not a free parameter. When y_i is Gaussian, the residuals $(y_i - \mu)^2$ are gamma distributed.

3.1 A local likelihood model analysis

By assuming that both μ and σ^2 are continuous functions, we are implicitly saying that $\sigma^2(\boldsymbol{x})$ is correlated with $\sigma^2(\boldsymbol{x} + \delta)$ for sufficiently small δ, and similar for μ. Consider the local likelihood estimation problem [Loader, 1999, Tibshirani and Hastie, 1987] at a point $\boldsymbol{x}_i$,

$$\log \tilde{p}_\theta(y_i|\boldsymbol{x}_i) = \sum_{j=1}^N w_j(\boldsymbol{x}_i) \log p_\theta(y_j|\boldsymbol{x}_j), \tag{1}$$

where w_j is a function that declines as $\|\boldsymbol{x}_j - \boldsymbol{x}_i\|$ increases, implying that the local likelihood at $\boldsymbol{x}_i$ is dependent on the points nearest to $\boldsymbol{x}_i$. Notice $\tilde{p}_\theta(y_i|\boldsymbol{x}_i) = p_\theta(y_i|\boldsymbol{x}_i)$ if $w_j(\boldsymbol{x}_i) = \mathbf{1}_{i=j}$. Consider, with this w, a uniformly drawn subsample (i.e. a standard mini-batch) of the data $\{\boldsymbol{x}_k\}_{k=1}^M$ and its corresponding stochastic gradient of Eq. 1 with respect to θ_{σ^2}. If for a point, $\boldsymbol{x}_i$, no points near it are in the subsample, then no other point will influence the gradient of $\sigma^2(\boldsymbol{x}_i)$, which will point in the direction of the MLE, that is highly uninformative as it does not exist unless $\mu(\boldsymbol{x}_i)$ is known. Local data scarcity, thus, implies that while we have sufficient data for fitting a *mean*, locally we have insufficient data for fitting a *variance*. Essentially, if a point is isolated in a mini-batch, all information it carries goes to updating μ and none is present for σ^2.

If we do not use mini-batches, we encounter that gradients wrt. θ_μ and θ_{σ^2} will both be scaled with $\frac{1}{2\sigma^2(\boldsymbol{x})}$ meaning that points with small variances effectively have higher learning rates [Nix and Weigend, 1994]. This implies a bias towards low-noise regions of data.

3.2 Horvitz-Thompson adjusted stochastic gradients

We will now consider a solution to this problem within the local likelihood framework, which will give us a reliable, but biased, stochastic gradient for the usual (nonlocal) log-likelihood. We will then show how this can be turned into an unbiased estimator.

If we are to add some local information, giving more reliable gradients, we should choose a w in Eq.1 that reflects this. Assume for simplicity that $w_j(\boldsymbol{x}_i) = \mathbf{1}_{\|\boldsymbol{x}_i - \boldsymbol{x}_j\| < d}$ for some $d > 0$. The gradient of $\log \tilde{p}_\theta(y|\boldsymbol{x}_i)$ will then be informative, as more than one observation will contribute to the local variance if d is chosen appropriately. Accordingly, we suggest a practical mini-batching algorithm that samples a random point $\boldsymbol{x}_j$ and we let the mini-batch consist of the k nearest neighbors of $\boldsymbol{x}_j$.[3] In order to allow for more variability in a mini-batch, we suggest sampling m points uniformly, and then sampling n points among the k nearest neighbors of each of the m initially sampled points. Note that this is a more informative sample, as all observations in the sample are likely to influence the same subset of parameters in θ, effectively increasing the degrees of freedom[4], hence the quality of variance estimation. In other words, if the variance network is sufficiently expressive, our Monte Carlo gradients under this sampling scheme are of smaller variation and more sparse. In the supplementary material, we empirically show that this estimator yields significantly more sparse gradients, which results in improved convergence. Pseudo-code of this sampling-scheme, can be found in the supplementary material.

While such a mini-batch would give rise to an informative stochastic gradient, it would not be an unbiased stochastic gradient of the (nonlocal) log-likelihood. This can, however, be adjusted by using the *Horvitz-Thompson (HT)* algorithm [Horvitz and Thompson, 1952], i.e. rescaling the log-likelihood contribution of each sample $\boldsymbol{x}_j$ by its inclusion probability π_j. With this, an unbiased estimate of the log-likelihood (up to an additive constant) becomes

$$\sum_{i=1}^{N} \left\{ -\frac{1}{2}\log(\sigma^2(\boldsymbol{x}_i)) - \frac{(y_i - \mu(\boldsymbol{x}_i))^2}{2\sigma^2(\boldsymbol{x}_i)} \right\} \approx \sum_{\boldsymbol{x}_j \in \mathcal{O}} \frac{1}{\pi_j} \left\{ -\frac{1}{2}\log(\sigma^2(\boldsymbol{x}_j)) - \frac{(y_j - \mu(\boldsymbol{x}_j))^2}{2\sigma^2(\boldsymbol{x}_j)} \right\} \quad (2)$$

where $\mathcal{O}$ denotes the mini-batch. With the nearest neighbor mini-batching, the inclusion probabilities can be calculated as follows. The probability that observation j is in the sample is n/k if it is among the k nearest neighbors of one of the initial m points, which are chosen with probability m/N, i.e.

$$\pi_j = \frac{m}{N} \sum_{i=1}^{N} \frac{n}{k} \mathbf{1}_{j \in \mathcal{O}_k(i)}, \quad (3)$$

where $\mathcal{O}_k(i)$ denotes the k nearest neighbors of $\boldsymbol{x}_i$.

Computational costs The proposed sampling scheme requires an upfront computational cost of $O(N^2 D)$ before any training can begin. We stress that this is pre-training computation and not

[3] By convention, we say that the nearest neighbor of a point is the point itself.

[4] Degrees of freedom here refers to the parameters in a Gamma distribution – the distribution of variance estimators under Gaussian likelihood. Degrees of freedom in general is a quite elusive quantity in regression problems.

updated during training. The cost is therefore relative small, compared to training a neural network for small to medium size datasets. Additionally, we note that the search algorithm does not have to be precise, and we could therefore take advantage of fast approximate nearest neighbor algorithms [Fu and Cai, 2016].

3.3 Mean-variance split training

The most common training strategy is to first optimize θ_μ assuming a constant σ^2, and then proceed to optimize $\theta = \{\theta_\mu, \theta_{\sigma^2}\}$ jointly, i.e. a *warm-up* of μ. As previously noted, the MLE of σ^2 does not exist when only a single observation is available and μ is unknown. However, the MLE *does* exist when μ is known, in which case it is $\hat{\sigma}^2(\boldsymbol{x}_i) = (y_i - \mu(\boldsymbol{x}_i))^2$, assuming that the continuity of σ^2 is not crucial. This observation suggests that the usual training strategy is substandard as σ^2 is never optimized assuming μ is known. This is easily solved: we suggest to never updating μ and σ^2 simultaneously, i.e. only optimize μ conditioned on σ^2, and vice versa. This reads as sequentially optimizing $p_\theta(y|\theta_\mu)$ and $p_\theta(y|\theta_{\sigma^2})$, as we under these conditional distributions we may think of μ and σ^2 as known, respectively. We will refer to this as *mean-variance split training (MV)*.

3.4 Estimating distributions of variance

When $\sigma^2(\boldsymbol{x}_i)$ is influenced by few observations, underestimation is still likely due to the left skewness of the gamma distribution of $\hat{\sigma}_i^2 = (y_i - \mu(\boldsymbol{x}_i))^2$. As always, when in a low data regime, it is sensible to be Bayesian about it; hence instead of point estimating $\hat{\sigma}_i^2$ we seek to find a distribution. Note that we are not imposing a prior, we are training the parameters of a Bayesian model. We choose the inverse-Gamma distribution, as this is the conjugate prior of σ^2 when data is Gaussian. This means $\theta_{\sigma^2} = \{\theta_\alpha, \theta_\beta\}$ where $\alpha, \beta > 0$ are the shape and scale parameters of the inverse-Gamma respectively. So the log-likelihood is now calculated by integrating out σ^2

$$\log p_\theta(y_i) = \log \int \mathcal{N}(y_i|\mu_i, \sigma_i^2) \mathrm{d}\sigma_i^2 = \log t_{\mu_i, \alpha_i, \beta_i}(y_i), \tag{4}$$

where $\sigma_i^2 \sim \textsc{Inv-Gamma}(\alpha_i, \beta_i)$ and $\alpha_i = \alpha(\boldsymbol{x}_i), \beta_i = \beta(\boldsymbol{x}_i)$ are modeled as neural networks. Having an inverse-Gamma prior changes the predictive distribution to a located-scaled[5] Student-*t* distribution, parametrized with μ, α and β. Further, the *t*-distribution is often used as a replacement of the Gaussian when data is scarce and the true variance is unknown and yields a *robust* regression [Gelman et al., 2014, Lange et al., 1989]. We let α and β be neural networks that implicitly determine the degrees of freedom and the scaling of the distribution. Recall the higher the degrees of freedom, the better the Gaussian approximation of the t-distribution.

3.5 Extrapolation architecture

If we evaluate the local log-likelihood (Eq. 1) at a point $\boldsymbol{x}_0$ far away from all data points, then the weights $w_i(\boldsymbol{x}_0)$ will all be near (or exactly) zero. Consequently, the local log-likelihood is approximately 0 regardless of the observed value $y(\boldsymbol{x}_0)$, which should be interpreted as a large entropy of $y(\boldsymbol{x}_0)$. Since we are working with Gaussian and *t*-distributed variables, we can recreate this behavior by exploiting the fact that entropy is only an increasing function of the variance. We can re-enact this behavior by letting the variance tend towards an *a priori* determined value η if $\boldsymbol{x}_0$ tends away from the training data. Let $\{\mathbf{c}_i\}_{i=1}^L$ be points in $\mathbb{R}^D$ that represent the training data, akin to inducing points in sparse GPs [Snelson and Ghahramani, 2006]. Then define $\delta(\boldsymbol{x}_0) = \min_i \|\mathbf{c}_i - \boldsymbol{x}_0\|$ and

$$\hat{\sigma}^2(\boldsymbol{x}_0) = \big(1 - \nu(\delta(\boldsymbol{x}_0))\big)\hat{\sigma}_\theta^2 + \eta\nu(\delta(\boldsymbol{x}_0)), \tag{5}$$

where $\nu : [0, \infty) \mapsto [0, 1]$ is a surjectively increasing function. Then the variance estimate will go to η as $\delta \to \infty$ at a rate determined by ν. In practice, we choose ν to be a scaled-and-translated sigmoid function: $\nu(x) = \mathrm{sigmoid}((x + a)/\gamma)$, where γ is a free parameter we optimize during training and $a \approx -6.9077\gamma$ to ensure that $\nu(0) \approx 0$. The inducing points $\mathbf{c}_i$ are initialized with k-means and optimized during training. This choice of architecture is similar to that attained by posterior Gaussian processes when the associated covariance function is stationary. It is indeed the behavior of these established models that we aim to mimic with Eq. 5.

[5]This means $y \sim F$, where $F = \mu + \sigma t(\nu)$. The explicit density can be found in the supplementary material.

4 Experiments

4.1 Regression

To test our methodologies we conduct multiple experiments in various settings. We compare our method to state-of-the-art methods for quantifying uncertainty: Bayesian neural network (BNN) [Hernández-Lobato and Adams, 2015], Monte Carlo Dropout (MC-Dropout) [Gal and Ghahramani, 2016] and Deep Ensembles (Ens-NN) [Lakshminarayanan et al., 2017]. Additionally we compare to two baseline methods: standard mean-variance neural network (NN) [Nix and Weigend, 1994] and GPs (sparse GPs (SGP) when standard GPs are not applicable) [Rasmussen and Williams, 2006]. We refer to our own method(s) as *Combined*, since we apply all the methodologies described in Sec. 3. Implementation details and code can be found in the supplementary material. Strict comparisons of the models should be carefully considered; having two seperate networks to model mean and variance seperately (as NN, Ens-NN and Combined) means that all the predictive uncertainty, *i.e.* both aleatoric and episteminc, is modeled by the variance networks alone. BNN and MC-Dropout have a higher emphasis on modeling epistemic uncertainty, while GPs have the cleanest separation of noise and model uncertainty estimation. Despite the methods quantifying different types of uncertainty, their results can still be ranked by test set log-likelihood, which is a proper scoring function.

Toy regression. We first return to the toy problem of Sec. 1, where we consider 500 points from $y = \boldsymbol{x} \cdot \sin(\boldsymbol{x}) + 0.3 \cdot \epsilon_1 + 0.3 \cdot \boldsymbol{x} \cdot \epsilon_2$, with $\epsilon_1, \epsilon_2 \sim \mathcal{N}(0, 1)$. In this example, the variance is *heteroscedastic*, and models should estimate larger variance for larger values of $\boldsymbol{x}$. The results[6] can be seen in Figs. 2 and 3. Our approach is the only one to satisfy all of the following: capture the heteroscedasticity, extrapolate high variance outside data region and not underestimating within.

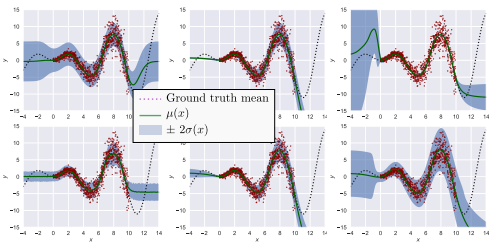


Figure 2: From top left to bottom right: GP, NN, BNN, MC-Dropout, Ens-NN, Combined.

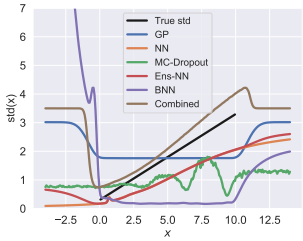


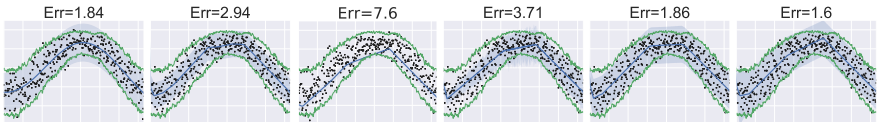
Figure 3: Standard deviation estimates as a function of $\boldsymbol{x}$.

Variance calibration. To our knowledge, no benchmark for quantifying variance estimation exists. We propose a simple dataset with known uncertainty information. More precisely, we consider weather data from over 130 years.[7] Each day the maximum temperature is measured, and the uncertainty is then given as the variance in temperature over the 130 years. The fitted models can be seen in Fig. 4. Here we measure performance by calculating the mean error in uncertainty: $\text{Err} = \frac{1}{N} \sum_{i=1}^{N} |\sigma^2_{\text{true}}(\boldsymbol{x}_i) - \sigma^2_{\text{est}}(\boldsymbol{x}_i)|$. The numbers are reported above each fit. We observe that our Combined model achieves the lowest error of all the models, closely followed by Ens-NN and GP. Both NN, BNN and MC-Dropout all severely underestimate the uncertainty.

Ablation study. To determine the influence of each methodology from Sec. 3, we experimented with four UCI regression datasets (Fig. 5). We split our contributions in four: the locality sampler (LS), the mean-variance split (MV), the inverse-gamma prior (IG) and the extrapolating architecture (EX). The combined model includes all four tricks. The results clearly shows that LS and IG methodologies has the most impact on test set log likelihood, but none of the methodologies perform worse than the baseline model. Combined they further improves the results, indicating that the proposed methodologies are complementary.

[6] The standard deviation plotted for *Combined*, is the root mean of the inverse-Gamma.

[7] `https://mrcc.illinois.edu/CLIMATE/Station/Daily/StnDyBTD2.jsp`



(a) GP (b) NN (c) BNN (d) MC-Dropout (e) Ens-NN (f) Combined

Figure 4: Weather data with uncertainties. Dots are datapoints, green lines are the true uncertainty, blue curves are mean predictions and the blue shaded areas are the estimated uncertainties.

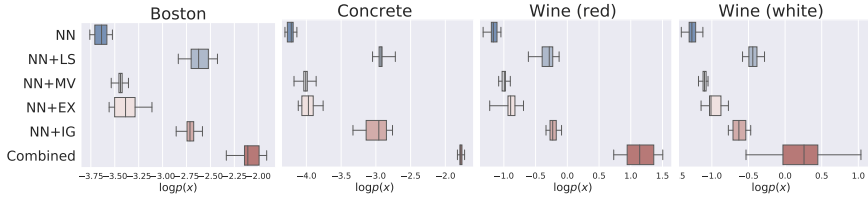


Figure 5: The complementary methodologies from Sec. 3 evaluated on UCI benchmark datasets.

UCI benchmark. We now follow the experimental setup from Hernández-Lobato and Adams [2015], by evaluating models on a number of regression datasets from the UCI machine learning database. Additional to the standard benchmark, we have added 4 datasets. Test set log-likelihood can be seen in Table 1, and the corresponding RMSE scores can be found in the supplementary material.

Our *Combined* model performs best on 10 of the 13 datasets. For the small *Boston* and *Yacht* datasets, the standard GP performs the best, which is in line with the experience that GPs perform well when data is scarce. On these datasets our model is the best-performing neural network. On the *Energy* and *Protein* datasets Ens-NN perform the best, closely followed by our Combined model. One clear advantage of our model compared to Ens-NN is that we only need to train one model, whereas Ens-NN need to train 5+ (see the supplementary material for training times for each model). The worst performing model in all cases is the baseline NN model, which clearly indicates that the usual tools for *mean* estimation does not carry over to *variance* estimation.

Active learning. The performance of active learning depends on predictive uncertainty [Settles, 2009], so we use this to demonstrate the improvements induced by our method. We use the same network architectures and datasets as in the UCI benchmark. Each dataset is split into: 20% train, 60% pool and 20% test. For each active learning iteration, we first train a model, evaluate the performance on the test set and then estimate uncertainty for all datapoints in the pool. We then select the n points with highest variance (corresponding to highest entropy [Houlsby et al., 2012]) and add these to the

	N	D	GP	SGP	NN	BNN	MC-Dropout	Ens-NN	Combined
Boston	506	13	$\mathbf{-1.76 \pm 0.3}$	-1.85 ± 0.25	-3.64 ± 0.09	-2.59 ± 0.11	-2.51 ± 0.31	-2.45 ± 0.25	-2.09 ± 0.09
Carbon	10721	7	-	3.74 ± 0.53	-2.03 ± 0.14	-1.1 ± 1.76	-1.08 ± 0.05	-0.44 ± 7.28	$\mathbf{4.35 \pm 0.16}$
Concrete	1030	8	-2.13 ± 0.14	-2.29 ± 0.12	-4.23 ± 0.07	-3.31 ± 0.05	-3.11 ± 0.12	-3.06 ± 0.32	$\mathbf{-1.78 \pm 0.04}$
Energy	768	8	-1.85 ± 0.34	-2.22 ± 0.15	-3.78 ± 0.04	-2.07 ± 0.08	-2.01 ± 0.11	$\mathbf{-1.48 \pm 0.31}$	-1.68 ± 0.13
Kin8nm	8192	8	-	2.01 ± 0.02	-0.08 ± 0.02	0.95 ± 0.08	0.95 ± 0.15	1.18 ± 0.03	$\mathbf{2.49 \pm 0.07}$
Naval	11934	16	-	-	3.47 ± 0.21	3.71 ± 0.05	3.80 ± 0.09	5.55 ± 0.05	$\mathbf{7.27 \pm 0.13}$
Power plant	9568	4	-	-1.9 ± 0.03	-4.26 ± 0.14	-2.89 ± 0.01	-2.89 ± 0.14	-2.77 ± 0.04	$\mathbf{-1.19 \pm 0.03}$
Protein	45730	9	-	-	-2.95 ± 0.09	-2.91 ± 0.00	-2.93 ± 0.14	$\mathbf{-2.80 \pm 0.02}$	$\mathbf{-2.83 \pm 0.05}$
Superconduct	21263	81	-	-4.07 ± 0.01	-4.92 ± 0.10	-3.06 ± 0.14	-2.91 ± 0.19	-3.01 ± 0.05	$\mathbf{-2.43 \pm 0.05}$
Wine (red)	1599	11	0.96 ± 0.18	-0.08 ± 0.01	-1.19 ± 0.11	-0.98 ± 0.01	-0.94 ± 0.01	-0.93 ± 0.09	$\mathbf{1.21 \pm 0.23}$
Wine (white)	4898	11	-	-0.14 ± 0.05	-1.29 ± 0.09	-1.41 ± 0.17	-1.26 ± 0.01	-0.99 ± 0.06	$\mathbf{0.40 \pm 0.42}$
Yacht	308	7	$\mathbf{0.16 \pm 1.22}$	-0.38 ± 0.32	-4.12 ± 0.17	-1.65 ± 0.05	-1.55 ± 0.12	-1.18 ± 0.21	$\mathbf{-0.07 \pm 0.05}$
Year	515345	90	-	-	-5.21 ± 0.87	-3.97 ± 0.34	-3.78 ± 0.01	-3.42 ± 0.02	$\mathbf{-3.01 \pm 0.14}$

Table 1: Dataset characteristics and tests set log-likelihoods for the different methods. A - indicates the model was infeasible to train. Bold highlights the best results.

training set. We set $n = 1\%$ of the initial pool size. This is repeated 10 times, such that the last model is trained on 30%. We repeat this on 10 random training-test splits to compute standard errors.

Fig. 6,show the evolution of average RMSE for each method during the data collection process for the *Boston*, *Superconduct* and *Wine (white)* datasets (all remaining UCI datasets are visualized in the supplementary material). In general, we observe two trends. For some datasets we observe that our *Combined* model outperforms all other models, achieving significantly faster learning. This indicates that our model is better at predicting the uncertainty of the data in the pool set. On datasets where the sampling process does not increase performance, we are on par with other models.

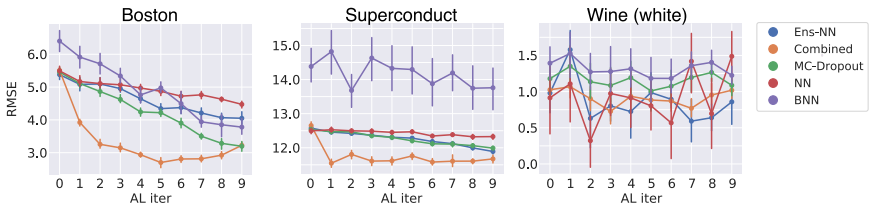


Figure 6: Average test set RMSE and standard errors in active learning. The remaining datasets are shown in the supplementary material.

4.2 Generative models

To show a broader application of our approach, we also explore it in the context of generative modeling. We focus on variational autoencoders (VAEs) [Kingma and Welling, 2013, Rezende et al., 2014] that are popular deep generative models. A VAE model the generative process:

$$p(\boldsymbol{x}) = \int p_\theta(\boldsymbol{x}|\boldsymbol{z})p(\boldsymbol{z})\mathrm{d}\boldsymbol{z}, \quad p_\theta(\boldsymbol{x}|\boldsymbol{z}) = \mathcal{N}\big(\boldsymbol{x}|\mu_\theta(\boldsymbol{z}), \sigma^2_\theta(\boldsymbol{z})\big) \quad \text{or} \quad p_\theta(\boldsymbol{x}|\boldsymbol{z}) = \mathcal{B}\big(\boldsymbol{x}|\mu_\theta(\boldsymbol{z})\big), \quad (6)$$

where $p(\boldsymbol{z}) = \mathcal{N}(\mathbf{0}, \mathbb{I}_d)$. This is trained by introducing a variational approximation $q_\phi(\boldsymbol{z}|\boldsymbol{x}) = \mathcal{N}(\boldsymbol{z}|\mu_\phi(\boldsymbol{x}), \sigma^2_\phi(\boldsymbol{x}))$ and then jointly training p_θ and q_ϕ. For our purposes, it suffcient to note that a VAE estimates both a mean and a variance function. Thus using standard training methods, the same problems arise as in the regression setting. Mattei and Frellsen [2018] have recently shown that estimating a VAE is ill-posed unless the variance is bounded from below. In the literature, we often find that

1. Variance networks are avoided by using a Bernoulli distribution, even if data is not binary.

2. Optimizing VAEs with a Gaussian posterior is considerably harder than the Bernoulli case. To overcome this, the variance is often set to a constant *e.g.* $\sigma^2(\boldsymbol{z}) = 1$. The consequence is that the log-likelihood reconstruction term in the ELBO collapses into an L2 reconstruction term.

3. Even though the generative process is given by Eq. 6, samples shown in the literature are often reduced to $\tilde{\boldsymbol{x}} = \mu(\boldsymbol{z}), \boldsymbol{z} \sim \mathcal{N}(\mathbf{0}, \mathbb{I})$. This is probably due to the wrong/meaningless variance term.

We aim to fix this by training the posterior variance $\sigma^2_\theta(\boldsymbol{z})$ with our Combined method. We do not change the encoder variance $\sigma^2_\phi(\boldsymbol{x})$ and leave this to future study.

Artificial data. We first evaluate the benefits of more reliable variance networks in VAEs on artificial data. We generate data inspired by the two moon dataset[8], which we map into four dimensions. The mapping is thoroughly described in the supplementary material, and we emphasize that we have deliberately used mappings that MLP's struggle to learn, thus with a low capacity network the only way to compensate is to learn a meaningful variance function.

In Fig. 7 we plot pairs of output dimensions using 5000 generated samples. For all pairwise combinations we refer to the supplementary material. We observe that samples from our Comb-VAE capture the data distribution in more detail than a standard VAE. For VAE the variance seems to be

[8] `https://scikit-learn.org/stable/modules/generated/sklearn.datasets.make_moons.html`

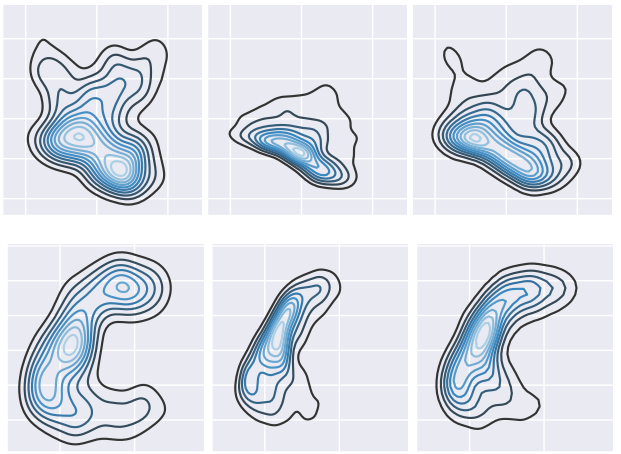


Figure 7: The ground truth and generated distributions. *Top*: x_1 vs. x_2. *Bottom*: x_2 vs x_3.

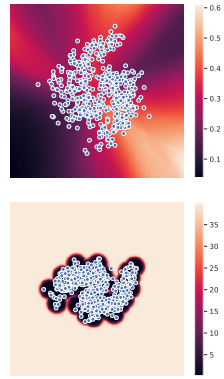


Figure 8: Variance estimates in latent space for standard VAE (top) and our Comb-VAE (bottom). Blue points are the encoded training data.

		MNIST	FashionMNIST	CIFAR10	SVHN
ELBO	VAE	2053.01 ± 1.60	1506.31 ± 2.71	1980.84 ± 3.32	3696.35 ± 2.94
	Comb-VAE	$\mathbf{2152.31 \pm 3.32}$	$\mathbf{1621.29 \pm 7.23}$	$\mathbf{2057.32 \pm 8.13}$	3701.41 ± 5.84
$\log p(x)$	VAE	1914.77 ± 2.15	1481.38 ± 3.68	1809.43 ± 10.32	3606.28 ± 2.75
	Comb-VAE	$\mathbf{2018.37 \pm 4.35}$	$\mathbf{1567.23 \pm 4.82}$	$\mathbf{1891.39 \pm 20.21}$	3614.39 ± 7.91

Table 2: Generative modeling of 4 datasets. For each dataset we report training ELBO and test set log-likelihood. The standard errors are calculated over 3 trained models with random initialization.

underestimated, which is similar to the results from regression. The poor sample quality of a standard VAE can partially be explained by the arbitrariness of decoder variance function $\sigma^2(\boldsymbol{z})$ away from data. In Fig. 8, we calculated the accumulated variance $\sum_{j=1}^{D} \sigma_j^2(\boldsymbol{z})$ over a grid of latent points. We clearly see that for the standard VAE, the variance is low where we have data and arbitrary away from data. However, our method produces low-variance region where the two half moons are and a high variance region away from data. We note that Arvanitidis et al. [2018] also dealt with the problem of arbitrariness of the decoder variance. However their method relies on post-fitting of the variance, whereas ours is fitted during training. Additionally, we note that [Takahashi et al., 2018] also successfully modeled the posterior of a VAE as a Student t-distribution similar to our proposed method, but without the extrapolation and different training procedure.

Image data. For our last set of experiments we fitted a standard VAE and our Comb-VAE to four datasets: MNIST, FashionMNIST, CIFAR10, SVHN. We want to measure whether there is an improvement to generative modeling by getting better variance estimation. The details about network architecture and training can be found in the supplementary material. Training set ELBO and test set log-likelihoods can be viewed in Table 2. We observe on all datasets that, on average tighter bounds and higher log-likelihood are achieved, indicating that we better fit the data distribution. We quantitatively observe (see Fig. 9) that variance has a more local structure for Comb-VAE and that the variance reflects the underlying latent structure.

5 Discussion & Conclusion

While variance networks are commonly used for modeling the predictive uncertainty in regression and in generative modeling, there have been no systematic studies of how to fit these to data. We have demonstrated that tools developed for fitting *mean* networks to data are subpar when applied to

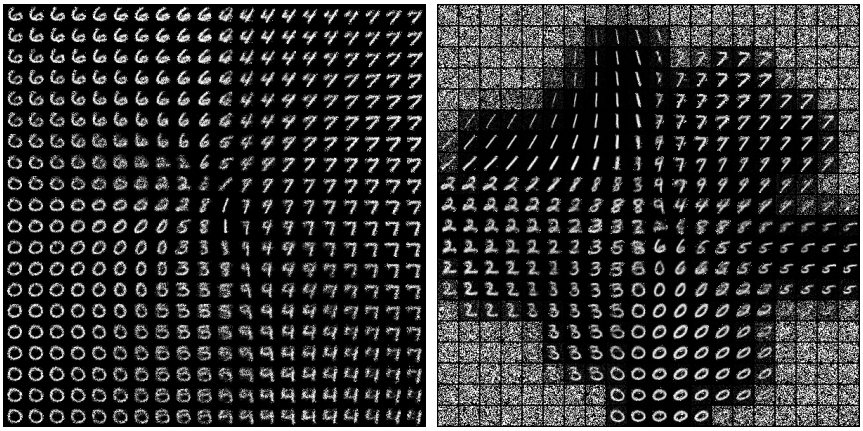


Figure 9: Generated MNIST images on a grid in latent space using the standard variance network (left) and proposed variance network (right).

variance estimation. The key underlying issue appears to be that it is not feasible to estimate both a mean and a variance at the same time, when data is scarce.

While it is beneficial to have separate estimates of both *epistemic* and *aleatoric* uncertainty, we have focused on *predictive uncertainty*, which combine the two. This is a lesser but more feasible goal.

We have proposed a new mini-batching scheme that samples locally to ensure that variances are better defined during model training. We have further argued that variance estimation is more meaningful when conditioned on the mean, which implies a change to the usual training procedure of joint mean-variance estimation. To cope with data scarcity we have proposed a more robust likelihood that model a distribution over the variance. Finally, we have highlighted that variance networks need to extrapolate differently from mean networks, which implies architectural differences between such networks. We specifically propose a new architecture for variance networks that ensures similar variance extrapolations to posterior Gaussian processes from stationary priors.

Our methodologies depend on algorithms that computes Euclidean distances. Since these often break down in high dimensions, this indicates that our proposed methods may not be suitable for high dimensional data. Since we mostly rely on nearest neighbor computations, that empirical are known to perform better in high dimensions, our methodologies may still work in this case. Interestingly, the very definition of variance is dependent on Euclidean distance and this may indicate that variance is inherently difficult to estimate for high dimensional data. This could possible be circumvented through a learned metric.

Experimentally, we have demonstrated that proposed methods are complementary and provide significant improvements over state-of-the-art. In particular, on benchmark data we have shown that our method improves upon the test set log-likelihood without improving the RMSE, which demonstrate that the uncertainty is a significant improvement over current methods. Another indicator of improved uncertainty estimation is that our method speeds up active learning tasks compared to state-of-the-art. Due to the similarities between active learning, Bayesian optimization, and reinforcement learning, we expect that our approach carries significant value to these fields as well. Furthermore, we have demonstrated that variational autoencoders can be improved through better generative variance estimation. Finally, we note that our approach is directly applicable alongside ensemble methods, which may further improve results.

Acknowledgements. This project has received funding from the European Research Council (ERC) under the European Union's Horizon 2020 research and innovation programme (grant agreement n° 757360). NSD, MJ and SH were supported in part by a research grant (15334) from VILLUM FONDEN. We gratefully acknowledge the support of NVIDIA Corporation with the donation of GPU hardware used for this research.

References

D. Amodei, C. Olah, J. Steinhardt, P. Christiano, J. Schulman, and D. Mané. Concrete problems in ai safety. *arXiv preprint arXiv:1606.06565*, 2016.

G. Arvanitidis, L. K. Hansen, and S. Hauberg. Latent space oddity: on the curvature of deep generative models. In *International Conference on Learning Representations*, 2018.

C. M. Bishop. Mixture density networks. Technical report, Citeseer, 1994.

A. Damianou and N. D. Lawrence. Deep gaussian processes. *Proceedings of the 16th International Conference on Artificial Intelligence and Statistics (AISTATS)*, 2013.

P. I. Frazier. A tutorial on bayesian optimization. *arXiv preprint arXiv:1807.02811*, 2018.

C. Fu and D. Cai. Efanna : An extremely fast approximate nearest neighbor search algorithm based on knn graph. 09 2016.

Y. Gal and Z. Ghahramani. Dropout as a bayesian approximation: Representing model uncertainty in deep learning. In *international Conference on Machine Learning*, pages 1050–1059, 2016.

A. Gelman, J. B. Carlin, H. S. Stern, D. B. Dunson, A. Vehtari, and D. B. Rubin. *Bayesian Data Analysis*. CRC Press, 2014.

C. Guo, G. Pleiss, Y. Sun, and K. Q. Weinberger. On calibration of modern neural networks. In *Proceedings of the 34th International Conference on Machine Learning-Volume 70*, pages 1321–1330. JMLR. org, 2017.

S. Hauberg. Only bayes should learn a manifold (on the estimation of differential geometric structure from data). *arXiv preprint arXiv:1806.04994*, 2018.

J. M. Hernández-Lobato and R. Adams. Probabilistic backpropagation for scalable learning of bayesian neural networks. In *International Conference on Machine Learning*, pages 1861–1869, 2015.

G. Hinton, N. Srivastava, A. Krizhevsky, I. Sutskever, and R. R. Salakhutdinov. Improving neural networks by preventing co-adaptation of feature detectors. *arXiv preprint*, arXiv, 07 2012.

M. D. Hoffman, D. M. Blei, C. Wang, and J. Paisley. Stochastic variational inference. *The Journal of Machine Learning Research*, 14(1):1303–1347, 2013.

D. G. Horvitz and D. J. Thompson. A generalization of sampling without replacement from a finite universe. *Journal of the American Statistical Association*, 47(260):663–685, 1952.

N. Houlsby, F. Huszar, Z. Ghahramani, and J. M. Hernández-lobato. Collaborative gaussian processes for preference learning. In F. Pereira, C. J. C. Burges, L. Bottou, and K. Q. Weinberger, editors, *Advances in Neural Information Processing Systems 25*, pages 2096–2104. Curran Associates, Inc., 2012.

S.-J. Huang, R. Jin, and Z.-H. Zhou. Active learning by querying informative and representative examples. In *Advances in neural information processing systems*, pages 892–900, 2010.

A. Kendall and Y. Gal. What uncertainties do we need in bayesian deep learning for computer vision? In *Advances in neural information processing systems*, pages 5574–5584, 2017.

D. P. Kingma and M. Welling. Auto-encoding variational bayes. *ICLR*, 12 2013.

A. D. Kiureghian and O. Ditlevsen. Aleatory or epistemic? does it matter? *Structural Safety*, 31(2):105 – 112, 2009. Risk Acceptance and Risk Communication.

B. Lakshminarayanan, A. Pritzel, and C. Blundell. Simple and scalable predictive uncertainty estimation using deep ensembles. In *Advances in Neural Information Processing Systems*, pages 6402–6413, 2017.

K. L. Lange, R. J. A. Little, and J. M. G. Taylor. Robust statistical modeling using the t distribution. *Journal of the American Statistical Association*, 84(408):881–896, 1989.

Y. LeCun, Y. Bengio, and G. E. Hinton. Deep learning. *Nature*, 521(7553):436–444, 2015.

C. Loader. *Local Regression and Likelihood*. Springer, New York, 1999.

D. J. C. MacKay. A Practical Bayesian Framework for Backpropagation Networks. *Neural Comput.*, 4(3): 448–472, may 1992.

W. Maddox, T. Garipov, P. Izmailov, D. Vetrov, and A. G. Wilson. A Simple Baseline for Bayesian Uncertainty in Deep Learning. *CoRR*, feb 2019.

P.-A. Mattei and J. Frellsen. Leveraging the exact likelihood of deep latent variable models. In *Proceedings of the 32Nd International Conference on Neural Information Processing Systems*, NIPS'18, pages 3859–3870, USA, 2018. Curran Associates Inc.

D. Nix and A. Weigend. Estimating the mean and variance of the target probability distribution. In *Proc. 1994 IEEE Int. Conf. Neural Networks*, pages 55–60 vol.1. IEEE, 1994.

T. Pearce, M. Zaki, A. Brintrup, and A. Neely. High-Quality Prediction Intervals for Deep Learning: A Distribution-Free, Ensembled Approach. In *Proceedings of the 35th International Conference on Machine Learning*, feb 2018.

C. E. Rasmussen and C. Williams. *Gaussian Processes for Machine Learning*. University Press Group Limited, 2006.

D. J. Rezende, S. Mohamed, and D. Wierstra. Stochastic backpropagation and approximate inference in deep generative models. In E. P. Xing and T. Jebara, editors, *Proceedings of the 31st International Conference on Machine Learning*, volume 32 of *Proceedings of Machine Learning Research*, pages 1278–1286, Bejing, China, 22–24 Jun 2014. PMLR.

B. Settles. Active learning literature survey. Technical report, University of Wisconsin-Madison Department of Computer Sciences, 2009.

E. Snelson and Z. Ghahramani. Sparse gaussian processes using pseudo-inputs. In *Advances in neural information processing systems*, pages 1257–1264, 2006.

C. Szepesvári. Algorithms for reinforcement learning. *Synthesis lectures on artificial intelligence and machine learning*, 4(1):1–103, 2010.

H. Takahashi, T. Iwata, Y. Yamanaka, M. Yamada, and S. Yagi. Student-t variational autoencoder for robust density estimation. In *Proceedings of the Twenty-Seventh International Joint Conference on Artificial Intelligence, IJCAI-18*, pages 2696–2702. International Joint Conferences on Artificial Intelligence Organization, 7 2018.

R. Tibshirani and T. Hastie. Local likelihood estimation. *Journal of the American Statistical Association*, 82 (398):559–567, 1987.

Paper B

Stochastic Differential Equations with Variational Wishart Diffusions

Stochastic Differential Equations with Variational Wishart Diffusions

Martin Jørgensen [1] **Marc Peter Deisenroth** [2] **Hugh Salimbeni** [3]

Abstract

We present a Bayesian non-parametric way of inferring stochastic differential equations for both regression tasks and continuous-time dynamical modelling. The work has high emphasis on the *stochastic* part of the differential equation, also known as the diffusion, and modelling it by means of Wishart processes. Further, we present a semi-parametric approach that allows the framework to scale to high dimensions. This successfully lead us onto how to model both latent and auto-regressive temporal systems with conditional heteroskedastic noise. We provide experimental evidence that modelling diffusion often improves performance and that this randomness in the differential equation can be essential to avoid over-fitting.

1. Introduction

An endeared assumption to make when modelling multivariate phenomena with Gaussian processes (GPs) is that of independence between processes, i.e. every dimension of a multivariate phenomenon is modelled independently. Consider the case of a two-dimensional temporal process $\boldsymbol{x}_t$ evolving as

$$\boldsymbol{x}_t := f(\boldsymbol{x}_{t-1}) + \boldsymbol{\epsilon}_t, \tag{1}$$

where $f(\boldsymbol{x}_{t-1}) = (f_1(\boldsymbol{x}_{t-1}), f_2(\boldsymbol{x}_{t-1}))^\top$, f_1 and f_2 independent, and $\boldsymbol{\epsilon}_t \sim \mathcal{N}(\mathbf{0}, \sigma^2 \boldsymbol{I})$. This model is commonly used in the machine learning community and is easy to use and understand, but for many real-world cases the noise is too simplistic. In this paper, we will investigate the noise term $\boldsymbol{\epsilon}_t$ and also make it dependent on the state $\boldsymbol{x}_{t-1}$. This is also known as heteroskedastic noise. We will refer to the sequence of $\boldsymbol{\epsilon}_t$ as the *diffusion* or *process noise*.

[1]Department for Mathematics and Computer Science, Technical University of Denmark [2]Department of Computer Science, University College London [3]G-Research. Correspondence to: Martin Jørgensen <marjor@dtu.dk>.

Proceedings of the 37^{th} International Conference on Machine Learning, Online, PMLR 119, 2020.

Why model the process noise? Assume that in the example above, the two states represent meteorological measurements: rainfall and wind speed. Both are influenced by confounders, such as atmospheric pressure, which are not measured directly. This effect can in the case of the model in (1) only be modelled through the diffusion $\boldsymbol{\epsilon}$. Moreover, wind and rain may not correlate identically for all states of the confounders.

Dynamical modelling with focus in the noise-term is not a new area of research. The most prominent one is the *Auto-Regressive Conditional Heteroskedasticity* (ARCH) model (Engle, 1982), which is central to scientific fields like econometrics, climate science and meteorology. The approach in these models is to estimate large process noise when the system is exposed to a shock, i.e. an unforeseen significant change in states. Thus, it does not depend on the value of some state, but rather on a linear combination of previous states.

In this paper, we address this shortcoming and introduce a model to handle the process noise by the use of *Wishart processes*. Through this, we can sample covariance matrices dependent on the input state. This allows the system to evolve as a homogeneous system rather than independent sequences. By doing so, we can avoid propagating too much noise—which can often be the case with diagonal covariances—and potentially improve on modelling longer-range dependencies. Volatility modelling with GPs has been considered by Wu et al. (2014); Wilson & Ghahramani (2010); Heaukulani & van der Wilk (2019).

For regression tasks, our model is closely related to several recent works exploring *continuous-time* deep neural networks (E, 2017; Haber & Ruthotto, 2017; Chen et al., 2018). Here the notion of depth is no longer a discrete quantity (i.e. the number of hidden layers), but an interval on which a continuous flow is defined. In this view, continuous-time learning takes residual networks (He et al., 2016) to their infinite limit, while remaining computationally feasible. The flow, parameterized by a differential equation, allows for time-series modelling, even with temporal observations that are not equidistant.

This line of work has been extended with stochastic equivalents (Twomey et al., 2019; Tzen & Raginsky, 2019; Liu et al., 2019; Li et al., 2020), and the work by

Andreas & Kandemir (2019), who model the drift and diffusion of an SDE with Bayesian neural networks. These approaches make the framework more robust, as the original approach can fail even on simple tasks (Dupont et al., 2019).

The work that inspired our model most was by Hegde et al. (2019). They model the random field that defines the SDE with a Gaussian field. They consider regression and classification problems. To this end, they can take deep GPs (Damianou & Lawrence, 2013; Salimbeni & Deisenroth, 2017) to their 'infinite limit' while avoiding their degeneracy discussed by Duvenaud et al. (2014).

Our main focus throughout this paper lies on the *stochasticity* of the flow, what impact it has and to which degree it can be tamed or manipulated to improve overall performance. Contributions:

- A model that unifies theory from conditional heteroskedastic dynamics, stochastic differential equations (SDEs) and regression. We show how to perform variational inference in this model.
- A scalable approach to extend the methods to high-dimensional input without compromising with inter-dimensional independence assumptions.

2. Background

In this section, we give an overview of the relevant material on GPs, Wishart processes, and SDEs.

2.1. Gaussian Processes

A *Gaussian process* (GP) is a distribution over functions $f : \mathbb{R}^d \to \mathbb{R}^D$, satisfying that for any finite set of points $\boldsymbol{X} := \big(\boldsymbol{x}_1, \dots, \boldsymbol{x}_N\big)^\top \in \mathbb{R}^{N\times d}$, the outputs $\big(f(\boldsymbol{x}_1), \dots, f(\boldsymbol{x}_N)\big)^\top \in \mathbb{R}^{N\times D}$ are jointly Gaussian distributed. A GP is fully determined by a mean function $\mu : \mathbb{R}^d \to \mathbb{R}^D$ and a covariance function $c : \mathbb{R}^d \times \mathbb{R}^d \to \mathbb{R}^{D\times D}$. This notation is slightly unorthodox, and we will elaborate.

The usual convention when dealing with *multi-output* GPs (i.e. $D > 1$) is to assume D i.i.d. processes that share the same covariance function (Álvarez & Lawrence, 2011), which equivalently can be done by choosing the covariance matrix $\boldsymbol{K} = k(\boldsymbol{X}, \boldsymbol{X}) \otimes \boldsymbol{I}_D$, where $\otimes$ denotes the Kronecker product and k is a covariance function for univariate output. For ease of notation we shall use $k^D(\boldsymbol{a}, \boldsymbol{b}) := k(\boldsymbol{a}, \boldsymbol{b}) \otimes \boldsymbol{I}_D$; that is, $k(\boldsymbol{a}, \boldsymbol{b})$ returns a kernel matrix of dimension number of rows in $\boldsymbol{a}$ times the number of rows in $\boldsymbol{b}$. This corresponds to the assumption of independence between output dimensions. Furthermore, we write $\boldsymbol{f} := f(\boldsymbol{X})$, $\boldsymbol{\mu} := \mathrm{vec}(\mu(\boldsymbol{X}))$ and denote by $\boldsymbol{K}$ the $ND \times ND$-matrix with $K_{i,j} = k^D(\boldsymbol{x}_i, \boldsymbol{x}_j)$. Then we can write in short $p(\boldsymbol{f}) = \mathcal{N}(\boldsymbol{\mu}, \boldsymbol{K})$.

As the number N of training data points gets large, the size of $\boldsymbol{K}$ becomes a challenge as well, due to a required inversion during training/prediction. To circumvent this, we consider *sparse* (or low-rank) GP methods. In this respect, we choose M auxiliary *inducing* locations $\boldsymbol{Z} = \big(\boldsymbol{z}_1, \dots, \boldsymbol{z}_M\big)^\top \in \mathbb{R}^{M\times d}$, and define their function values $\boldsymbol{u} := f(\boldsymbol{Z}) \in \mathbb{R}^{M\times D}$. Since any finite set of function values are jointly Gaussian, $p(\boldsymbol{f}, \boldsymbol{u})$ is Gaussian as well, and we can write $p(\boldsymbol{f}, \boldsymbol{u}) = p(\boldsymbol{f}|\boldsymbol{u})p(\boldsymbol{u})$, where $p(\boldsymbol{f}|\boldsymbol{u}) = \mathcal{N}(\tilde{\boldsymbol{\mu}}, \tilde{\boldsymbol{K}})$ with

$$\tilde{\boldsymbol{\mu}} = \boldsymbol{\mu} + \boldsymbol{\alpha}^\top \mathrm{vec}(\boldsymbol{u} - \mu(\boldsymbol{Z})), \tag{2}$$

$$\tilde{\boldsymbol{K}} = k^D(\boldsymbol{X}, \boldsymbol{X}) - \boldsymbol{\alpha}^\top k^D(\boldsymbol{Z}, \boldsymbol{Z})\boldsymbol{\alpha}, \tag{3}$$

where $\boldsymbol{\alpha} = k^D(\boldsymbol{X}, \boldsymbol{Z})k^D(\boldsymbol{Z}, \boldsymbol{Z})^{-1}$. Here it becomes evident why this is computationally attractive, as we only have to deal with the inversion of $k^D(\boldsymbol{Z}, \boldsymbol{Z})$, which due to the structure, only requires inversion of $k(\boldsymbol{Z}, \boldsymbol{Z})$ of size $M\times M$. This is opposed to a matrix of size $ND \times ND$ had we not used the low-rank approximation and independence of GPs.

We will consider variational inference to marginalise $\boldsymbol{u}$ (Titsias, 2009). Throughout the paper, we will choose our variational posterior to be $q(\boldsymbol{f}, \boldsymbol{u}) = p(\boldsymbol{f}|\boldsymbol{u})q(\boldsymbol{u})$, where $q(\boldsymbol{u}) := \mathcal{N}(\boldsymbol{m}, \boldsymbol{S})$, similar to Hensman et al. (2013). Further, q factorises over the dimensions, i.e. $q(\boldsymbol{u}) = \prod_{j=1}^D \mathcal{N}(\boldsymbol{m}_j, \boldsymbol{S}_j)$, where $\boldsymbol{m} = (\boldsymbol{m}_1, \dots, \boldsymbol{m}_D)$ and $\boldsymbol{S}$ is a block-diagonal $MD\times MD$-matrix, with block-diagonal entries $\{\boldsymbol{S}_j\}_{j=1}^D$. In this case, we can analytically marginalise $\boldsymbol{u}$ in (2) to obtain

$$q(\boldsymbol{f}) = \int p(\boldsymbol{f}|\boldsymbol{u})q(\boldsymbol{u})d\boldsymbol{u} = \mathcal{N}(\boldsymbol{\mu}_f^q, \boldsymbol{K}_f^q), \tag{4}$$

$$\boldsymbol{\mu}_f^q = \boldsymbol{\mu} + \boldsymbol{\alpha}^\top \mathrm{vec}(\boldsymbol{m} - \mu(\boldsymbol{Z})), \tag{5}$$

$$\boldsymbol{K}_f^q = k^D(\boldsymbol{X}, \boldsymbol{X}) - \boldsymbol{\alpha}^\top \big(k^D(\boldsymbol{Z}, \boldsymbol{Z}) - \boldsymbol{S}\big)\boldsymbol{\alpha}, \tag{6}$$

which resembles (2)–(3), but which is analytically tractable given variational parameters $\big\{\boldsymbol{m}, \boldsymbol{S}, \boldsymbol{Z}\big\}$.

Recall that a *vector field* is a mapping $f : \mathbb{R}^d \to \mathbb{R}^D$ that associates a point in $\mathbb{R}^d$ with a vector in $\mathbb{R}^D$. A *Gaussian (random) field* is a vector field, such that for any finite collection of points $\{\boldsymbol{x}_i\}_{i=1}^N$, their associated vectors in $\mathbb{R}^D$ are jointly Gaussian distributed, i.e. a Gaussian field *is* a GP. We shall use both terminologies, but when we refer to a Gaussian field, we will think of the outputs as having a *direction*.

2.2. Wishart Processes

The *Wishart distribution* is a distribution over symmetric, positive semi-definite matrices. It is the multidimensional generalisation of the χ^2-distribution. Suppose $\boldsymbol{F}_v$ is a D-variate Gaussian vector for each $v = 1, \dots, \nu$ independently, say $\boldsymbol{F}_v \sim \mathcal{N}(\boldsymbol{0}, \boldsymbol{A})$. Then $\boldsymbol{\Sigma} = \sum_{v=1}^{\nu} \boldsymbol{F}_v \boldsymbol{F}_v^\top$ is Wishart

distributed with ν degrees of freedom and scale matrix $\boldsymbol{A}$. We write for short $\boldsymbol{\Sigma} \sim \mathcal{W}_D(\boldsymbol{A}, \nu)$. By Bartlett's decomposition (Kshirsagar, 1959), this can also be represented as $\boldsymbol{\Sigma} = \boldsymbol{L}\boldsymbol{F}\boldsymbol{F}^\top\boldsymbol{L}^\top$, where $\boldsymbol{F}$ is a $D \times \nu$-matrix with all entries unit Gaussian and $\boldsymbol{A} = \boldsymbol{L}\boldsymbol{L}^\top$.

With this parametrization we define Wishart processes, as in (Wilson & Ghahramani, 2010):

Definition 1. *Let $\boldsymbol{L}$ be a $D \times D$ matrix, such that $\boldsymbol{L}\boldsymbol{L}^\top$ is positive semidefinite and $f_{d,v} \sim \mathcal{GP}\big(0, k_{d,v}(\boldsymbol{x}, \boldsymbol{x}')\big)$ independently for every $d = 1, \dots, D$ and $v = 1 \dots, \nu$, where $\nu \geq D$. Then if*

$$\Sigma(\boldsymbol{x}) = \boldsymbol{L}\left(\sum_{v=1}^{\nu} \boldsymbol{f}_v(\boldsymbol{x})\boldsymbol{f}_v^\top(\boldsymbol{x})\right)\boldsymbol{L}^\top \tag{7}$$

is Wishart distributed for any marginal $\boldsymbol{x}$, and if for any finite collection of points $\boldsymbol{X} = \{\boldsymbol{x}_i\}_{i=1}^N$ the joint distribution $\Sigma(\boldsymbol{X})$ is determined through the covariance functions $k_{d,v}$, then $\Sigma(\cdot)$ is a Wishart process. We will write

$$\Sigma \sim \mathcal{WP}_D(\boldsymbol{L}\boldsymbol{L}^\top, \nu, \kappa), \tag{8}$$

where κ is the collection of covariance functions $\{k_{d,v}\}$.

If $\boldsymbol{\Sigma}$ follows a Wishart distribution with ν degrees of freedom and scale matrix $\boldsymbol{L}\boldsymbol{L}^\top$ of size $D \times D$, then for some $\rho \times D$-matrix $\boldsymbol{R}$ of rank ρ, we have that $\boldsymbol{R}\boldsymbol{\Sigma}\boldsymbol{R}^\top \sim \mathcal{W}_\rho(\boldsymbol{R}\boldsymbol{L}\boldsymbol{L}^\top\boldsymbol{R}^\top, \nu)$. That is, $\boldsymbol{R}\boldsymbol{\Sigma}\boldsymbol{R}^\top$ is Wishart distributed on the space of $\rho \times \rho$ symmetric, positive semi-definite matrices.

The Wishart distribution is closely related to the Gaussian distribution in a Bayesian framework, as it is the conjugate prior to the precision matrix of a multivariate Gaussian. Furthermore, it is the distribution of the maximum likelihood estimator of the covariance matrix.

The Wishart process is a slight misnomer as the posterior processes are *not* marginally Wishart. This is due to the mean function not being constant 0, and a more accurate name could be *matrix-Gamma* processes. We shall not refrain from the usual terminology: a Wishart process is a stochastic process, whose *prior* is a Wishart process.

2.3. Stochastic Differential Equations

We will consider SDEs of the form

$$d\boldsymbol{x}_t = \mu(\boldsymbol{x}_t)dt + \sqrt{\boldsymbol{\Sigma}(\boldsymbol{x}_t)}dB_t, \tag{9}$$

where the last term of the right-hand side is the Itô integral (Itô, 1946). The solution $\boldsymbol{x}_t$ is a stochastic process, often referred to as a *diffusion process*, and μ and $\boldsymbol{\Sigma}$ are the drift and diffusion coefficients, respectively. In (9), B_t denotes the Brownian motion.

The Brownian motion is the GP satisfying that all increments are independent in the sense that, for $0 \leq s_1 < t_1 \leq s_2 < t_2$, then $B_{t_1 - s_1}$ is independent from $B_{t_2 - s_2}$. Further, any increment has distribution $B_t - B_s \sim \mathcal{N}(0, t - s)$. Lastly, $B_0 = 0$. This is equivalent to the GP with constant mean function 0 and covariance function $(t, s) \mapsto \min\{s, t\}$ (Rasmussen & Williams, 2006).

Given some initial condition (e.g. $\boldsymbol{x}_0 = \boldsymbol{0}$), we can generate *sample paths* $[0, T] \to \mathbb{R}^D$ by the Euler-Maruyama method. Euler-Maruyama (Kloeden & Platen, 2013) finely discretizes the temporal dimension $0 = t_0 < t_1 < \ldots < t_l = T$, and *pushes* $\boldsymbol{x}_{t_i}$ along the vector field $\boldsymbol{x}_{t_{i+1}} = \boldsymbol{x}_{t_i} + \boldsymbol{\mu}(\boldsymbol{x}_{t_i})\Delta_i + \sqrt{\boldsymbol{\Sigma}(\boldsymbol{x}_{t_i})\Delta_i}\boldsymbol{N}$, where $\boldsymbol{N} \sim \mathcal{N}(\boldsymbol{0}, \boldsymbol{I}_D)$ and $\Delta_i = t_{i+1} - t_i$.

3. Model and variational inference

We consider a random field $f : \mathbb{R}^D \times [0, T] \to \mathbb{R}^D$ and a GP $g : \mathbb{R}^D \to \mathbb{R}^\eta$. Their priors are

$$f \sim \mathcal{GP}(0, k_f(\cdot, \cdot) \otimes \boldsymbol{I}_D), \quad g \sim \mathcal{GP}(0, k_g(\cdot, \cdot) \otimes \boldsymbol{I}_\eta). \tag{10}$$

We also have a Wishart process $\Sigma : \mathbb{R}^D \times [0, T] \to \mathcal{G}$, where $\mathcal{G}$ is the set of symmetric, positive semi-definite $D \times D$ matrices; the specific prior on this will follow in Section 3.1. We will approximate the posteriors of f, g and Σ with variational inference, but first we will formalise the model.

We propose a continuous-time deep learning model that can propagate noise in high-dimensions. This is done by letting the diffusion coefficient $\boldsymbol{\Sigma}(\boldsymbol{x}_t)$ of an SDE be governed by a Wishart process. The model we present factorises as

$$\begin{aligned} p(\boldsymbol{y}, \Theta) =& p(\boldsymbol{y}|\boldsymbol{g})p(\boldsymbol{g}|\boldsymbol{x}_T, \boldsymbol{u}_g)p(\boldsymbol{u}_g)p(\boldsymbol{x}_T|\boldsymbol{f}) \\ & \cdot p(\boldsymbol{f}|\boldsymbol{\Sigma}, \boldsymbol{u}_f)p(\boldsymbol{u}_f)p(\boldsymbol{\Sigma}|\boldsymbol{u}_\Sigma)p(\boldsymbol{u}_\Sigma), \end{aligned} \tag{11}$$

where $\Theta := \big\{\boldsymbol{g}, \boldsymbol{u}_g, \boldsymbol{x}_T, \boldsymbol{f}, \boldsymbol{u}_f, \boldsymbol{\Sigma}, \boldsymbol{u}_\Sigma\big\}$ denotes all variables to be marginalised. We assume that data $\mathcal{D} = \big\{(\boldsymbol{x}_i, \boldsymbol{y}_i)\big\}_{i=1}^N$ is i.i.d. given the process, such that $p(\boldsymbol{y}|\boldsymbol{g}) = \prod_{i=1}^N p(\boldsymbol{y}_i|\boldsymbol{g}_i)$. We approximate the posterior of g with the variational distribution as in (4), i.e.

$$q(\boldsymbol{g}_i) = \int p(\boldsymbol{g}_i|\boldsymbol{u}_g)q(\boldsymbol{u}_g)d\boldsymbol{u}_g \tag{12}$$

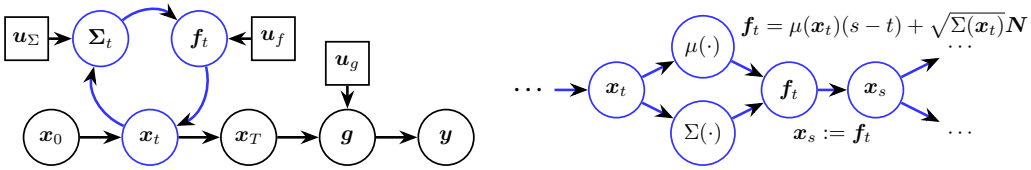
$$= \mathcal{N}(\tilde{\mu}_g(\boldsymbol{x}_i), \tilde{k}_g(\boldsymbol{x}_i, \boldsymbol{x}_i)), \tag{13}$$

where

$$\tilde{\mu}_g(\boldsymbol{x}_i) = \boldsymbol{\alpha}_g^\top(\boldsymbol{x}_i)\text{vec}(\boldsymbol{m}_g), \tag{14}$$

$$\begin{aligned} \tilde{k}_g(\boldsymbol{x}_i, \boldsymbol{x}_i) =& k_g^\eta(\boldsymbol{x}_i, \boldsymbol{x}_i) \\ & - \boldsymbol{\alpha}_g^\top(\boldsymbol{x}_i)\big(k_g^\eta(\boldsymbol{Z}_g, \boldsymbol{Z}_g) - \boldsymbol{S}_g\big)\boldsymbol{\alpha}_g(\boldsymbol{x}_i), \end{aligned} \tag{15}$$

where $\boldsymbol{\alpha}_g(\boldsymbol{x}_i) := k_g^\eta(\boldsymbol{x}_i, \boldsymbol{Z}_g)k_g^\eta(\boldsymbol{Z}_g, \boldsymbol{Z}_g)^{-1}$. Here $\boldsymbol{m}_g$ is an $M \times \eta$ matrix, and $\boldsymbol{S}_g$ is an $M\eta \times M\eta$-matrix, constructed as η different $M \times M$-matrices $\boldsymbol{S}_g = \{\boldsymbol{S}_j\}_j^\eta$. During inference (Quiñonero Candela & Rasmussen, 2005), we



(a) Graphical model based on Eq. (11)

(b) Cycle from (a) and how it moves along the time-axis.

Figure 1. (a) Graphical model based on the factorisation in Eq. (11); (b) The cycle from (a), which represents the *field* f, and how it moves along the time-axis. Here $\boldsymbol{N} \sim \mathcal{N}(\boldsymbol{0}, (s-t)\boldsymbol{I})$. Blue represents the flow/SDE, square nodes are variational variables.

additionally assume that the marginals $g_i = g(\boldsymbol{x}_i)$ are independent when conditioned on $\boldsymbol{u}_g$. This is an approximation to make inference computationally easier.

The inputs to g are given as the state distribution of an SDE at a fixed time point $T \geq 0$. We construct this SDE from the viewpoint of a random field. Consider the random walk with step size Δ on the simplest Gaussian field, where any state has mean $\boldsymbol{\mu}$ and covariance $\boldsymbol{\Sigma}$. For any time point t, the state distribution is tractable, i.e. $p(\boldsymbol{x}_t) = \boldsymbol{x}_0 + \sum_{s=1}^{S} \mathcal{N}(\Delta_s \boldsymbol{\mu}, \Delta_s \boldsymbol{\Sigma})$, where $\sum \Delta_s = t$ and S is any positive integer.

For a state-dependent Gaussian field, we define the random walk

$$\boldsymbol{x}_{t+\Delta} = \boldsymbol{x}_t + \mu(\boldsymbol{x}_t)\Delta + \sqrt{\boldsymbol{\Sigma}(\boldsymbol{x}_t)\Delta}\boldsymbol{N}, \tag{16}$$

with $\boldsymbol{N} \sim \mathcal{N}(\boldsymbol{0}, \boldsymbol{I})$. Given an initial condition $\boldsymbol{x}_0$, the state $\boldsymbol{x}_S$ after S steps is given by

$$\boldsymbol{x}_S = \boldsymbol{x}_0 + \sum_{s=0}^{S-1} \Big(\mu(\boldsymbol{x}_s)\Delta + \sqrt{\boldsymbol{\Sigma}(\boldsymbol{x}_s)\Delta}\boldsymbol{N}\Big). \tag{17}$$

In the limit $\Delta \to 0$, this random walk dynamical system is given by the diffusion process (Durrett, 2018)

$$\boldsymbol{x}_T - \boldsymbol{x}_0 = \int_0^T \mu(\boldsymbol{x}_t)dt + \int_0^T \sqrt{\boldsymbol{\Sigma}(\boldsymbol{x}_t)}dB_t, \tag{18}$$

where B is a Brownian motion. This is an SDE in the Îto-sense, which we numerically can solve by the Euler-Maruyama method. We will see that by a particular choice of variational distribution that $\boldsymbol{\Sigma}(\boldsymbol{x}_t)$ will be the realisation of a Wishart process. The coefficients in (18) are determined as the mean and covariance of a Gaussian field f. The posterior of f is approximated with a Gaussian $q(\boldsymbol{f}_i) = \mathcal{N}(\mu_f^q(\boldsymbol{x}_i), k_f^q(\boldsymbol{x}_i, \boldsymbol{x}_i))$, where

$$\mu_f^q(\boldsymbol{x}_i) = \boldsymbol{\alpha}_f^\top(\boldsymbol{x}_i)\text{vec}(\boldsymbol{m}_f), \tag{19}$$

$$k_f^q(\boldsymbol{x}_i, \boldsymbol{x}_i) = k_f^D(\boldsymbol{x}_i, \boldsymbol{x}_i) \tag{20}$$

$$-\boldsymbol{\alpha}_f^\top(\boldsymbol{x}_i)\big(k_f^D(\boldsymbol{Z}_f, \boldsymbol{Z}_f) - \boldsymbol{S}_f\big)\boldsymbol{\alpha}_f(\boldsymbol{x}_i), \tag{21}$$

and $\boldsymbol{\alpha}_f(\cdot) = k_f^D(\cdot, \boldsymbol{Z}_f)k_f^D(\boldsymbol{Z}_f, \boldsymbol{Z}_f)^{-1}$.

So far, we have seen how we move a data point $\boldsymbol{x}_0$ through the SDE (18) to $\boldsymbol{x}_T$, and further through the GP g, to make a prediction. However, each coordinate of $\boldsymbol{x}$ moves independently. By introducing the Wishart process, we will see how this assumption is removed.

3.1. Wishart-priored Gaussian random field

We are still considering the Gaussian field f, whose posterior is approximated by the variational distribution $q(\boldsymbol{f})$. To regularise (or learn) the noise propagated through this field into g, while remaining within the Bayesian variational framework, we define a hierarchical model as

$$p(\boldsymbol{f}) = \int p(\boldsymbol{f}|\boldsymbol{u}_f, \boldsymbol{\Sigma})p(\boldsymbol{u}_f)p(\boldsymbol{\Sigma}|\boldsymbol{u}_\Sigma)p(\boldsymbol{u}_\Sigma)d\{\boldsymbol{\Sigma}, \boldsymbol{u}_f, \boldsymbol{u}_\Sigma\}, \tag{22}$$

where Σ is a Wishart process. Specifically, its prior is

$$\boldsymbol{\Sigma} \sim \mathcal{WP}_D(\boldsymbol{L}\boldsymbol{L}^\top, \nu, k_f), \tag{23}$$

that is any marginal $\boldsymbol{\Sigma}(\boldsymbol{x}_t) = \boldsymbol{L}\boldsymbol{J}\boldsymbol{J}^\top\boldsymbol{L}^\top$, where $\boldsymbol{J}$ is the $D \times \nu$-matrix with all independent entries $j_{d,v}(\boldsymbol{x}_t)$ drawn from GP's that share the same prior $j_{d,v}(\cdot) \sim \mathcal{GP}(0, k_f(\cdot,\cdot))$. To approximate the posterior of the Wishart process we choose a variational distribution

$$q(\boldsymbol{J}, \boldsymbol{u}_\Sigma) = q(\boldsymbol{J}|\boldsymbol{u}_\Sigma)q(\boldsymbol{u}_\Sigma) := p(\boldsymbol{J}|\boldsymbol{u}_\Sigma)q(\boldsymbol{u}_\Sigma), \tag{24}$$

where $q(\boldsymbol{u}_\Sigma) = \prod_{d=1}^{D}\prod_{v=1}^{\nu}\mathcal{N}(\boldsymbol{m}_{d,v}^\Sigma, \boldsymbol{S}_{d,v}^\Sigma)$. Here, $\boldsymbol{m}_{d,v}^\Sigma$ is $M \times 1$ and $\boldsymbol{S}_{d,v}^\Sigma$ is $M \times M$ for each pair $\{d, v\}$. Notice the same kernel is used for the Wishart process as is used for the random field f, that is: only one kernel *controls* the vector field f. The posterior of $\boldsymbol{\Sigma}$ is naturally defined through the posterior of J. Given our choice of kernel, this approximate posterior is identical to Eqs. (19)-(21), only changing the variational parameters to $\boldsymbol{m}_\Sigma$ and $\boldsymbol{S}_\Sigma$, and D changes to $D\nu$.

What remains to be defined in (11) is $p(\boldsymbol{f}|\boldsymbol{\Sigma}, \boldsymbol{u}_f)$. Since

$\boldsymbol{\Sigma}(\boldsymbol{x}_t)$ is a $D\times D$-matrix we define

$$p\big(\boldsymbol{f}|\{\boldsymbol{\Sigma}(\boldsymbol{x}_i)\}_{i=1}^N,\boldsymbol{u}_f\big)=\mathcal{N}\big(\tilde{\mu}(\boldsymbol{X}),\tilde{k}_f^{\Sigma}(\boldsymbol{X},\boldsymbol{X})\big), \quad (25)$$

$$\tilde{\mu}(\boldsymbol{x}_i)=\boldsymbol{\alpha}_f^{\top}(\boldsymbol{x}_i)\text{vec}(\boldsymbol{u}_f), \quad (26)$$

$$\tilde{k}_f^{\Sigma}(\boldsymbol{x}_i,\boldsymbol{x}_j)=\big(\boldsymbol{\Sigma}(\boldsymbol{x}_i)-\boldsymbol{h}_{ij}\big)\delta_{ij}, \quad (27)$$

where $\boldsymbol{h}_{ij}=\boldsymbol{\alpha}_f(\boldsymbol{x}_i)^{\top}k_f^D(\boldsymbol{Z}_f,\boldsymbol{Z}_f)\alpha_f(\boldsymbol{x}_j)$ and δ_{ij} is Kronecker's delta. Notice this, conditioned on the Wishart process, constitutes a FITC-type model (Snelson & Ghahramani, 2006).

This goes beyond the assumption of independent output dimensions, and instead makes the model learn the inter-dimensional dependence structure through the Wishart process Σ. This structure shall also be learned in the variational inference setup. The posterior of conditional $\boldsymbol{f}$ is approximated by

$$\begin{aligned}q(\boldsymbol{f},\boldsymbol{u}_f|\{\boldsymbol{\Sigma}(\boldsymbol{x}_i)\}_{i=1}^N)&=q(\boldsymbol{f}|\{\boldsymbol{\Sigma}(\boldsymbol{x}_i)\}_{i=1}^N,\boldsymbol{u}_f)q(\boldsymbol{u}_f)\\&=p(\boldsymbol{f}|\{\boldsymbol{\Sigma}(\boldsymbol{x}_i)\}_{i=1}^N,\boldsymbol{u}_f)q(\boldsymbol{u}_f),\end{aligned} \quad (28)$$

where $q(\boldsymbol{u}_f):=\mathcal{N}(\boldsymbol{m}_f,k_f^D(\boldsymbol{Z}_f,\boldsymbol{Z}_f))$. At first, this might seem restrictive, but covariance estimation is already in Σ and the variational approximation is the simple expression

$$q(\boldsymbol{f}|\{\boldsymbol{\Sigma}(\boldsymbol{x}_i)\}_{i=1}^N)=\prod_{i=1}^N\mathcal{N}\big(\alpha_f^{\top}(\boldsymbol{x}_i)\boldsymbol{m}_f,\boldsymbol{\Sigma}(\boldsymbol{x}_i)\big). \quad (29)$$

The marginalisation can then be computed with Jensen's inequality

$$\begin{aligned}\log p(\boldsymbol{y})&=\log\int p(\boldsymbol{y},\Theta)d\Theta\\&\geq\int\log\Big(\frac{p(\boldsymbol{y},\Theta)}{q(\Theta)}\Big)q(\Theta)d\Theta\\&=\int\log p(\boldsymbol{y}|\boldsymbol{g})q\big(\boldsymbol{g}|\Theta\setminus\{\boldsymbol{g}\}\big)d\Theta\\&\quad-\text{KL}\big(q(\boldsymbol{u}_g)\|p(\boldsymbol{u}_g)\big)\\&\quad-\text{KL}\big(q(\boldsymbol{u}_f)\|p(\boldsymbol{u}_f)\big)-\text{KL}\big(q(\boldsymbol{u}_\Sigma)\|p(\boldsymbol{u}_\Sigma)\big),\end{aligned} \quad (30)$$

or, in a more straightforward language,

$$\begin{aligned}\log p(\boldsymbol{y})\geq\mathbb{E}_{q(g)}[\log p(\boldsymbol{y}|\boldsymbol{g})]-\text{KL}\big(q(\boldsymbol{u}_g)\|p(\boldsymbol{u}_g)\big)\\-\text{KL}\big(q(\boldsymbol{u}_f)\|p(\boldsymbol{u}_f)\big)-\text{KL}\big(q(\boldsymbol{u}_\Sigma)\|p(\boldsymbol{u}_\Sigma)\big).\end{aligned} \quad (31)$$

The right-hand side in (31) is the so-called *evidence lower bound* (ELBO). The first term, the expectation, is analytically intractable, due to $q(g)$ being non-conjugate to the likelihood. Therefore, we determine it numerically with Monte Carlo (MC) or with Gauss-Hermite quadrature (Hensman et al., 2015). With MC, often a few samples are enough for reliable inference (Salimans & Knowles, 2013).

The KL-terms in (31) can be computed analytically as they all involve multivariate Gaussians. Still, due to some of the modelling constraints, it is helpful to write them out, which yields

$$\text{KL}\big(q(\boldsymbol{u}_g)\|p(\boldsymbol{u}_g)\big)=\sum_{d=1}^{\eta}\text{KL}\big(q(\boldsymbol{u}_{g_d})\|p(\boldsymbol{u}_{g_d})\big), \quad (32)$$

$$\text{KL}\big(q(\boldsymbol{u}_\Sigma)\|p(\boldsymbol{u}_\Sigma)\big)=\sum_{d=1}^{D}\sum_{v=1}^{\nu}\text{KL}\big(q(\boldsymbol{u}_{\Sigma_{d,v}})\|p(\boldsymbol{u}_{\Sigma_{d,v}})\big), \quad (33)$$

where in both instances we used the independence between the GPs. The remaining one is special. Since both distribution share the same covariance it reduces to

$$\text{KL}\big(q(\boldsymbol{u}_f)\|p(\boldsymbol{u}_f)\big)=\frac{1}{2}\sum_{d=1}^{D}\boldsymbol{m}_{f_d}^{\top}k_f^D(\boldsymbol{Z}_f,\boldsymbol{Z}_f)^{-1}\boldsymbol{m}_{f_d}. \quad (34)$$

Here, $k_f^D(\boldsymbol{Z}_f,\boldsymbol{Z}_f)^{-1}$ is already known from the computation of (33), as the kernel and inducing locations are shared.

Summarising this section, we have inputs $\boldsymbol{x}_0:=\boldsymbol{x}$ that are warped through an SDE (governed by a random field f) with drift μ and diffusion Σ that is driven by *one* kernel k_f^D. The value of this SDE, at some given time T, is then used as input to a final layer g, i.e. $g(\boldsymbol{x}_T)$ predicts targets $y(\boldsymbol{x})$. All this is inferred by maximising the ELBO (31).

3.2. Complexity and scalability

The computational cost of estimating $\boldsymbol{\Sigma}$ with a Wishart, as opposed to a diagonal matrix, can be burdensome. For the diagonal, the cost is $\mathcal{O}(DNM^2)$ since we need to compute (3) D times. Sampling $D\nu$ GP values and then matrix-multiplying it with a $D\times\nu$ matrix is of complexity $\mathcal{O}(D\nu NM^2+D\nu D)$. Hence, if we, for simplicity, let $\nu=D$, we have overhead cost of $\mathcal{O}(D^2NM^2+D^3)$. Note this is only the computational budget associated with the diffusion coefficients of the random field; the most costly one.

On this inspection, we propose a way to overcome a too heavy burden if D is large. Naturally this involves an approximation; this time a low-rank approximation on the dimensionality-axis. Recall that, if $\boldsymbol{\Sigma}_\rho\sim\mathcal{WP}_\rho(\boldsymbol{I},\nu,\kappa)$, then $\boldsymbol{\Sigma}_D:=\boldsymbol{L}\boldsymbol{\Sigma}_\rho\boldsymbol{L}^{\top}\sim\mathcal{WP}_D(\boldsymbol{L}\boldsymbol{L}^{\top},\nu,\kappa)$. The matrices naturally are of rank $\rho\ll D$. The computational overhead is reduced to $\mathcal{O}(\rho^2NM^2+D\rho^2)$ if $\nu=\rho$. This same structure was introduced by Heaukulani & van der Wilk (2019) for time-series modelling of financial data; and it reminisces the structure of Semiparametric Latent Factor Models (SLFM) (Seeger et al., 2005). That is, we have ρ GPs, and the D-dimensional outputs are all linear combinations of these. For clarity, we need only to compute/sample $\sqrt{\boldsymbol{\Sigma}_D}=\boldsymbol{L}\boldsymbol{J}$,

where $\boldsymbol{J}$ is a $\rho \times \nu$ matrix, with GP values according to the approximate posterior $q(\boldsymbol{J})$, where D replaced by ρ.

3.3. Further model specifications

If ρ is too small it can be difficult to identify a good diffusion coefficient as the matrix is too restricted by the low rank. One possible way to overcome this is too add 'white noise' to the matrix

$$\boldsymbol{\Sigma} = \boldsymbol{L}\boldsymbol{F}\boldsymbol{F}^\top\boldsymbol{L}^\top + \boldsymbol{\Lambda}, \tag{35}$$

where $\boldsymbol{\Lambda}$ is a diagonal $D \times D$-matrix. In many situations, this will ensure that the diffusion is full rank, and this provides more freedom in estimating the marginal variances. However, if the values on the diagonal of $\boldsymbol{\Lambda}$ are estimated by maximum likelihood, we have to be cautious. If $\boldsymbol{\Lambda}$ becomes to 'dominant', inference can turn off the Wishart-part, potentially leading to overfitting.

Consider the matrix $\boldsymbol{L}$, that makes up the scale matrix of the Wishart process. It is fully inferred by maximum likelihood, hence there is no KL-term to regularise it. Effectively, this can turn off the stochasticity of the flow by making some matrix norm of $\boldsymbol{L}$ be approximately zero. Then the flow is only determined by its drift and overfitting is a likely scenario.

To alleviate this concern we propose to regularise $\boldsymbol{L}$ by its rownorms. That is,

$$\forall d = 1, \dots, D: \quad \sum_{r=1}^{\rho} L_{d,r}^2 = 1, \tag{36}$$

where $L_{d,r}$ denotes the entries of $\boldsymbol{L}$. First of all, this ensures that the prior variance for all dimensions is determined by the kernel hyperparameters, as it makes the diagonal of the scale matrix $\boldsymbol{L}\boldsymbol{L}^\top$ equal to 1. This way the variance in each dimension is a 'fair' linear combination of the ρ GPs that control the Wishart.

3.4. Extending to time series

The specified model can be specified to model temporal data $\mathcal{D} = \{\boldsymbol{y}_i, t_i\}_{i=1}^N$ in a straightforward way. In a few lines, see also Figure 1, we write

$$\boldsymbol{x}_t = \boldsymbol{x}_0 + \int_0^t \mu(\boldsymbol{x}_s)ds + \int_0^t \sqrt{\boldsymbol{\Sigma}(\boldsymbol{x}_s)}dB_s, \tag{37}$$

$$f(\cdot)|\boldsymbol{\Sigma}(\cdot), \mathcal{D} \sim \mathcal{GP}(\mu(\cdot), \boldsymbol{\Sigma}(\cdot)), \tag{38}$$

$$\boldsymbol{\Sigma}(\cdot) \sim \mathcal{WP}(\cdot|\mathcal{D}), \tag{39}$$

$$p(\boldsymbol{y}_t|\boldsymbol{x}_t) = \mathcal{N}(g(\boldsymbol{x}_t), \boldsymbol{A}\boldsymbol{\Sigma}(\boldsymbol{x}_t)\boldsymbol{A}^\top + \boldsymbol{\Lambda}). \tag{40}$$

If g is not the identity mapping, we can define a *latent dynamical* model. Say g is a GP mapping from $\mathbb{R}^D$ to $\mathbb{R}^\eta$. This is similar to GP state space models (GPSSM) where the dynamics, or transitions, are defined $\boldsymbol{x}_t = f(\boldsymbol{x}_{t-1}) + \boldsymbol{\epsilon}_x$ and $\boldsymbol{y}_t = g(\boldsymbol{y}_t) + \boldsymbol{\epsilon}_y$, for GPs f and g and some noise variables $\boldsymbol{\epsilon}_x$ and $\boldsymbol{\epsilon}_y$, usually Gaussian with zero mean and diagonal covariance matrix (Deisenroth et al., 2012; Eleftheriadis et al., 2017).

The latent dynamics defined in (37)–(39) are not restricted to have equi-temporal measurements and model non-diagonal covariance structure both in the latent states $\boldsymbol{x}$ and in the observed states $\boldsymbol{y}$ through the matrix $\boldsymbol{A}$, which is an $\eta \times D$-matrix. Adding the diagonal $\eta \times \eta$-matrix $\boldsymbol{\Lambda}$ is necessary to avoid singularity. Even though $\boldsymbol{\Sigma}(\cdot)$ is a $D \times D$-matrix, we can still lower-rank approximate with a ρ-rank matrix, as described in Section 3.2. The log-likelihood we compute is

$$\begin{aligned}\log p(\boldsymbol{y}_t|\boldsymbol{g}_t, \boldsymbol{\Sigma}(\boldsymbol{x}_t)) = \frac{\eta}{2}\log(2\pi) - \log(\det(\boldsymbol{B})) \\ - \frac{1}{2}(\boldsymbol{y}_t - \boldsymbol{g}_t)^\top \boldsymbol{B}^{-1}(\boldsymbol{y}_t - \boldsymbol{g}_t),\end{aligned} \tag{41}$$

where $\boldsymbol{B} := \boldsymbol{A}\boldsymbol{\Sigma}(\boldsymbol{x}_t)\boldsymbol{A}^\top + \boldsymbol{\Lambda}$. As a consequence of the matrix-determinant lemma and the Woodbury identity, we can evaluate the likelihood cheaply, because of $\boldsymbol{B}$'s structure. The ELBO that we optimise during training is similar to (31), only the likelihood term is different: it is swapped for a variational expectation over (41). We assume independence between all temporal observations, i.e. $p(\mathcal{D}) = \prod_{i=1}^N p(\{\boldsymbol{y}_i, t_i\})$.

4. Experiments

We evaluate the presented model in both regression and a dynamical setup. In both instances, we use baselines that are similar to our model to easier distinguish the influence the diffusion has on the experiments. We evaluate on a well-studied regression benchmark and on a higher-dimensional dynamical dataset.

4.1. Regression

We compare our model, which we will dub *Wishart-priored GP flow* (*diffWGP*), to three baseline models in order to shed light on some properties of the diffWGP.

GP flows Reproducing the model from Hegde et al. (2019) will give indications, if it is possible to increase overall performance by modelling the randomness in the flow. This model has a diagonal matrix $\boldsymbol{\Sigma}$ with entries determined solely by the chosen covariance function. We will refer to this model with *diffGP*.

No noise flows We also evaluate the model, where $\boldsymbol{\Sigma} = \boldsymbol{0}$, i.e. the situation where the flow is *deterministic*. The remaining part of the flow is still as in (19) to make fair

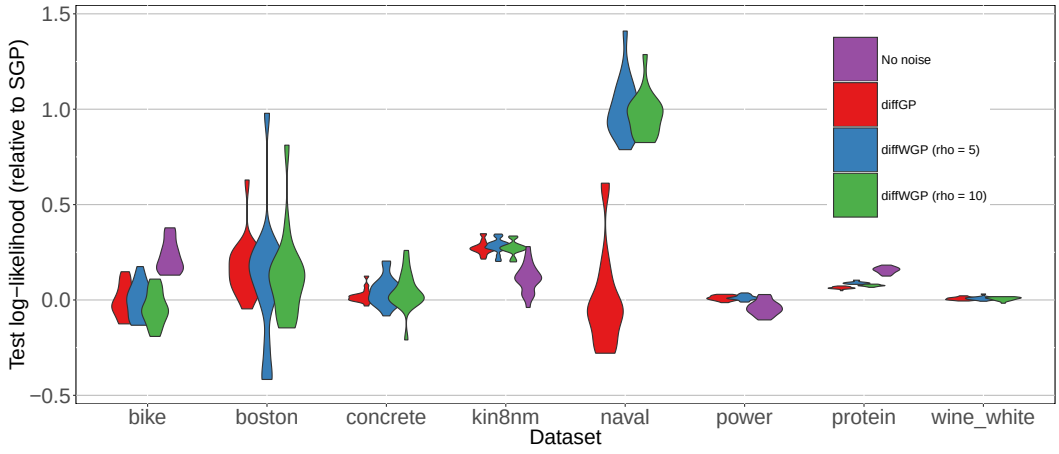


Figure 2. Test-set log-likelihood values on eight UCI regression datasets. The violin plots show the test-set log (likelihood-ratio) of baseline diffusion models with respect to the SGP baseline. Values greater than 0 indicate an improvement over SGP. Key findings are that No noise can overfit heavily (boston, concrete, naval), and diffWGP performs best on most datasets. The figure has been cut for readability—this explain why occasionally purple violins are missing.

comparisons. All the relevant KL-terms are removed from the ELBO (31). We refer to this as *No noise*.

Sparse GPs Also in the variational setup we shall compare to vanilla sparse GPs, i.e. the model introduced by Titsias (2009). We will refer to this as *SGP*.

4.1.1. Experimental Setup

In all experiments, we choose 100 inducing points for the variational distributions, all of which are Gaussians. All models are trained for 50000 iterations with a mini-batch size of 2000, or the number of samples in the data if smaller. In all instances, the first 10000 iterations are *warm-starting* the final layer GP g, keeping all other parameters fixed. We use the Adam-optimiser with a step-size of 0.01. After this all flows (this excludes SGP) are initialised with a constant mean 0 and covariance functions chosen as RBF with automatic relevance determination (ARD), initialised with tiny signal noise to ensure $\boldsymbol{x}_0 \approx \boldsymbol{x}_T$. The time variable T is always 1.

The remaining 40000 iterations (SGP excluded) are updating again with Adam with a more cautious step-size of 0.001. For the diffWGP, the first 4000 of these are warm-starting the KL-terms associated with the flow to speed up convergence. Note that this model fits more parameters than the baseline models. For the diffWGP, we update the ELBO

$$\begin{aligned}&\mathbb{E}_{q(g)}[\log p(y|g)] - \mathrm{KL}\big(q(\boldsymbol{u}_g)\|p(\boldsymbol{u}_g)\big)\\ &\quad - c^2\mathrm{KL}\big(q(\boldsymbol{u}_f)\|p(\boldsymbol{u}_f)\big) - c\mathrm{KL}\big(q(\boldsymbol{u}_\Sigma)\|p(\boldsymbol{u}_\Sigma)\big),\end{aligned} \tag{42}$$

	diffGP vs. SGP	*diffWGP vs. diffGP*
Bike (14)	0.8695	0.2262
Boston (13)	**<0.0001**	0.9867
Concrete (8)	**0.0042**	**0.0348**
kin8nm (8)	**<0.0001**	**0.0164**
Naval (26)	0.8695	**<0.0001**
Power (4)	**<0.0001**	0.1387
Protein (9)	**<0.0001**	**<0.0001**
Wine_white (11)	**0.0003**	0.3238

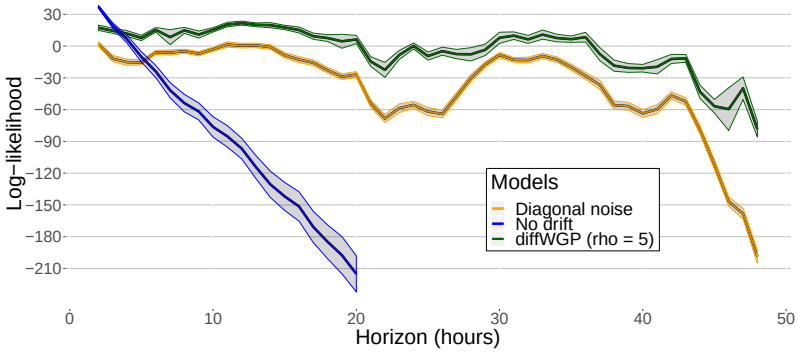
Table 1. Wilcoxons paired signed rank-test. Listed are the p-values of the hypothesis of equal median versus alternative that location shift is negative. Bold highlights the significant ones at a 0.05 confidence level. In parenthesis are the input dimensionality of the datasets. Results are for $\rho = 5$.

where $c = \min(1, \frac{iteration}{4000})$, i.e. we warm-start the regularising KL-terms.

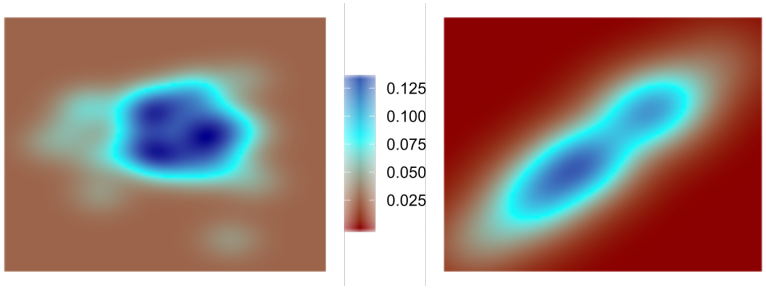
4.1.2. UCI Regression Benchmark

Figure 2 shows the results on eight UCI benchmark datasets over 20 train-test splits (90/10). On the y-axis we see the distribution of the test-set log-likelihood subtracted by the SGP log-likelihood on the same split. Values greater than 0 are improvements over the baseline SGP. An analogous plot with RMSE is supplied in the supplementary material. In Table 1, we use Wilcoxon's paired rank test to evaluate whether the more advanced models perform better.

Key observations are: *not* having noise in the flow (No



(a) The log-likelihood of *forecasted* measurements up to 48 hours. The bold lines mark the average log-likelihood at a given hour based on 50 simulations. The associated shaded areas span twice the standard error.



(b) The density (colour) of the 48-hour horizon predictions of temperature measurement in Tiantan (x-axis) and Dongsi (y-axis). These locations are within a few kilometres of each other. *Left*: diagonal noise case; *Right*: Wishart noise. The Wishart detects a correlation between these two temperature measurements, as we would expect for such nearby locations.

Figure 3. (a): The performance of predictions plotted over the forecast horizon. (b): The joint development of two temperature measurements over the forecasted time-horizon for two different models.

noise) seem to lead to overfitting, except in two cases, where a more expressive model is preferred. In one of these cases (protein) Wishart modelling improves both the RMSE and the log-likelihood. In one case (boston), overfitting was absurdly large: on this dataset we were not able to reproduce the results from Hegde et al. (2019) either. In four cases (concrete, kin8nm, power, wine_white), *No noise* overfitted mildly. In two of these cases, *diffWGP* improved over *diffGP*. The two cases, where no improvement is significant, are simple cases, wine_white and power, which are almost linear or low-dimensional. On the naval dataset, the *No noise* model could not run due to numerical issues. Here *diffWGP* outperforms *diffGP* in the log-likelihood. We conjecture this is because of the high dimensionality and the fact that almost no observation noise is present. We found no substantial influence of the parameter ρ; if any then it actually seems to prefer lower-rank approximations. This emphasises that training Wishart processes is difficult, and further research in this area is needed.

4.2. Auto-regressive modelling of air quality

We evaluate our dynamical model on atmospheric air-quality data from Beijing (Zhang et al., 2017). We pre-processed the data for three locations in the city (Shunyi, Tiantan, Dongsi), which each have hourly observation of ten features over the period of 2014–2016. Explicitly, the ten features are: the concentration of PM2.5, PM10, SO2, NO2, CO, O3, the temperature and dew point temperature, air pressure and amount of precipitation.[1]

[1]Full data set available at `https://archive.ics.uci.edu/ml/datasets/Beijing+Multi-Site+Air-Quality+Data`.

We use the first two years of this dataset for training and aim to forecast into the first 48 hours of 2016. Including the variables year, month, day and hour, we have in total 34 features for the three cities and 17520 temporal observations for training. Missing values were linearly interpolated. All features were standardised.

To analyse properties of our proposed model, we perform an ablation study with the following models:

diffWGP The model proposed in the paper to model the diffusion with Wisharts.

Diagonal noise The drift term remains as in the diffWGP model, but the diffusion is restricted to diagonal, i.e. correlated diffusion cannot be modelled. This becomes the model

$$\boldsymbol{x}_t = \boldsymbol{x}_s + \mu(\boldsymbol{x}_s)(t-s) + \sqrt{\boldsymbol{\Lambda}(t-s)}\boldsymbol{\epsilon}_t, \quad \boldsymbol{\epsilon} \sim \mathcal{N}(\boldsymbol{0}, \boldsymbol{I}). \tag{43}$$

No drift The drift is constantly zero, and the diffusion is modelled by a Wishart, which results in the model

$$\boldsymbol{x}_t = \boldsymbol{x}_s + \sqrt{\big(\boldsymbol{A}\boldsymbol{\Sigma}(\boldsymbol{x}_t)\boldsymbol{A}^\top + \boldsymbol{\Lambda}\big)(t-s)}\boldsymbol{\epsilon}_t. \tag{44}$$

This model is a continuous-time version of the model presented by Heaukulani & van der Wilk (2019).

In all instances, we train by minibatching shorter sequences, and we use the Adam optimiser (Kingma & Ba, 2014) with a learning rate 0.01. Due to the large amount of temporal observation compared to small batches we ease off on momentum.

Figure 3(a) shows how the different models *forecast* future observations by reporting the log-likelihood traces of individual models at test time. The figure shows the mean and two times the standard error, which we obtain from 50 simulations. At first, we see that having no drift starts off better, but quickly drops in performance. This is not unexpected, as the data has structure in its evolution. The difference between the models with drift, but different diffusions, are more interesting for this dataset. Overall, Wishart diffusions perform best, and it seems to be resilient and take only few and relatively small 'dips'.

We expect this dataset to have highly correlated features. The three locations in Beijing are, in distance, close to each other; naturally the different air measurements are similar in their evolution over time. Figure 3(b) illustrates how a model with diagonal noise is incapable of learning this joint development of temperature measurements. Here, the Wishart learns that when the temperature in Dongsi is high, it is also high in Tiantan. This behaviour is seen in many pairs of the features considered, and it suggests *diffWGP* has dynamics moving on a manifold of smaller dimension than if diagonal noise was considered. This supports the hypothesis that *diffWGP* moves as *one* dynamical systems, opposed to 34.

5. Conclusion

In a non-parametric Bayesian way, we presented a scalable approach to continuous-time learning with high emphasis on correlated process noise. This noise is modelled with a Wishart process, which lets high-dimensional data evolve as a single system, rather than D independent systems. We presented a way to scale this to high dimensions. We found that it is never worse taking the dependence structure in the process noise into account. However, with certain types of data, it can mitigate overfitting effects and improve performance.

Code is publicly available at: `https://github.com/JorgensenMart/Wishart-priored-SDE`.

Acknowledgements

MJ was supported by a research grant (15334) from VILLUM FONDEN.

References

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

Andreas, L. and Kandemir, M. Differential Bayesian neural nets. *arXiv:1912.00796*, 2019.

Chen, T. Q., Rubanova, Y., Bettencourt, J., and Duvenaud, D. K. Neural ordinary differential equations. In *Advances in Neural Information Processing Systems*, 2018.

Damianou, A. and Lawrence, N. D. Deep Gaussian processes. In *Artificial Intelligence and Statistics*, 2013.

Deisenroth, M. P., Turner, R., Huber, M., Hanebeck, U. D., and Rasmussen, C. E. Robust filtering and smoothing with Gaussian processes. *IEEE Transactions on Automatic Control*, 57(7):1865–1871, 2012.

Dupont, E., Doucet, A., and Teh, Y. W. Augmented neural ODEs. In *Advances in Neural Information Processing Systems*, 2019.

Durrett, R. *Stochastic calculus: a practical introduction*. CRC press, 2018.

Duvenaud, D., Rippel, O., Adams, R., and Ghahramani, Z. Avoiding pathologies in very deep networks. In *Artificial Intelligence and Statistics*, 2014.

E, W. A proposal on machine learning via dynamical systems. *Communications in Mathematics and Statistics*, 5 (1):1–11, 2017.

Eleftheriadis, S., Nicholson, T. F. W., Deisenroth, M. P., and Hensman, J. Identification of Gaussian process state space models. In *Advances in Neural Information Processing Systems*, 2017.

Engle, R. F. Autoregressive conditional heteroscedasticity with estimates of the variance of United Kingdom inflation. *Econometrica: Journal of the Econometric Society*, pp. 987–1007, 1982.

Haber, E. and Ruthotto, L. Stable architectures for deep neural networks. *Inverse Problems*, 34(1):014004, 2017.

He, K., Zhang, X., Ren, S., and Sun, J. Deep residual learning for image recognition. In *Conference on Computer Vision and Pattern Recognition*, 2016.

Heaukulani, C. and van der Wilk, M. Scalable Bayesian dynamic covariance modeling with variational Wishart and inverse Wishart processes. In *Advances in Neural Information Processing Systems*, 2019.

Hegde, P., Heinonen, M., Lähdesmäki, H., and Kaski, S. Deep learning with differential Gaussian process flows. In *Artificial Intelligence and Statistics*, 2019.

Hensman, J., Fusi, N., and Lawrence, N. D. Gaussian processes for big data. In *Uncertainty in Artificial Intelligence*, 2013.

Hensman, J., Matthews, A. G. d. G., Filippone, M., and Ghahramani, Z. MCMC for variationally sparse Gaussian processes. In *Advances in Neural Information Processing Systems*, 2015.

Itô, K. On a stochastic integral equation. *Proceedings of the Japan Academy*, 22(2):32–35, 1946.

Kingma, D. P. and Ba, J. Adam: A method for stochastic optimization. *arXiv:1412.6980*, 2014.

Kloeden, P. E. and Platen, E. *Numerical Solution of Stochastic Differential Equations*, volume 23. Springer Science & Business Media, 2013.

Kshirsagar, A. M. Bartlett Decomposition and Wishart distribution. *The Annals of Mathematical Statistics*, 30 (1):239–241, 1959.

Li, X., Wong, T.-K. L., Chen, R. T. Q., and Duvenaud, D. Scalable gradients for stochastic differential equations. In *Artificial Intelligence and Statistics*, 2020.

Liu, X., Xiao, T., Si, S., Cao, Q., Kumar, S., and Hsieh, C.-J. Neural SDE: Stabilizing neural ODE networks with stochastic noise. *arXiv:1906.02355*, 2019.

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

Rasmussen, C. E. and Williams, C. *Gaussian Processes for Machine Learning*. MIT Press, 2006.

Salimans, T. and Knowles, D. A. Fixed-form variational posterior approximation through stochastic linear regression. *Bayesian Analysis*, 8(4):837–882, 2013.

Salimbeni, H. and Deisenroth, M. P. Doubly stochastic variational inference for deep Gaussian processes. In *Advances in Neural Information Processing Systems*, 2017.

Seeger, M., Teh, Y.-W., and Jordan, M. Semiparametric latent factor models. Technical report, 2005.

Snelson, E. and Ghahramani, Z. Sparse Gaussian processes using pseudo-inputs. In *Advances in Neural Information Processing Systems*, 2006.

Titsias, M. Variational learning of inducing variables in sparse Gaussian processes. In *Artificial Intelligence and Statistics*, 2009.

Twomey, N., Kozłowski, M., and Santos-Rodríguez, R. Neural ODEs with stochastic vector field mixtures. *arXiv:1905.09905*, 2019.

Tzen, B. and Raginsky, M. Neural stochastic differential equations: deep latent Gaussian models in the diffusion limit. *arXiv:1905.09883*, 2019.

Wilson, A. G. and Ghahramani, Z. Generalised Wishart processes. *arXiv:1101.0240*, 2010.

Wu, Y., Hernández-Lobato, J. M., and Ghahramani, Z. Gaussian process volatility model. In *Advances in Neural Information Processing Systems*. 2014.

Zhang, S., Guo, B., Dong, A., He, J., Xu, Z., and Chen, S. X. Cautionary tales on air-quality improvement in Beijing. *Proceedings of the Royal Society A: Mathematical, Physical and Engineering Sciences*, 473, 2017.

Paper C

Isometric Gaussian Process Latent Variable Model for Dissimilarity Data

Isometric Gaussian Process Latent Variable Model for Dissimilarity Data

Martin Jørgensen
marjor@dtu.dk
Technical University of Denmark

Søren Hauberg
sohau@dtu.dk
Technical University of Denmark

Abstract

We propose a fully generative model where the latent variable respects both the distances and the topology of the modeled data. The model leverages the Riemannian geometry of the generated manifold to endow the latent space with a well-defined stochastic distance measure, which is modeled as Nakagami distributions. These stochastic distances are sought to be as similar as possible to observed distances along a neighborhood graph through a censoring process. The model is inferred by variational inference and is therefore fully generative. We demonstrate how the new model can encode invariances in the learned manifolds.

1 Introduction

Dimensionality reduction aims to compress data to a lower dimensional representation while preserving the underlying signal and suppressing noise. Contemporary nonlinear methods mostly call upon the *manifold assumption* [Bengio et al., 2013] stating that the observed data is distributed near a low-dimensional manifold embedded in the observation space. Beyond this unifying assumption, methods often differ by focusing on one of three key properties (Table 1).

Topology preservation. A *topological space* is a set of points whose *connectivity* is invariant to continuous deformations. For finite data, connectivity is commonly interpreted as a clustering structure, such that topology preserving methods do not form new clusters or break apart existing ones. For visualization purposes, the *uniform manifold approximation projection (UMAP)* [McInnes et al., 2018] appears to be the current state-of-the-art within this domain.

	Generative	Topology	Distance
PCA	(✓)	✗	(✓)
MDS	✗	✗	✓
IsoMap	✗	(✗)	✓
t-SNE	✗	(✓)	✓
UMAP	✗	✓	✓
GPLVM	✓	✗	✗
Iso-GPLVM (our)	✓	✓	✓

Table 1: A list of common dimensionality reduction methods and coarse overview of their features.

Distance preservation. Methods designed to find low-dimensional representation with pairwise distances that are similar to those of the observed data may generally be viewed as a variant of *multi-dimensional scaling (MDS)* [Ripley, 2007]. Usually, this is achieved by a direct minimization of the *stress* defined as

$$\text{stress} = \sum_{i<j\leq N} (d_{ij} - \|\boldsymbol{z}_i - \boldsymbol{z}_j\|)^2, \tag{1}$$

where d_{ij} are the *dissimilarity* (or *distance*) of two data points $\boldsymbol{x}_i$ and $\boldsymbol{x}_j$, and $\mathcal{Z} = \{\boldsymbol{z}_i\}_{i=1}^N$ denote the low-dimensional representation in $\mathbb{R}^q$.

More advanced methods have been built on top of this idea. In particular, *IsoMap* [Tenenbaum et al., 2000] computes d_{ij} along a neighborhood graph using Dijkstra's algorithm. This bears some

Preprint. Under review.

resemblance to *t-SNE* [Maaten and Hinton, 2008] that uses the Kullback-Leibler divergence to match distribution in low-dimensional Euclidean spaces with the data in high dimensions.

Generative models. A common trait for the mentioned methods is that they learn features in a mapping from high-dimensions to low, but not the reverse. This makes the methods mostly useful for visualization. *Generative models* [Kingma and Welling, 2014, Rezende et al., 2014, Lawrence, 2005, Goodfellow et al., 2014, Rezende and Mohamed, 2015] allow us to make new samples in high-dimensional space. Of particular relevance to us, is the *Gaussian process latent variable model (GP-LVM)* [Lawrence, 2005, Titsias and Lawrence, 2010] that learns a stochastic mapping $f : \mathbb{R}^q \to \mathbb{R}^D$ jointly with the latent representations $\boldsymbol{z}$. This is achieved by marginalizing the mapping under a Gaussian process prior [Rasmussen and Williams, 2006]. The generative approach allows the methods to extend beyond visualization to e.g. missing data imputation, data augmentation and semi-supervised tasks [Mattei and Frellsen, 2019, Urtasun and Darrell, 2007].

In this paper we learn a Riemannian manifold using Gaussian processes on which distances on the manifold match the *local* distances as is implied by the Riemannian assumption. Assuming the observed data lies on a Riemannian q-submanifold of $\mathbb{R}^D$ with infinite injectivity radius, then our approach can learn a q-dimensional representation that is isometric to the original manifold. Similar statements only hold true for traditional manifold learning methods that embed into $\mathbb{R}^q$ if the original manifold is flat. We learn global and local structure through a common technique from survival analysis, combined with a likelihood model based on the theory of Gaussian process arc-lengths. Lastly, we show how the GP approach allow us to marginalize the latent representation and produce a fully Bayesian non-parametric generative model. We envision how learning generative models by pairwise dissimilarities easily allow for encoding invariances.

2 Background material

2.1 Gaussian Processes

A Gaussian process (GP) [Rasmussen and Williams, 2006] is a distribution over functions, $f : \mathbb{R}^q \to \mathbb{R}$, which satisfy that for any finite set of points $\{\boldsymbol{z}_i\}_{i=1}^N$, in the domain $\mathbb{R}^q$, the output $\boldsymbol{f} = \big(f(\boldsymbol{z}_1), \dots, f(\boldsymbol{z}_N)\big)$ have a joint Gaussian distribution. This Gaussian is fully determined by a mean function $\mu : \mathbb{R}^q \to \mathbb{R}$ and a covariance function $k : \mathbb{R}^q \times \mathbb{R}^q \to \mathbb{R}$, such that

$$p(\boldsymbol{f}) = \mathcal{N}(\boldsymbol{\mu}, \boldsymbol{K}), \tag{2}$$

where $\boldsymbol{\mu} = \big(\mu(\boldsymbol{z}_1, \dots, \mu(\boldsymbol{z}_N)\big)$ and $\boldsymbol{K}$ is the $N \times N$-matrix with (i, j)-th entry $k(\boldsymbol{z}_i, \boldsymbol{z}_j)$.

GPs are well-suited for Bayesian non-parametric regression, since if we condition on data $\mathcal{D} = \{\boldsymbol{z}, x\}$, where x denote the labels, then the posterior of $f(\boldsymbol{z}^*)$, at a test location $\boldsymbol{z}^*$, is given as

$$p\big(f(\boldsymbol{z}^*)|\mathcal{D}\big) = \mathcal{N}(\boldsymbol{\mu}^*, \boldsymbol{K}^*), \quad \begin{cases} \mu(\boldsymbol{z}^*) + \boldsymbol{\mu}^* = k(\boldsymbol{z}^*, \boldsymbol{z})^\top k(\boldsymbol{z}, \boldsymbol{z})^{-1} x, \\ \boldsymbol{K}^* = k(\boldsymbol{z}^*, \boldsymbol{z}^*) - k(\boldsymbol{z}^*, \boldsymbol{z})^\top k(\boldsymbol{z}, \boldsymbol{z})^{-1} k(\boldsymbol{z}^*, \boldsymbol{z}). \end{cases} \tag{3}$$

We see that this posterior computation involves inversion of the $N \times N$-matrix $\boldsymbol{K}$, which has complexity $\mathcal{O}(N^3)$. To overcome this computational burden in inference we consider variational sparse GP regression, which introduces M auxiliary points $\boldsymbol{u}$, that approximate the posterior of f with a variational distribution q. For a review of variational GP methods, we refer to Titsias [2009].

2.2 Riemannian Geometry

A *manifold* is a topological space, for which each point on it has a neighborhood that is homeomorphic to Euclidean space; that is, manifolds are locally linear spaces. Such manifolds can be embedded into spaces of higher dimension than the dimensionality of the associated Euclidean space; the manifold *itself* has the same dimension as the local Euclidean space. A q-dimensional manifold $\mathcal{M}$ can, for our purposes thus, be seen as a surface embedded in $\mathbb{R}^D$. In order to make quantitative statements along the manifold we require it to be *Riemannian*.

Definition 1. *A Riemannian manifold $\mathcal{M}$ is a smooth q-manifold equipped with an inner product*

$$\langle \cdot, \cdot \rangle_{\boldsymbol{x}} : \mathcal{T}_{\boldsymbol{x}}\mathcal{M} \times \mathcal{T}_{\boldsymbol{x}}\mathcal{M} \to \mathbb{R}, \qquad \boldsymbol{x} \in \mathcal{M}, \tag{4}$$

that is smooth in $\boldsymbol{x}$. Here $\mathcal{T}_{\boldsymbol{x}}\mathcal{M}$ denotes the tangent space of $\mathcal{M}$ evaluated at $\boldsymbol{x}$.

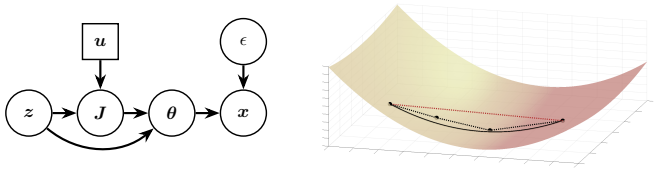


Figure 1: *Left:* A graphical representation of the model: $\boldsymbol{x}$ is the observational input, $\boldsymbol{J}$ is the Gaussian process manifold and $\boldsymbol{\theta}$ are the parameters it yields based on latent embedding $\boldsymbol{z}$. ϵ is a hyperparameter for the neighbor-graph embedding and $\boldsymbol{u}$ are variational parameters. *Right:* Illustration of the task: the dashed lines are Euclidean distances in three dimensions. The black ones are *neighbors* and their distance along the two-dimensional manifold should *match* the 3d-Euclidean distance. The red is not a neighbor-pair and the manifold distance should not match it.

The length of a curve is easily defined from the Riemannian inner product. If $\boldsymbol{c} : [0, 1] \to \mathcal{M}$ is a smooth curve, its length is given by $s = \int_0^1 \|\dot{\boldsymbol{c}}(t)\| \mathrm{d}t$. On an embedded manifold $f(\mathcal{M})$ this becomes

$$s = \int_0^1 \|\dot{f}(\boldsymbol{c}(t))\dot{\boldsymbol{c}}(t)\| \mathrm{d}t. \tag{5}$$

A metric on $\mathcal{M}$ can then be defined as

$$d_{\mathcal{M}}(\boldsymbol{x}, \boldsymbol{y}) = \inf_{\boldsymbol{c} \in C^1(\mathcal{M})} \big\{ s | \boldsymbol{c}(0) = \boldsymbol{x} \text{ and } \boldsymbol{c}(1) = \boldsymbol{y} \big\}, \qquad \boldsymbol{x}, \boldsymbol{y} \in \mathcal{M}. \tag{6}$$

2.3 The Nakagami distribution

We consider random manifolds immersed by a GP. The length of a curve (5) on such a manifold is necessarily random as well. Fortunately, since this manifold is a Gaussian field, then curve lengths are well-approximated with the Nakagami m-distribution [Bewsher et al., 2017].

The Nakagami distribution [Nakagami, 1960] describes the length of an isotropic Gaussian vector, but Bewsher et al. [2017] have meticulously demonstrated that this also provides a good approximation to the arc length of a GP. The Nakagami has density function

$$g(s) = \frac{2m^m}{\Gamma(m)\Omega^m} s^{2m-1} \exp\Big(-\frac{m}{\Omega} s^2 \Big), \qquad s \geq 0, \tag{7}$$

and it is parametrised by $m \geq 1/2$ and $\Omega > 0$; here Γ denotes the Gamma function. The parameters are interpretable by the equations

$$\Omega = \mathbb{E}[s^2] \quad \text{and} \quad m = \frac{\Omega^2}{\mathrm{Var}(s^2)}, \tag{8}$$

which can be used to infer the parameters through samples, although it does involve a fourth moment.

3 Model and variational inference

With prerequisites settled, we now set up a Gaussian process latent variable model that is *locally* distance preserving and *globally* topology preserving. Notation-wise we let $\mathcal{Z}$ denote the latent representation of a dataset $\mathcal{X} = \{\boldsymbol{x}_i\}_{i=1}^N$, $\boldsymbol{x_i} \in \mathbb{R}^D$, and let $f : \boldsymbol{z} \mapsto \boldsymbol{x}$ be the generative mapping.

3.1 Distance and topology preservation

The *manifold assumption* hypothesizes that high-dimensional data in $\mathbb{R}^D$ lie near a manifold with small intrinsic dimension. A manifold suggests that, a neighborhood around any point is approximately homeomorphic to a linear space. So nearby points are approximately linear, but non-nearby points have distances *greater* than the linear approximation suggests.

We shall build a Gaussian process latent variable model (GP-LVM) [Lawrence, 2005] that is explicitly designed for distance and topology preservation. The vanilla GP-LVM takes on the Gaussian likelihood where observations $\mathcal{X}$ are assumed i.i.d. when conditioned on a Gaussian process f. That is, $p(\mathcal{X}|f) = \prod_{i=1}^{N} p(\boldsymbol{x}_i|f(\boldsymbol{z}_i))$ and $p(\boldsymbol{x}_i|f(\boldsymbol{z}_i)) = \mathcal{N}(\boldsymbol{x}_i|f(\boldsymbol{z}_i), \sigma^2)$. In contrast, we consider a likelihood over pairwise distances between observations.

Neighborhood graph. To model locality, we condition our model on a graph embedding of the observed data $\mathcal{X}$. The graph is the ϵ-*nearest neighbor* embedded graph; that is, the undirected graph with vertices $V = \mathcal{X}$ and edges $E = \{e_{ij}\}$, where e_{ij} is in E, only if $d(\boldsymbol{x}_i, \boldsymbol{x}_j) < \epsilon$, for some metric d. Equivalently, $G = (V, E)$ can be represented by its adjacency matrix A_G with entries

$$a_{ij} = \mathbf{1}_{d(\boldsymbol{x}_i, \boldsymbol{x}_j) < \epsilon}. \tag{9}$$

In Sec. 3.5 we discuss how to choose ϵ informedly, but for now we view it as a hyperparameter.

Manifold distances. To arrive at a likelihood over pairwise distances, we first recall that the linear interpolation between $\boldsymbol{z}_i$ and $\boldsymbol{z}_j$ in the latent space has curve length

$$s_{ij} = \int_0^1 \|\boldsymbol{J}(\boldsymbol{c}(t))\dot{\boldsymbol{c}}(t)\| \mathrm{d}t, \quad \boldsymbol{c}(t) = \boldsymbol{z}_i(1-t) + \boldsymbol{z}_j t. \tag{10}$$

As the manifold distance $d_{\mathcal{M}}$ is the length of the shortest connecting curve, then s_{ij} is by definition an upper bound on $d_{\mathcal{M}}$. However, as the manifold is locally homeomorphic to a Euclidean space, then we can expect s_{ij} to be a good approximation of the distance to nearby points, i.e.

$$d_{\mathcal{M}}(\boldsymbol{z}_i, \boldsymbol{z}_j) \approx s_{ij} \qquad \text{for } \|\boldsymbol{x}_i - \boldsymbol{x}_j\| < \epsilon \tag{11}$$

$$d_{\mathcal{M}}(\boldsymbol{z}_i, \boldsymbol{z}_j) \leq s_{ij} \qquad \text{otherwise.} \tag{12}$$

The behavior we seek is that local interpolation in latent space should mimic local interpolation in data space only if the points are close in data space. If they are far apart, they should *repel* each other in the sense that the linear interpolation in latent space should have *large* curve length.

Censoring. To encode this behavior in the likelihood, we introduce *censoring* [Lee and Wang, 2003] into our objective function. This method is usually applied to missing data in survival analysis, when the event of something happening is known to occur later than some time point.

We may think of censoring as modeling inequalities in data. The censored likelihood function for i.i.d. data t_i following distribution function G_θ, with density function g_θ, is defined as

$$L(\{t_i\}_{i=1}^N|\theta, T) = \prod_{t_i<T} g_\theta(t_i) \prod_{t_i \geq T} (1 - G_\theta(T)), \tag{13}$$

where θ are the parameters of the distribution G and T is some 'time point', where the experiment ended. Carreira-Perpiñan [2010] remark that most neighborhood-embedding methods have loss functions with two terms: one attracting close point and one scattering term for far away connections. Censoring provides a *likelihood* with similar such terms.

Local distance likelihood. From earlier, we know that if the manifold $f(\mathcal{M})$ is a Gaussian field, then distances (10) are approximately Nakagami distributed. Thus, we write our likelihood as

$$L(\left\{\{e_{ij}\}_{i<j}\right\}_{i=1}^{N-1}|\theta, \epsilon) = \prod_{e_{ij}<\epsilon} g_\theta(e_{ij}) \prod_{e_{ij} \geq \epsilon} (1 - G_\theta(\epsilon)), \tag{14}$$

where G_θ is the distribution function of a Nakagami with parameters $\theta = \{m, \Omega\}$. Hence, the log-likelihood we shall maximise is

$$\begin{aligned} l\left(\left\{\{e_{ij}\}_{i<j}\right\}_{i=1}^{N-1} \middle| \theta, \epsilon\right) = &- \sum_{e_{ij}<\epsilon} \left(\log\Gamma(m_{ij}) + m_{ij} \log\left(\frac{\Omega_{ij}}{m_{ij}}\right) - (2m_{ij} - 1)\log(e_{ij}) + \frac{m_{ij} e_{ij}^2}{\Omega_{ij}} \right) \\ &- \sum_{e_{ij} \geq \epsilon} \left(\log\Gamma(m_{ij}) - \log\left(\Gamma(m_{ij}) - \gamma(m_{ij}, \frac{m_{ij}}{\Omega_{ij}} e_{ij}^2) \right) \right), \end{aligned} \tag{15}$$

where Γ and γ denotes the Gamma function and lower incomplete gamma function respectively and m_{ij} and Ω_{ij} are the Nakagami-parameters of Eq. 10.

Until now, we have introduced the log-likelihood based of an ϵ-NN graph, that preserves geometric features. Next we marginalize all other parameters to make a generative model.

3.2 Marginalizing the representation

We have a loss function (15) that matches distances e_{ij} with parameters $\theta_{ij} = \{m_{ij}, \Omega_{ij}\}$. We now seek to first fit these parameters and marginalize them to obtain a full generative approach. First, we will assume that conditioned on θ, we get the independent observations, i.e.

$$p(\mathcal{E}|\theta, \epsilon) = \prod_{1 \leq i < j \leq N} p(e_{ij}|\theta_{ij}, \epsilon) = L(\{\{e_{ij}\}_{i<j}\}_{i=1}^{N-1}|\theta, \epsilon), \tag{16}$$

as known from Eq. 14. We infer these parameters of the Nakagami by introducing a latent Gaussian field J and a latent representation $\boldsymbol{z}$. This allows us to define curve length (10), which we assume is also Nakagami distributed. In practice, we draw[1] m samples of s_{ij} from Eq. 10, and estimate the mean and variance of their second moment. This gives estimates of m_{ij} and Ω_{ij} via Eq. 8.

Essentially, we match distances on the manifold J with the observed distances $\mathcal{E}$. We marginalize this manifold

$$p(\mathcal{E}|\boldsymbol{z}) = \int p(\mathcal{E}|\theta)p(\theta|\boldsymbol{J}, \boldsymbol{z})p(\boldsymbol{J})\mathrm{d}\theta\mathrm{d}\boldsymbol{J}, \qquad \text{where} \tag{17}$$

$$p(\theta|\boldsymbol{J}, \boldsymbol{z}) := \int p(\theta|s)p(s|\boldsymbol{J}, \boldsymbol{z})\mathrm{d}s, \quad \text{and} \quad p(\theta|s) = \begin{cases} \delta_{\mathbb{E}s^2}(\Omega) \\ \delta_{\Omega/\mathrm{Var}(s^2)}(m), \end{cases} \tag{18}$$

where δ denotes the Dirac probability measure and $p(s|\boldsymbol{J}, \boldsymbol{z})$ is the approximate Nakagami distribution (10). This means that s_{ij} and e_{ij} are both Nakagami variables that share the same parameters, which interpretively means the manifold distances s_{ij} match the embedding distances e_{ij}.

Further, to make it generative, we can pose a prior on $\boldsymbol{z}$ and marginalize this in Eq. 17. We infer everything variationally [Blei et al., 2017], and choose a variational distribution over the marginalized variables. We approximate the posterior $p(\theta, \boldsymbol{J}, \boldsymbol{z}, \boldsymbol{u}|\mathcal{E})$ with

$$q(\theta, \boldsymbol{J}, \boldsymbol{z}, \boldsymbol{u}) := q(\theta|\boldsymbol{J}, \boldsymbol{z})q(\boldsymbol{J}, \boldsymbol{u})q(\boldsymbol{z}), \tag{19}$$

where $\boldsymbol{u}$ is an inducing variable [Titsias, 2009], and

$$q(\theta|\boldsymbol{J}, \boldsymbol{z}) = p(\theta|\boldsymbol{J}, \boldsymbol{z}), \quad q(\boldsymbol{J}, \boldsymbol{u}) = p(\boldsymbol{J}|\boldsymbol{u})q(\boldsymbol{u}) \quad \text{and} \quad q(\boldsymbol{z}) = \mathcal{N}(\boldsymbol{\mu}_z, \boldsymbol{A}_z), \tag{20}$$

where $\boldsymbol{\mu}_z$ is a vector of size N and $\boldsymbol{A}_z$ is a diagonal $N \times N$-matrix. Further $q(\boldsymbol{u}) = \mathcal{N}(\boldsymbol{\mu}_u, \boldsymbol{S})$ is a full M-dimensional Gaussian.

This allow us to bound the log-likelihood (15), with the evidence lower bound (ELBO)

$$\log p(\mathcal{E}) = \log \int \frac{p(\mathcal{E}, \theta, \boldsymbol{J}, \boldsymbol{z}, \boldsymbol{u})}{q(\theta, \boldsymbol{J}, \boldsymbol{z}, \boldsymbol{u})} q(\theta, \boldsymbol{J}, \boldsymbol{z}, \boldsymbol{u})\mathrm{d}\theta\mathrm{d}\boldsymbol{J}\mathrm{d}\boldsymbol{u}\mathrm{d}\boldsymbol{z} \tag{21}$$

$$\geq \mathbb{E}_\theta[l(\mathcal{E}|\theta)] - \mathrm{KL}\big(q(\boldsymbol{u})||p(\boldsymbol{u})\big) - \mathrm{KL}\big(q(\boldsymbol{z})||p(\boldsymbol{z})\big), \tag{22}$$

where both KL-terms are analytically tractable, but the first term has to be approximated using Monte Carlo. The right hand side here is readily optimized with gradient descent type algorithms.

In summary, we have a latent representation $\mathcal{Z}$ and a Riemannian manifold immersed as a GP $\boldsymbol{J}$. This implies that between any two points $\boldsymbol{z}_i$ and $\boldsymbol{z}_j$, we can compute s_{ij}, which is approximately Nakagami. With censoring we can match s_{ij} with observation e_{ij}, if $e_{ij} < \epsilon$; else we push s_{ij} to have all its mass on $[\epsilon, \infty)$. It is optimized with variational inference, by maximizing Eq. 22.

3.3 Generating new samples

All inference thus far has been done in a *coordinate-free* manner; in other words, we have yet to embed our manifold $f(\mathcal{M})$ in $\mathbb{R}^D$. We can do this embedding with Euclidean isometries, translation and rotation, and inspired by the fundamental theorem of analysis

$$f(\boldsymbol{z}_i) = f(\boldsymbol{z}_j) + \int_0^1 j(\boldsymbol{c}(t))\dot{\boldsymbol{c}}(t)\mathrm{d}t, \quad \boldsymbol{c}(t) = \boldsymbol{z}_j(1-t) + \boldsymbol{z}_i t. \tag{23}$$

[1] We can approximate s by finely discretizing $\boldsymbol{c}$ and sum over the integrand.

In this view, the translation part can be done by the original points, as we assume $f(\boldsymbol{z}_j) \approx \boldsymbol{x}_j$ and we can, for a new point $\boldsymbol{z}^*$, define a generator as

$$x^* := f(\boldsymbol{z}_i) + \mathfrak{R}(\boldsymbol{z}_i) \int_0^1 \boldsymbol{J}(\boldsymbol{c}(t))\dot{\boldsymbol{c}}(t)\mathrm{d}t, \quad \boldsymbol{c}(t) = \boldsymbol{z}_i(1-t) + \boldsymbol{z}^* t, \tag{24}$$

where $\mathfrak{R}$ is a $D \times D$ rotation-matrix, that can be optimized to best fit with the original data $\mathcal{D}$ and $\boldsymbol{J}$ is the inferred Jacobian from Eq. 22. This is a rather naive way, since it needs many local embeddings and follows the intuition of Zhang and Zha [2003]. A more principled way would be to learn an isometry f by regression methods.

3.4 Invariance learning and geometric constraints

Why is it worth learning the manifold in a coordinate-free way, if we still need to fit values afterwards? Invariances are easily encoded via dissimilarity pairs by introducing equivalence classes in saying $d(\boldsymbol{x}_i, \boldsymbol{x}_j) = 0$ if $\boldsymbol{x}_i$ and $\boldsymbol{x}_j$ are in the same equivalence class. Popular choices of such equivalence classes are rotations, translations and scaling. Many constraints one could wish to impose on models can be formulated as geometric constraints. It holds true also for GPLVM-based models as seen in Urtasun et al. [2008], who wish to encode topological information, and Zhang et al. [2010], who highlight invariant models' usefulness in causal inference. Geometric constraints can alternatively be encoded with GPs that take their output directly on a Riemannian manifold [Mallasto et al., 2018].

The geometry of latent variable models in general is an active field of study [Arvanitidis et al., 2018, Tosi et al., 2014], and Simard et al. [2012] and Kumar et al. [2017] argues that the tangent (Jacobian) space serves a convenient way to encode invariances.

3.5 Topological Data Analysis

The model is naturally affected by the hyperparameter ϵ. We argue that it can be chosen in a geometrically founded way using Topological Data Analysis [Carlsson, 2009]. By constructing a *Rips diagram* [Fasy et al., 2014] one can find ϵ such that the ϵ-NN graph captures the right topology of data. It is beyond this paper to summarize the techniques; we refer readers to Chazal and Michel [2017].

4 Experiments

We perform experiments first on a classical toy dataset and on the image dataset MNIST. We refer to the presented model as *Isometric Gaussian Process Latent Variable Model* (Iso-GPLVM). For some comparisons we evaluate other models also based on dissimilarity data. In all cases we initialize Iso-GPLVM with IsoMap, as it is known that GP-based methods are sensitive to initializations [Bitzer and Williams, 2010]. We use the Adam-optimizer [Kingma and Ba, 2014] with a learning rate of $3 \cdot 10^{-3}$ and optimize sequentially $q(\boldsymbol{z})$ and $q(\boldsymbol{u})$ separately. We use $m = 100$ inducing points for $q(\boldsymbol{u})$ and an ARD-kernel as covariance function for the GP.

4.1 Swiss roll

The 'swiss roll' was introduced by Tenenbaum et al. [2000] to highlight the difficulties of non-linear manifold learning. The point cloud resides on a 2-dimensional manifold embedded in $\mathbb{R}^3$ and can be thought as a paper rolled around itself (see Fig 2A).

We find a 2-dimensional latent embedding by four methods: classical MDS, t-SNE, IsoMap and Iso-GPLVM. From Fig. 2 we observe the linear MDS is unable to capture the highly non-linear manifold. t-SNE captures some local structure, but the global outlook is far from the ground truth. We tried with several tunings of the perplexity hyperparameter (60 in the plot), but none of them successfully captured the structure. It is known that t-SNE is prone to make clusters, even if clusters are not a natural part of a dataset [Amid and Warmuth, 2018].

Naturally, as the dataset was constructed for the 'geodesic' approach of IsoMap, this captures both global and local structure. On closer inspection, we see the linear interpolations, stemming from Dijkstra's algorithm, leaves some artificial 'holes' in the manifold. Hence, on a smaller scale it can be argued that he topology of the manifold is captured imperfectly. The plot suggests Iso-GPLVM closes these holes and approximates the topology of an unfolded paper. We used $\epsilon = 0.4$.

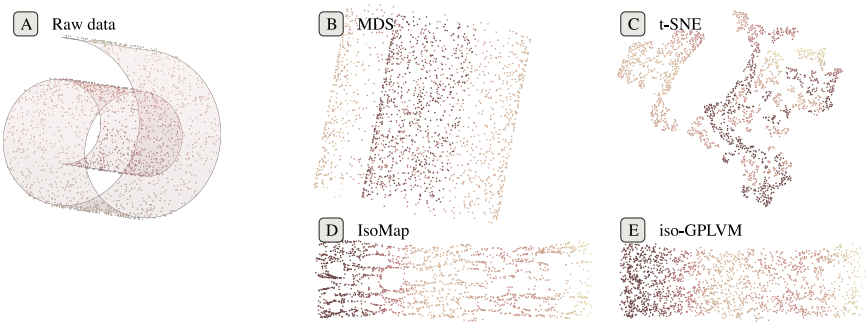


Figure 2: Data (A) and embeddings (B–E). All embeddings are shown with a unit aspect ratio to highlight that only IsoMap (D) and Iso-GPLVM (E) recover the elongated structure of the swiss roll.

4.2 MNIST

Metrics. We evaluate our model on 5000 images from MNIST, and we foremost wish to highlight how invariances can be encoded with dissimilarity data. In particular, we consider fitting our model to data under three different distance measures. We consider the classical Euclidean distance measure

$$d(\boldsymbol{x}_i, \boldsymbol{x}_j) = \|\boldsymbol{x}_i - \boldsymbol{x}_j\|. \tag{25}$$

Further, we consider a metric that is invariant under image rotations

$$d_{\text{ROT}}(\boldsymbol{x}_i, \boldsymbol{x}_j) = \inf_{\theta \in [0, 2\pi)} \Big\{ d\left(R_\theta(\boldsymbol{x}_i), \boldsymbol{x}_j\right) \Big\}, \tag{26}$$

where R_θ rotates an image by θ radians. We note $d_{\text{ROT}}(\boldsymbol{x}_i, \boldsymbol{x}_j) \leq d(\boldsymbol{x}_i, \boldsymbol{x}_j)$ always. Finally, we introduce a *lexicographic* metric [Rodriguez-Velazquez, 2018]

$$d_{\text{LEX}}(\boldsymbol{x}_i, \boldsymbol{x}_j) = \begin{cases} r, & \text{if } y_i \neq y_j, \\ \min\{2r,\ d(\boldsymbol{x}_i, \boldsymbol{x}_j)\}, & \text{if } y_i = y_j, \end{cases} \tag{27}$$

which in the censoring phase enforce images carrying different labels to repel each other. This is a handy way to encode a topology or clustering based on discrete variables, when such are available. For all metrics, we have normalized the data and have set $r = 7$.

Results. Figure 3(A—C) show the latent embeddings of the three metrics. The background color is the measure $\mathbb{E}\left[\sqrt{\det(\boldsymbol{J}^\top \boldsymbol{J})}\right]$, which provide a view of the Riemannian geometry of the latent space. Bishop et al. [1997] call this measure the *magnification factor*. Large values (light color) imply that trajectories moving in this area are longer and likely also more uncertain [Hauberg, 2018].

Panels A, D and E base their latent embedding on the Euclidean metric. We observe that IsoMap (D) and Iso-GPLVM (A) appear similar in shape, unsurprisingly as we initialize with IsoMap, but Iso-GPLVM finds a cleaner separation of the digits. Particularly, this is evident for the *six, three and eight digits*. The *fives* seem to group into several tighter cluster, and this behavior is found for t-SNE as well. Overall, from a clustering perspective, t-SNE visually is superior; but distances *between* clusters in (A) can be larger than the straight lines that connect them. This is evident from the lighter background color between cluster, say, *zeros* and *threes*. We note that IsoMap and t-SNE has no associated Riemannian metric and as such distances between any input cannot be computed.

The rotation invariant metric result in a latent embedding where different classes significantly overlap. Upon closer inspection we, however, note several interesting properties of the embedding. *Zero digits* are well separated from other classes as a rotated 0 does not resemble any other digits; the *one digits* form a cluster that is significantly more compact than other digits as there is limited variation left after rotations have been factored out; *two and five digits* significantly overlap, which is most likely due to 5 digits resembling 2 digits when rotated $180°$; similar observations hold for the *four, nine and six digits*; and a partial overlap between *three and eight digits* as is often observed. The overall darker background is due to the rotational invariant metric being shorter than the Euclidean counterpart.

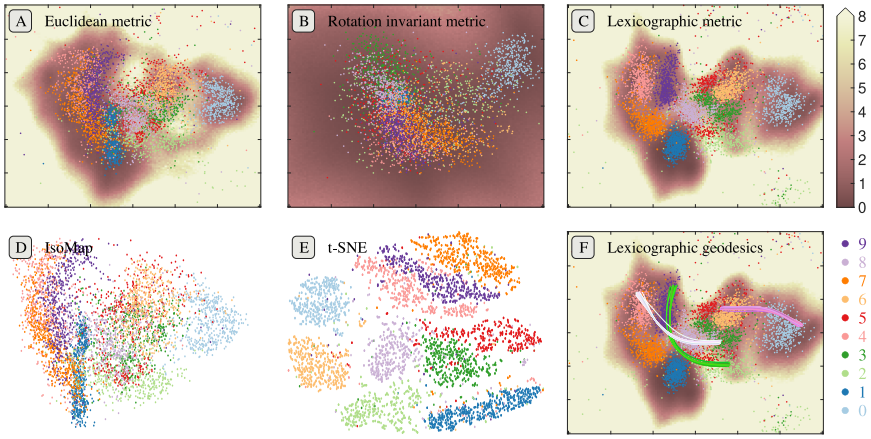


Figure 3: Embeddings of MNIST attained with our method under different metrics (A—C) and for baselines IsoMap (D) and t-SNE (E). The background color show the expected volume measure associated with the Riemannian metric $\mathbb{E}\left[\sqrt{\det(\boldsymbol{J}^\top \boldsymbol{J})}\right]$. A large measure generally indicate high uncertainty of the manifold. Panel F shows Riemannian geodesics under the lexicographic metric.

In terms of clustering the lexicographic approach outshines the other metrics. This is expected as the metric use label information, but neatly illustrate how domain-specific metrics can be developed from weak or partial information. Most classes are well-separated except for a region in the middle of the plot. Note how this region has high uncertainty.

The Riemannian geometry of the latent space imply that geodesics (shortest paths) can be computed in our model. Figure 3F shows example geodesics under the lexicographic metric. Their highly non-linear appearance emphasizes the curvature of the learned manifold. The green geodesics has one endpoint in a cluster of nine digits and move along this cluster avoiding the uncertain area of eights and fives, as opposed on linearly interpolating through them.

5 Discussion

We introduced a model for non-linear dimensionality reduction from dissimilarity data. It is the first of its kind based on Gaussian processes. The non-linearity of the method stems both from the Gaussian processes, but also from the censoring in the likelihood. It unifies ideas from Gaussian processes, Riemannian geometry and neighborhood graph embeddings. Unlike traditional manifold learning methods that embed into $\mathbb{R}^q$, we embed into a q-dimensional Riemannian manifold through the learned metric. This allows us to learn latent representations that are isometric to the true underlying manifold.

The model does have limitations. The generation of new samples was only naively considered, and further research of how to isometrically embed a manifold $\mathcal{M}$ into $\mathbb{R}^D$ to fit with observation is warranted. Existence is ensured as the observed data manifold is one such embedding. The Nakagami distribution that approximates the arc lengths of Gaussian processes is prone to overestimate the variance [Bewsher et al., 2017] and better approximations would improve our method. Further, the model inherits problems of optimizing the latent variables and it has previously been noted that good performance in this regime is linked with good initialization [Bitzer and Williams, 2010].

Our experiments highlight that Iso-GPLVM can learn the geometry of data and geometric constraints are easier encoded by learning a manifold contra doing GP regression. The uncertainty quantification associated with GPs follow through and further highlights the connection between uncertainty, geometry and topology. To the best of our knowledge, our model is the first of its kind that, locally, can asses the quality of the manifold approximation through the associated Riemannian measure.

6 Broader Impact

We present a general methodology for learning low-dimensional representations from pairwise distances, such that the associated model is fully generative and both distance and topology preserving. The model is further suitable for encoding *a priori* known invariances through a choice of metric. The contribution is largely methodological.

We envision the model being applied for data where it is easier to express prior knowledge through the design of an appropriate distance function. For instance, in much biological data there is side-information regarding the underlying evolutionary structure, which can be used to develop suitable evolutionary metrics.

The flexibility of the approach does open the door for misuse. For instance misleading visualizations (of the latent variables) can be easily created by a malicious choice of metric. The lexicographic example illustrates this potential misuse as one can imagine forcing groups apart with this mechanism, even if the data disapproves such groupings.

Acknowledgements

This project has received funding from the European Research Council (ERC) under the European Union's Horizon 2020 research and innovation programme (grant agreement no 757360). MJ and SH were supported in part by a research grant (15334) from VILLUM FONDEN.

References

E. Amid and M. K. Warmuth. A more globally accurate dimensionality reduction method using triplets. *arXiv:1803.00854 [cs]*, Mar. 2018.

G. Arvanitidis, L. K. Hansen, and S. Hauberg. Latent space oddity: On the curvature of deep generative models. In *6th International Conference on Learning Representations, ICLR 2018*, 2018.

Y. Bengio, A. Courville, and P. Vincent. Representation Learning: A Review and New Perspectives. *IEEE Trans. Pattern Anal. Mach. Intell.*, 35(8):1798–1828, Aug. 2013.

J. Bewsher, A. Tosi, M. Osborne, and S. Roberts. Distribution of gaussian process arc lengths. In *Artificial Intelligence and Statistics*, pages 1412–1420, 2017.

C. M. Bishop, M. Svens' en, and C. K. Williams. Magnification factors for the som and gtm algorithms. In *Proceedings 1997 Workshop on Self-Organizing Maps*, 1997.

S. Bitzer and C. K. Williams. Kick-starting gplvm optimization via a connection to metric mds. In *NIPS 2010 Workshop on Challenges of Data Visualization*, 2010.

D. M. Blei, A. Kucukelbir, and J. D. McAuliffe. Variational inference: A review for statisticians. *Journal of the American statistical Association*, 112(518):859–877, 2017.

G. Carlsson. Topology and data. *Bulletin of the American Mathematical Society*, 46(2):255–308, 2009.

M. Á. Carreira-Perpiñan. The elastic embedding algorithm for dimensionality reduction. In *Proceedings of the 27th International Conference on International Conference on Machine Learning*, pages 167–174, 2010.

F. Chazal and B. Michel. An introduction to topological data analysis: fundamental and practical aspects for data scientists, 2017.

B. T. Fasy, F. Lecci, A. Rinaldo, L. Wasserman, S. Balakrishnan, and A. Singh. Confidence sets for persistence diagrams. *Ann. Statist.*, 42(6):2301–2339, 12 2014. doi: 10.1214/14-AOS1252. URL `https://doi.org/10.1214/14-AOS1252`.

I. Goodfellow, J. Pouget-Abadie, M. Mirza, B. Xu, D. Warde-Farley, S. Ozair, A. Courville, and Y. Bengio. Generative Adversarial Nets. In *Advances in Neural Information Processing Systems (NIPS)*, 2014.

S. Hauberg. Only bayes should learn a manifold (on the estimation of differential geometric structure from data). *arXiv preprint arXiv:1806.04994*, 2018.

D. P. Kingma and J. Ba. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*, 2014.

D. P. Kingma and M. Welling. Auto-Encoding Variational Bayes. In *Proceedings of the 2nd International Conference on Learning Representations (ICLR)*, 2014.

A. Kumar, P. Sattigeri, and T. Fletcher. Semi-supervised learning with gans: Manifold invariance with improved inference. In *Advances in Neural Information Processing Systems*, pages 5534–5544, 2017.

N. Lawrence. Probabilistic non-linear principal component analysis with Gaussian process latent variable models. *Journal of machine learning research*, 6(Nov):1783–1816, 2005.

E. T. Lee and J. Wang. *Statistical Methods for Survival Data Analysis*, volume 476. John Wiley & Sons, 2003.

L. v. d. Maaten and G. Hinton. Visualizing data using t-sne. *Journal of machine learning research*, 9(Nov): 2579–2605, 2008.

A. Mallasto, S. Hauberg, and A. Feragen. Probabilistic riemannian submanifold learning with wrapped gaussian process latent variable models. In *Proceedings of the 19th international Conference on Artificial Intelligence and Statistics (AISTATS)*, 2018.

P.-A. Mattei and J. Frellsen. Miwae: Deep generative modelling and imputation of incomplete data sets. In *International Conference on Machine Learning*, pages 4413–4423, 2019.

L. McInnes, J. Healy, and J. Melville. Umap: Uniform manifold approximation and projection for dimension reduction. *arXiv preprint arXiv:1802.03426*, 2018.

M. Nakagami. The m-distribution—a general formula of intensity distribution of rapid fading. In *Statistical Methods in Radio Wave Propagation*, pages 3–36. Elsevier, 1960.

C. Rasmussen and C. Williams. *Gaussian Processes for Machine Learning*. Adaptive Computation and Machine Learning. MIT Press, Cambridge, MA, USA, 2006.

D. J. Rezende and S. Mohamed. Variational inference with normalizing flows. *arXiv preprint arXiv:1505.05770*, 2015.

D. J. Rezende, S. Mohamed, and D. Wierstra. Stochastic Backpropagation and Approximate Inference in Deep Generative Models. In *Proceedings of the 31st International Conference on Machine Learning (ICML)*, 2014.

B. D. Ripley. *Pattern recognition and neural networks*. Cambridge university press, 2007.

J. A. Rodriguez-Velazquez. Lexicographic metric spaces: Basic properties and the metric dimension, 2018.

P. Y. Simard, Y. A. LeCun, J. S. Denker, and B. Victorri. Transformation invariance in pattern recognition–tangent distance and tangent propagation. In *Neural networks: tricks of the trade*, pages 235–269. Springer, 2012.

J. B. Tenenbaum, V. de Silva, and J. C. Langford. A Global Geometric Framework for Nonlinear Dimensionality Reduction. *Science*, 290(5500):2319, 2000.

M. Titsias. Variational learning of inducing variables in sparse gaussian processes. In *Artificial Intelligence and Statistics*, pages 567–574, 2009.

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

A. Tosi, S. Hauberg, A. Vellido, and N. D. Lawrence. Metrics for Probabilistic Geometries. In *The Conference on Uncertainty in Artificial Intelligence (UAI)*, July 2014.

R. Urtasun and T. Darrell. Discriminative gaussian process latent variable model for classification. In *Proceedings of the 24th International Conference on Machine Learning*, pages 927–934, 2007.

R. Urtasun, D. J. Fleet, A. Geiger, J. Popović, T. J. Darrell, and N. D. Lawrence. Topologically-constrained latent variable models. In *Proceedings of the 25th International Conference on Machine Learning*, pages 1080–1087, 2008.

K. Zhang, B. Schölkopf, and D. Janzing. Invariant gaussian process latent variable models and application in causal discovery. In *Proceedings of the Twenty-Sixth Conference on Uncertainty in Artificial Intelligence*, pages 717–724, 2010.

Z. Zhang and H. Zha. Principal manifolds and nonlinear dimension reduction via local tangent space alignment. *SIAM J. Sci. Comput.*, 26, 01 2003. doi: 10.1137/S1064827502419154.

Paper D

Reparametrization Invariance for non-parametric Causal Discovery

Reparametrization Invariance for non-parametric Causal Discovery

Martin Jørgensen and Søren Hauberg
marjor@dtu.dk sohau@dtu.dk
Technical University of Denmark

Abstract

Causal discovery estimates the underlying physical process that generates the observed data: does X cause Y or does Y cause X? Current methodologies use structural conditions to turn the causal query into a statistical query, when only observational data is available. But what if the these statistical queries are sensitive to causal invariants? This study investigates one such invariant: *the causal relationship between X and Y is invariant to the marginal distributions of X and Y*. We propose an algorithm that use a non-parametric estimator that is robust to changes in the marginal distributions. This way we may marginalize the marginals, and inspect what relationship is intrinsically there. The resulting causal estimator is competitive with current methodologies and has high emphasis on the *uncertainty* in the causal query; an aspect just as important as the query itself.

1 Introduction

Determining causal relationships is a constant challenge, and the ultimate goal of the natural sciences. The gold standard for establishing such relationships is *intervention studies*, where the physical state of a system is manually modified to determine whether this changes the system behavior. Such experiments are, however, often infeasible as the interventions can be unethical, physically impossible, expensive and so forth. This begs the question of whether causal relationships can be estimated from data in a systematic manner. Most work in this direction has been for high-dimensional data used to estimate directed acyclic graphs (DAGs), but in recent years the most simple of these, the two-vertex DAG, has gained more attention. The methods for determining these causal bindings go under the name of *causal discovery*, and the usual approach is to assume some structural equation model, and probabilistically verify its existence.

By assuming a particular model, it becomes possible to establish conditions under which the causal direction is unique, thereby providing a formalism to the causal question. From a practical point of view, this formalism is, however, only useful when the structural model assumption is known to be true, which is seldom the case.

In this paper, we explore the case of bivariate causal inference when model assumptions are challenged by shifts in marginal distributions. We propose an estimator

based on comparing regression errors, as in Blöbaum et al. [1], but in a non-parametric way. This provides an estimator that is more robust to these distributional shift than well-known methods for bivariate causal discovery, while staying on-par in performance.

1.1 Related Work

In his seminal work, Pearl [2] introduced causal inference for high-dimensional observational data, phrased as the estimation of a causal structure. This is a DAG, where random variables are nodes and an edge $X \to Y$ indicates that X is a (direct) cause of Y. Given more than three variables, such edges can be estimated through conditional independence tests, e.g. an edge between X and Y can be discarded if they are independent conditioned on a third variable Z. This idea, however, breaks down in the bivariate case, which is the main focus of the present paper.

In the bivariate case, one usually must impose assumptions that break the symmetry of correlation. This is achieved by assuming two models — one for $X \to Y$ and another for $Y \to X$ — and choosing among these either by 1) verifying exactly one of the underlying models or 2) proposing a score/complexity measure for choosing the simplest model following the principles of Occam's razor.

Model Verification [3–5]: It is natural to assume an *additive noise model (ANM)* [3, 4], i.e. $Y = f(X) + N_Y$, where $N_Y \perp\!\!\!\perp X$. Hoyer et al. [3] show that when f is nonlinear, then the true causal direction can be identified. Similar results hold when f is linear and the noise is non-Gaussian [4]. However, if the underlying system is not an ANM, the analysis is inapplicable – e.g. in the presence of hidden confounders. Zhang and Hyvärinen [5] extend the ANM to allow for an unknown bijective mapping of the observations and show that this structure is identifiable for many joint distributions $\mathbb{P}_{(X,Y)}$.

Model Scoring [1, 6–8]: An intuitive scoring mechanism is to regress Y from X and vice versa and ask which direction has higher likelihood. This is, e.g., implemented by Mooij et al. [8] who propose using a Gaussian Process Latent Variable Model [9] to handle the noise/latent observations. The chosen causal direction must then be *biased* towards the prior over the latent points and sensitive to hyperparameters, which is an implicit model assumption.

Blöbaum et al. [1] take an approach based on asymmetry of regression error, and show that this asymmetry is coherent with the causal direction under certain assumptions, of which the most important are the independence of the cause and the causal *mechanism* [10] and that this mechanism is monotonic as a function of the cause. Loosely, they show that when the noise is sufficiently small and $X \to Y$, then

$$\mathbb{E}[\mathrm{Var}(Y|X)] \leq \mathbb{E}[\mathrm{Var}(X|Y)]. \tag{1}$$

They quantify these measures by parametric regression.

Janzing et al. [7] propose the *Information Geometric Causal Inference* (IGCI) scoring mechanism. This is derived from the assumption that data is noise free, i.e. $Y = f(X)$, and on the postulate that the true causal mechanism f is independent of the

cause X. This is realized by non-parametrically estimating the expected log-derivative of f:

$$\mathbb{E}[\log|f'|] \approx \frac{1}{N-1}\sum_{i=1}^{N-1}\log\frac{|y_{i+1}-y_i|}{x_{i+1}-x_i}, \tag{2}$$

where $x_{i+1} > x_i$ for $i = 1,\ldots,N-1$, and both X and Y have been preprocessed to make them comparable, i.e. standardized wrt. a Gaussian or a uniform base measure. The direction with the smallest log-derivative is then chosen as being causal. While this mechanism provides no guarantees in the presence of noise, IGCI has been successful on real world data; in Sec. 3.4 we, however, demonstrate that this success is likely due to a bias in the studied benchmark data.

1.2 Causal invariant

As seen above, current causal inference propose one model for each causal direction, and then select among them. This begs the questions, *what if the data does not support either model?* and *can causal relationships be discovered without restrictive model assumptions?* If we believe that the causal and probabilistic domains abide by different rules, then our causal estimators should follow other paradigms than model verification/selection. We can think of this as *model-bias*: many existing methods are too sensitive to distributional and structural restrictions of probabilistic models. By this we mean that the *hypothesis* of causality is tested in a domain sensitive to marginal distributions and structural equations.

We recap the basic definition of causality as expressed by the *do-calculus* [2].

Definition 1. *If for some $x \neq \hat{x}$, we have that $\mathbb{P}(Y|\mathrm{do}(x)) \neq \mathbb{P}(Y|\mathrm{do}(\hat{x}))$, then X is a cause of Y.*

The interventional distribution, $\mathbb{P}(Y|\mathrm{do}(x))$, is only attainable if before the experiment is conducted the experimenter has made sure $X = x$, i.e. the experimenter has *intervened*. If the above definition is satisfied, we denote this by $X \to Y$. It is immediately clear that the above definition can hold in both directions. Further, for the task at hand, to estimate the causal direction from $\mathbb{P}_{(X,Y)}$, without access to the interventional distribution apparent in the definition, one can only make qualified guesses.

Imposing model assumptions can, in the spirit of Occam's razor, be seen as qualified guessing. However, any such a priori interpretation of the data will bias the causal prediction. To minimize such bias, we advocate a bivariate causal inference approach that tries to stay clear of scores tied to probabilistic models, and only rely on a test statistic that is well-defined for almost all datasets.

Like Pearl [2], we consider causal structures that are DAGs. Then, if $X \to Y$, we must also have $X \to g(Y)$ for any function g, since the contrary would construct a cycle. If g is a bijection, this is equivalent to $f(X) \to Y$, where $f = g^{-1}$. This motivates our guiding principle.

Principle A (Invariant causality)**.** *A deterministic bijective reparametrization of the observed variables does not change the causal direction.*

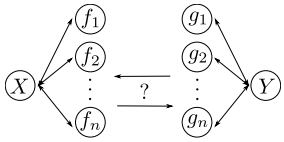


Figure 1: Visualisation of Principle A. If X is a cause of Y, then *theoretically* arrows from f_i to g_i must point right for *all* $i = 1, \ldots, n$. We suggest to test this *empirically*.

We only consider bijections, as we *a priori* do not know if $X \to Y$ or $Y \to X$. The above then states that the causal relationship between X and Y is the same as between $f(X)$ and $g(Y)$, for bijections f and g. Equivalently, our choice of units should not influence the causal direction; i.e., the marginal distributions of X and Y must not matter. Note that most model-based causal inference schemes are not closed under nonlinear reparametrizations and, hence, violate Principle A. For instance, a nonlinear reparametrization of an ANM does not yield another ANM.

Principle A is illustrated in Figure 1. If for some f_i and g_j, where $i, j = 1, \ldots, n$, we have that g_j is a cause of f_i, then X is not a cause of Y. Likewise, if f_i is a cause of g_j, then Y can not be a cause of X, since X there is a direct path from X to Y in the causal graph. Our idea is to construct n bijections of both X and Y, and test the causal relationship among these. If the decisions are unanimous, the causal link is likely strong. If they are inconsistent, this gives uncertainty in the causal estimator and we may interpret this inconsistency over bijections as uncertainty associated with the causal decision making.

This discussion of invariances in causal estimators has not involved how to realize Principle A. To this end, we consider the setup from Blöbaum et al. [1]. The inequality in Eq. 1 is shown to hold for small noise settings, when the condition

$$\mathrm{Cov}\left(\frac{\partial \mathbb{E}[Y|X=x]}{\partial x}, \mathbb{E}[\mathrm{Var}(Y|X=x)]p_X(x)\right) = 0, \tag{3}$$

is satisfied. Here p_X denotes the marginal distribution of the cause X. This criterion is similar to IGCI's idea that the expected log-derivative of the conditional mean is uncorrelated with the marginal distribution of the cause, and positively correlated in the anti-causal direction. These '*uncorrelated mechanism*' ideas [11] fall under the causal principle of modularity and autonomy. For a broader review see Peters et al. [10].

In summary, Blöbaum et al. [1] prove that under similar conditions to what we shall impose, then the prediction error is greater in the anti-causal direction compared to the causal – at least when the noise is small. Experimentally, they do regression by predetermined *types*, such as polynomial or neural nets. We are interested in marginalising the underlying distribution, thus it is not obvious that some parametric form of regression should be robust to this. In the next section, we present a non-parametric estimator of the regression error. This should be seen as a means to realizing Principle A. If anything, causal inference is about decision-making under imperfect or uncertain information.

These are the outlines of the present work, which we use to derive a simple causal inference scheme (Sec. 2). We evaluate this scheme in Sec. 3 and find that the empirical

performance is on par with current standard methodologies, but with the additional benefit that we provide well-calibrated uncertainties over causal predictions. On this path, we further derive and validate an extension to handle more than two variables and find that this establishes a link between our proposed estimator and classic conditional independence tests for causal structures [2]. All proofs are in the supplementary materials.

2 Quadratic Variation in Causal Discovery

If $f(X)$ is a predictor of Y, then $f(X)$ (trivially) correlates with Y. This motivates us to measure the correlation between Y and the predictor $\mathbb{E}[Y|X]$. We will show that we can quantify this completely non-parametrically, i.e. not making distributional assumptions on X and Y, besides finite second moments. In Sec. 2.2 we will show how this also allows us to apply Principle A.

To derive an estimator of this correlation, we first recap some theory from stochastic processes. Let Y_t denote a real-valued stochastic process on some probability space, and $t > 0$. The *quadratic variation* [12] of Y_t is the increasing process defined as

$$\langle Y \rangle_t := \lim_{S \to 0} \sum_{i=1}^{n} (Y_{t_i} - Y_{t_{i-1}})^2, \tag{4}$$

where S is the mesh[1] of partitions of the interval $[0, t]$. We define the *mean quadratic variation (MQV)* as the scaling $\langle Y \rangle_t / t$, which can be seen as a measure of averaged noise over the time interval $[0, t]$. Notice that estimators akin to Eq. (4) for non-time series are well-known in non-parametric regression [13].

For the problem at hand, consider two real-valued random variables X and Y from a joint distribution $\mathbb{P}_{(X,Y)}$. Similar to other methods of causal discovery, we shall see Y as a function of X, and vice versa. In particular we view it as a stochastic process on the interval $\text{supp}(X)$, which we assume to be bounded.

Theorem 2.1. *Let X have support on a compact and connected subset C of $\mathbb{R}$, and assume that $\mathbb{E}[Y|X = x]$ is a continuous differentiable function over C. Assume $\mathbb{E}Y^2 < \infty$. Let further (x_i, y_i), $i = 1, \ldots, N$, be iid samples from $\mathbb{P}_{(X,Y)}$. If we order, such that $x_{i+1} \geq x_i$ for all $i = 1, \ldots, N-1$, then it holds that*

$$\frac{1}{N-1} \sum_{i=1}^{N-1} (y_{i+1} - y_i)^2 \to 2\mathbb{E}\text{Var}(Y|X), \tag{5}$$

as $N \to \infty$.

Theorem 2.1 motivates computing the following quantity for unit variance observations

$$C_{X \to Y} := 1 - \frac{1}{2(N-1)} \sum_{i=1}^{N-1} \big(y_{i+1} - y_i\big)^2, \tag{6}$$

[1] For a partition $0 < t_1 < t_2 < \ldots < t_N < t$, we denote the mesh as the longest distance between two points $\max\{(t_{i+1} - t_i) | i = 1, \ldots, N-1)\}$.

since

$$C_{X\to Y} \to 1 - \frac{\mathbb{E}[\text{Var}(Y|X)]}{\text{Var}(Y)} \tag{7}$$

$$= 1 - \frac{\text{Var}(Y) - \text{Var}(\mathbb{E}[Y|X])}{\text{Var}(Y)} \tag{8}$$

$$= \frac{\text{Var}(\mathbb{E}[Y|X])}{\text{Var}(Y)} \tag{9}$$

$$= \text{Corr}(\mathbb{E}[Y|X], Y)^2, \tag{10}$$

as $N \to \infty$. Eq. 6 measures the quality of a prediction of Y from X without realizing the implied regression, and without making specific assumptions over this regression, i.e. it measures the regression error non-parametrically. A causal inference scheme, as suggested by Bloebaum's condition, is then to infer the direction $X \to Y$, if $C_{X\to Y} > C_{Y\to X}$; and symmetrically for the other direction.

Notice the similarity here with the estimator in [1] (also seen in Eq. (1)), this imply that doing causal inference with (6), inherits the guarantees formulated there. Notice the regression performed in [1] is here implicitly done *non-parametrically*, and as such with fewer structural assumptions. For future reference, we denote the estimator (6) as the *Mean Quadratic Variation (MQV)*.

While we advocate a model-free approach, the above analysis does make *some* assumptions, which should be understood prior to drawing conclusions from data.

- We assume the causal mechanism is *continuous*.
- We assume X has compact and connected support in order to bound the mesh of the partition generated by the sample.
- We assume there exist a unique sorting of x, which is not the case if there are duplicated values. If duplicates are present, MQV can potentially — by statistical anomaly — sort the y values and detect a signal which is not there.

We circumvent with the last two issues by resampling and perturbing the data. Based on the given sample, we estimate the underlying probability distribution $\hat{\mathbb{P}}_{(X,Y)}$, then resample the same amount of data points from this distribution and reevaluate MQV (6). We repeat this procedure several times. This approach both secures unique sorting and has an element of bootstrapping that quantifies the sensitivity to unusual observations. This approach gives empirical distributions of both $C_{X\to Y}$ and $C_{Y\to X}$, and we can then assert probabilities to the event $C_{X\to Y} > C_{Y\to X}$, and its mutual counterpart. Algorithm 1 summarize these ideas, and a practical realization is described in Sec. 3.

2.1 Weak Identifiability

The guarantees by Blöbaum et al. [1] apply to our approach too. We can, however, make some insights into when our approach is *sensible*. First, we consider when it should *not* be relied upon.

Algorithm 1

Input N iid samples of (X, Y).
$\mu \leftarrow$ Estimate the underlying probability distribution of (X, Y).
for i from 1 through m **do**
    $(\tilde{X}, \tilde{Y}) \leftarrow$ Sample N points ($\tilde{x}$,$\tilde{y}$) from μ.
    $Cx_i \leftarrow C_{\tilde{X}\to\tilde{Y}}$; $Cy_i \leftarrow C_{\tilde{Y}\to\tilde{X}}$
$p_x \leftarrow \mu(Cx > Cy)$; $p_y \leftarrow \mu(Cy > Cx)$

Proposition 2.2. *Let $a, b, c, d \in \mathbb{R}$ and $a, c \neq 0$. Assume X and Y are random variables with compact support. Then*

(1) If we have that $\mathbb{E}[Y|X = x] = ax + b$ and $\mathbb{E}[X|Y = y] = cy + d$, then $C_{X\to Y} = C_{Y\to X}$, in the limit of infinite data.

(2) We have $C_{aX+b\to cY+d} = C_{X\to Y}$.

This tells us that when the relationship between X and Y is near linear, we cannot make an informed decision. Note that the use of bootstrapping ensure that both decisions have low confidence, such that the user is at least aware of the lack of identifiability. Although, the linear case is often ideal when considering structural equation models, it is not necessarily simpler in general.

In the noise-free setting, more formal statements can be made.

Proposition 2.3. *If X and Y are random variables with compact support, and there exists measurable f such that $Y = f(X)$, then $C_{X\to Y} \geq C_{Y\to X}$, in the limit of infinite data.*

This directly ties into the definition of causality, since if there is no noise we have that $\mathbb{P}_{Y|X=x} = \mathbb{P}_{Y|\mathrm{do}(x)}$, such that the correct causal decision will be taken. Practically, this indicates, that we should make few incorrect decision in the low-noise regime. Notice that if f is bijective, there exists a function $g = f^{-1}$ such that $X = g(Y)$. Then $C_{X\to Y} = C_{Y\to X}$, such that any taken decision will have low confidence. As before, bootstrapping implies that the user is *aware of this low confidence*.

2.2 Reparametrization Invariance

Principle A informs us that causal decisions should not rely on a specific parametrization of the observations; it is an intrinsic property of the system, rather than a property of the observation space in which we measure. One approach to extracting this intrinsic property is to consider a large number of different parametrizations, in order to be partially invariant to the particular choice of parametrization. Since MQV (6) itself is non-parametric, it is meaningful to evaluate $C_{X\to Y}$ under different parametrizations of the observed data. The simplicity of MQV (6), thus, allow us to realize Principle A. We emphasize that this principle is truly causal, yet most model-based approaches, ANMs in particular, cannot aid its realization. Analogously, parametric approaches to regression-error based causal inference [1] are not, in general, invariant to changes in marginal distributions.

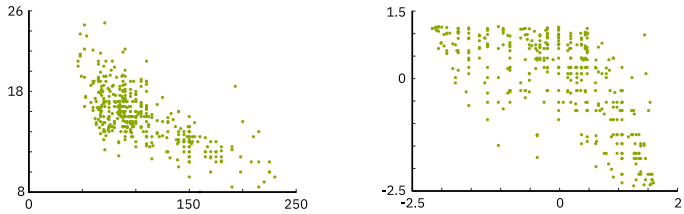


Figure 2: *Left:* a scatterplot of Horsepower (x) vs. Acceleration (y) from the Auto-MPG Dataset (pair0016 from CEP [14]). *Right:* a bijective reparametrizations of the same. The invariance principle states, that the causal decisions taken, should be identical for these two datasets.

We provide a straight-forward realization of the above considerations: define a distribution over bijective reparametrizations f, g, sample $C_{f(X)\to g(Y)}$, and infer a causal direction. We postpone practical implementation details to Sec. 3 and supplementary materials.

Algorithm 2

Input N iid samples of (X, Y). Positive integers M and m.
for j from 1 through M **do**
    Generate random bijection f and g.
    $\mu \leftarrow$ Estimate the underlying probability distribution of $(f(X), g(Y))$.
    for i from 1 through m **do**
        $(\tilde{F}, \tilde{G}) \leftarrow$ Sample N points $(\tilde{f}, \tilde{g})$ from μ.
        $Cx_{ij} \leftarrow C_{\tilde{F}\to\tilde{G}}$; $Cy_{ij} \leftarrow C_{\tilde{G}\to\tilde{F}}$
From samples Cx_{ij} and Cy_{ij} empirically evaluate $p_x = \mathbb{P}(C_X > C_Y)$ and $p_y = 1 - p_x$.

In Fig. 2 we illustrate the invariance principle. Naturally, the marginal distributions of X and Y changes dramatically, and we may think of Algorithm 2 as *integrating out* the marginals. Principle A dictates that the causal link between X and Y is unaltered under these changes; our method then investigates if the estimator (Eq. 6) is too. If this is not the case, we may choose to say that our method cannot estimate a causal relationship. It is clear that any causal inference method based on distributional aspects of the observed is sensitive to these bijections. We empirically investigate this in Sec. 3.4.

2.3 Causal Confidence

The proposed approach can be realized through sampling. This imply that our approach naturally assigns probabilities p_x and p_y to each causal direction. From this, we can near-trivially define a *confidence*, which allow us to rank decisions, as

$$\text{conf}(d) := |p_x(d) - 0.5|. \tag{11}$$

It is a feature of our approach, that the confidence in a decision is an *integral* part of the decision itself. Notice p_x reflect both statistical and model uncertainties, respectively thinking of the bootstrap and reparametrization considerations.

2.4 The Multivariate Generalization

The main focus of this paper is the bivariate case, but the idea neatly generalize to the multivariate setting. So far, we have looked at the variance process (more specifically the *MQV*), but by the polarization identity [12], we can expand to triplets (X, Y, Z) and see that the covariance conditioned on Z is

$$\text{Cov}(X,Y)_Z := \sum_{i=1}^{N-1} w_{i,i+1} \left(\left(s_{i+1} - s_i\right)^2 - (t_{i+1} - t_i)^2 \right), \tag{12}$$

where $s_i = x_i + y_i$ and $t_i = x_i - y_i$ and $\sum_{i=1}^{N-1} w_{i,i+1} = \frac{1}{8}$. Furthermore, the sorting is chosen such that $z_i \leq z_{i+1}$. This expression is symmetric in X and Y, but not in Z; and notice how Eq. 12 in its unaveraged version is exactly the covariance process from stochastic process theory. Hence, we call it the *mean co-quadratic variation*. We state the following Theorem without proof here, as it is analogous to Theorem 2.1.

Theorem 2.4. *Let $(x_i, y_i, z_i)_{i=1,\dots,N}$ be iid samples from $\mathbb{P}_{(X,Y,Z)}$, and assume that Z has compact and connected support $C \subset \mathbb{R}$. Assume further that $\mathbb{E}[X|Z=z]$ and $\mathbb{E}[Y|Z=z]$ are both continuously differentiable over C. Define $s_i = x_i + y_i$ and $t_i = x_i - y_i$ for all $i = 1, \dots, N$. Then*

$$\frac{1}{8(N-1)} \sum_{i=1}^{N-1} \left(\left(s_{i+1} - s_i\right)^2 - (t_{i+1} - t_i)^2 \right), \tag{13}$$

tends to $\mathbb{E}\text{Cov}(X, Y|Z)$ as $N \to \infty$.

By the law of total covariance, we have

$$\text{Cov}(X,Y) - \text{Cov}(\mathbb{E}[X|Z], \mathbb{E}[Y|Z]) = \mathbb{E}[\text{Cov}(X,Y|Z)], \tag{14}$$

implying that if Eq. 12 is close to zero, then most of the covariation between X and Y can be explained by Z. This indicates that X and Y might be independent given Z. Of course, this is generally not a sufficient condition, but it is necessary. The following statement gives sufficient conditions [15].

Theorem 2.5. *Two random variables X and Y are independent if and only if*

$$Cov(f(X), g(Y)) = 0, \tag{15}$$

for any pair of functions f and g that are bounded and continuous.

This sufficient condition allows for a simple conditional independence test: transform the observed values with bounded continuous functions and check if Eq. 12 is zero for any such transformation. This naive test is exactly how we algorithmically realize Principle A, which illustrates that our non-parametric estimators aligns with the fundamental ideas from Pearl [2] and DAG estimation.

3 Experiments

We now evaluate the empirical behavior of the proposed model-free approach. We consider ANM and IGCI as baseline methods, and report results first on simulated data, and then on the real-world CEP benchmark dataset [14]. For all comparisons below, when we state IGCI, we mean the slope-based estimator with uniform reference measure. For ANM, we applied the GP regression and the Hilbert-Schmidt Independence Criterion [16]. Implementations are from the publicly available code given by Mooij et al. [14]. We rank our decisions based on the confidence score in Eq. 11, while ANM and IGCI come with their own confidence scores [14]. Algorithmic details of our estimator are available in the supplementary material alongside the associated source code.

We shall also compare to *Regression Error-based Causal Inference* (RECI) from Blöbaum et al. [1], as their approach is highly similar to the one presented here. This imply we have to choose a method of regression, and we regress by using the logistic-function class. This decision was made based on what we found were the overall best performance in their paper.

It was a motivation for us to present a non-parametric way of doing (implicit) regression, that would alleviate the need to pick a fair regression method for both causal and anti-causal direction. Further, the non-parametricity in our estimator is essential to apply bijections meaningfully.

Lastly, we performed a small experiment on the sensitivity of how the reparametrizations were sampled. The outline of this experiments was that as long as the bijections were *diverse* enough, there is little sensitivity to the choice of distribution.

3.1 Simulated Pairs

The data considered here is 100 pairs, each consisting of 1000 observations, simulated according to the procedure introduced by Mooij et al. [14]; trying to mimic real-world data. There are four setups: the general simulated data (SIM), the data generated with low noise to the effect (SIM-ln), the data with one confounder present (SIM-c), and finally the data where the cause is Gaussian and the additive noise is too (SIM-G).

As we have observed, we would expect our method to perform well at least on the low-noise data, as one would too for IGCI. The results for all datasets are visualized in Fig. 3. This figure should be read from right to left as taking all decisions, we then sequentially discard the decisions we are most uncertain about. The 10 blue lines are outputs from Algorithm 2, indicating the inherent randomness in the decision-making. We see that our method outperforms IGCI in most cases, and is comparable to ANM. Equivalently, the cyan lines are the outputs from Algorithm 1 - that is, without bijections.

In all experiments, we note that our choice of ranking (11) prefers easier decisions, which is evident from the *concave* shape of the result curve. Note that uncertainty is larger for decisions that are considered difficult (low confidence), but the performance on high confidence decisions is generally better than both ANM and IGCI. From Fig. 3 it is visible that MQV reports: '*I don't know*' when one blue line turns into many. At this crucial point MQV is consistently as good or better than ANM.

Interestingly, if we take decisions with ANM in the same order as MQV, we obtain more preferable concave curves; in fact such that the performance resembles MQV on

	SIM	SIM-c	SIM-ln	SIM-G
RECI	64.1 ± 1.4	63.2 ± 2.4	$\mathbf{83.1 \pm 2.3}$	$\mathbf{74.9 \pm 3.9}$
MQV (w/o bijections)	62.2 ± 0.9	63.4 ± 0.1	$\mathbf{82.7 \pm 0.8}$	$\mathbf{73.5 \pm 0.8}$
MGV (w/ bijections)	$\mathbf{68.8 \pm 2.1}$	$\mathbf{65.9 \pm 2.5}$	$\mathbf{82.0 \pm 3.5}$	61.5 ± 2.6

Table 1: Average number of correct decisions and standard deviations over 10 runs. Bold marks statistically significantly best method. Only regression-error based methods are listed, as ANM and IGCI do not have error-bars; their performance can be read from Figure 3.

high confidence situations. We investigated this on all 4 datasets, and the concavity is visualized in Fig. 4. Here all decision are taken as determined by ANM, but ordered wrt. Eq. 11; the lines indicate the difference to the black lines of Fig. 3. Hence the lines are bound to go through $(0, 0)$ and $(100, 0)$, and in between any positive number imply an improvement over ANM's own ranking. In particular, we note that for decisions where our estimator is certain, we generally improve upon ANM's ranking.

The overall performance of RECI is shown in Table 1. We see that, unsurprisingly, this is *very* similar to our approach without bijections. Another key observation from Figure 3 is that on the most 'nature-like' datasets, SIM and SIM-c, taking observations into account improves overall decision making. On SIM-G we conjecture there might be a bias in our estimator (without bijections), why it therefore might still be more 'safe' to include bijections. This conjecture is based on that the Gaussian is the maximum entropy distribution when mean and variance is known (standardized in our case), which might make the regression error of $Y \to$ X tend to be larger if X is Gaussian, than if it was not. This sort of bias is alleviated by marginalizing marginal distributions (with bijections).

3.2 Real-World Data

The CAUSEEFFECTPAIRS (CEP) database[2] is currently 108 datasets, of which 103 are bivariate. It consists of real-world observations annotated with a causal direction [14]. As such, the 103 pairs are not independent, as several originate from the same datasets, and to make up for this each pair has an associated weight. Our results on this dataset are plotted in Fig. 5 and we see that we are comparable to other known methods when we *integrate out* random bijections. The blue dots in the figure are our results for respectively Algorithm 1 and 2 for 10 runs. Most runs for Algorithm 2 yielded accuracies in the range $0.62 - 0.64$, with one run having accuracy 0.66 and two around 0.59. We see that Algorithm 2 is comparable to other known methods, while Algorithm 1 is subpar ($^7/_{10}$ runs had accuracy in $0.58 - 0.61$). ANM yields an accuracy of 0.63, and for IGCI 0.64. Most importantly, this illustrates that Principle A is not hollow talk, since marginalizing bijections seem to significantly improve performance.

Over 10 runs the RECI method provide accuracies on the range of $[0.46, 0.62]$, averaging at 0.53. This is worse that our approach, even without bijections, and not significantly better than random guessing.

[2]https://webdav.tuebingen.mpg.de/cause-effect/ as it appeared in December 2019.

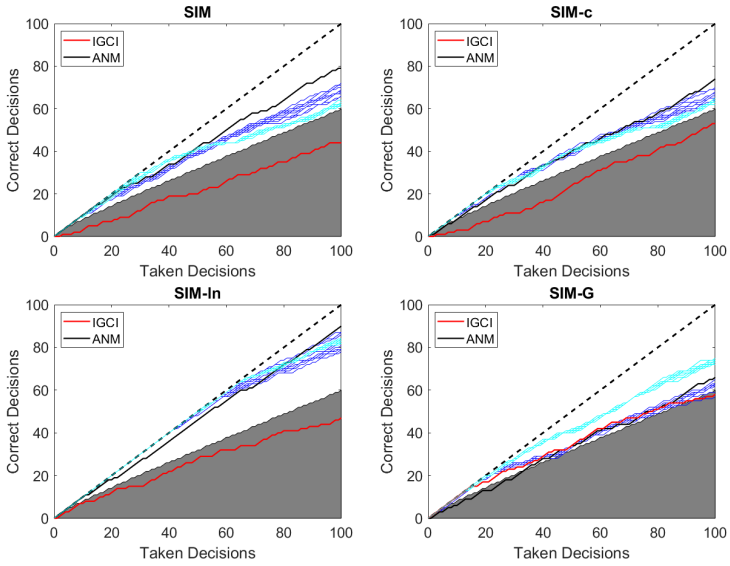


Figure 3: For the four different synthetic datasets, the blue lines (MQV with bijections) are the proportion of correct decisions where the decisions have been ranked according to the heuristic (11). The cyan lines are MQV without bijections. ANM and IGCI have other confidence scores [14]. The shaded area is what falls below the 0.975 quantile of a binomial distribution with $p = 0.5$.

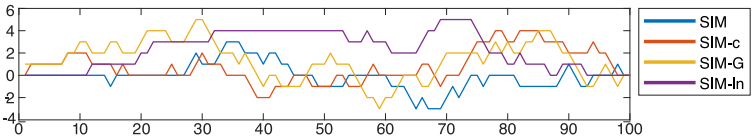


Figure 4: Illustration of changes in concavity for ANM under changes in confidence score. Any curve above the constant line 0 imply more concave decision curve, hence the ranking we propose is better than the original [14]. Especially, concavity in the left-most side of the plot is important, as this reflect the most confident decisions.

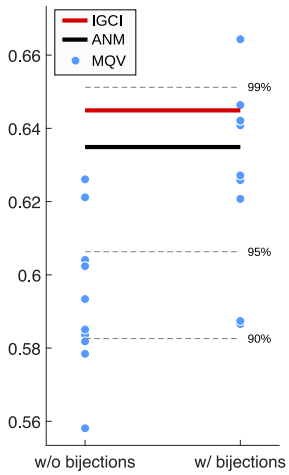


Figure 5: Performance on the CEP-benchmark. The blue dots (MQV) illustrate the inherent randomness in the algorithms. The dashed grey lines are quantiles had we tossed a fair coin for each decision. The average performance of RECI is 0.53; below the plotted window.

3.3 The Multivariate Generalization

We empirically illustrate the generalization to triplets (X, Y, Z) on data generated similar to that in Sec. 3.1. We generate 100 DAGs of the type $X \leftarrow Z \rightarrow Y$, and a further 100 DAGs similar but with an added edge, either $X \rightarrow Y$ or $Y \rightarrow X$. For clarity, this means that we should find 100 times that $X \perp\!\!\!\perp Y|Z$ and 100 times not. We construct a test that transform the variables X and Y with bijections, and test whether any such transformation make their absolute conditional covariance $\mathrm{Cov}(f(X), g(Y))_Z$ (calculated as Eq. (12)) exceed some threshold. In the following we have set this threshold to 0.15, and rejected the hypothesis of independence if more than 1% of the samples go above this.

The results in Table 2 exemplify that (co-)quadratic variation is not misplaced in the causal framework, knowing that causality and conditional independence testing have been closely related for decades [2]. We further notice that Theorem 2.4 assumes Z to be one-dimensional, but this extend to higher dimensions if one just finds a *sorting* in this space. Keep in mind that, if the mesh tends to zero, then the convergence from above is still assured. For practical considerations one would then find a permutation of z_i, $i = 1, \ldots, N$ such that $\max_{i=1,\ldots,N-1} \|z_{i+1} - z_i\|_2$ is as small as possible. This is a non-trivial problem for higher dimensions than 1. Suggestions here could be to use some kernel methods [17], or some ranking on data manifolds [18].

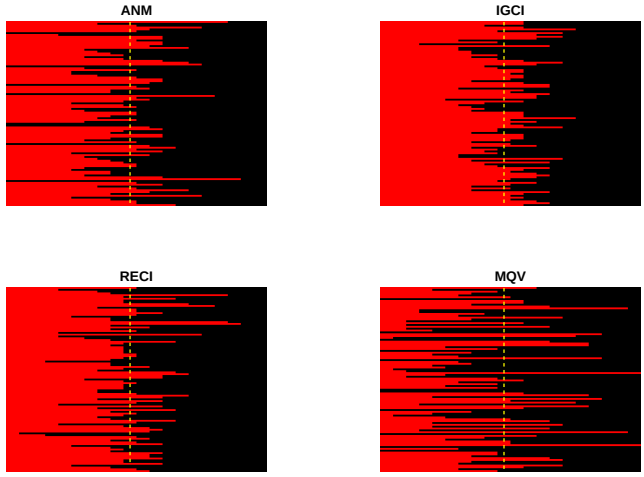


Figure 6: On the 100 pairs (rows) from SIM, we applied 20 random bijections (columns). Above illustrates how the bijections influenced the decision. Red is an incorrect decision. Table 3 quantitatively summarize the plots of this figure. We observe that MQV has 'fuller' bars, indicating that decisions are less influenced by bijections.

	Positives	Negatives	Total
True	99	1	100
False	3	97	100

Table 2: Conditional independence results on synthetic data.

3.4 Robustness

We evaluate how restrictive model assumptions are for the invariance principle; more specifically we measure how robust the different causal inference methods are if we transform X and Y with bijections. This gives an indication of which methods align with Principle A. Fig. 6 has in each row one pair from the dataset SIM. To each of these pairs, we applied 20 random bijections and kept track of the decisions made. Black and red are respectively incorrect and correct decisions. We can see that MQV is more robust to Principle A, as we have more *full bars* (or near full) along the rows, implying the decisions are less likely to be altered by the bijections. In fairness it should be stated that the bijections are not identical in-between plots. We may quantify this sensitivity with the entropy, i.e. for each pair evaluate $-d_1 \log d_1 - d_2 \log d_2$, where d_1 and d_2 are the fraction of times decision $X \to Y$ and $Y \to X$ were made. In Table 3 the average entropy of causal decisions over all pairs in a data set is listed, which indicates how robust a method is to bijections: small entropy imply robustness.

Table 3 provide evidence to the hypothesis that assuming a model is not robust under random bijections. Our method deals better with this. One naturally also observes a

	SIM	SIM-c	SIM-ln	SIM-G	CEP
ANM	0.5953	0.5838	0.5457	0.5748	0.5591
IGCI	0.6620	0.6718	0.6692	0.6655	**0.1247**
RECI	0.6097	0.6092	0.5902	0.6066	0.6416
MQV	**0.4895**	**0.4461**	**0.4906**	**0.4376**	0.4381

Table 3: The mean entropy of decisions under random bijections.

clear deviant in Table 3, IGCI-decisions on CEP are nearly closed under bijections, and there must be some entity in the data explaining this. Following up on this, we introduce a *strawman estimator*

$$S_{X\to Y} := \frac{\#\text{ of unique values in } X}{\#\text{ of unique values in } Y}, \tag{16}$$

and infer $X \to Y$ if $S_{X\to Y} < 1$. Evidently this measure is invariant if we biject X and Y, but its relation to causal decision taking is not evident. On CEP this procedure takes the same decision as IGCI on 98 out of 103 pairs, and the strawman estimator alone has an accuracy around $0.57-0.61$ (in 3 cases $S_{X\to Y} = S_{Y\to X}$, and we flip a coin). Thus, we conjecture that the success IGCI has had on the CEP-Benchmark is a spurious correlation due to duplicated values in the data. This is supported by the fact that IGCI discard duplicated values.

4 Discussion and conclusion

We took a novel approach to bivariate causal discovery, by imposing invariance on the causal domain rather than distributional assumptions. We did this by quantifying the regression errors in a non-parametric manner, which allowed for us to meaningfully take advantage of the proposed invariant principle (Principle A). We provide a thorough empirical analysis on the impact of this principle.

The results show that this approach is feasible and is competitive with the current methodologies, that impose structural model assumptions. We find both the theoretical and computational ease of the approach highly appealing. However, we do not consider the present work complete, and we hope that future work in the field will take into account that if causal models are closed under reparametrizations, then so should its estimators. The results show that the non-parametricity of the mean quadratic variation (MQV) is more robust under reparametrizations, and that taking this into account significantly improves performance. This insight also proposed an explanation for the good performance of IGCI [7] on real-world observations to which there has been previous speculation [7, 14].

Further, we have demonstrated that MQV extends to higher dimensions in ways that are similar to the traditional conditional independence tests used for estimating DAGs [2].

Finally, the presented method pays high attention to the uncertainty of any causal estimation, which results in a confidence measure that outperforms the baselines and shows good promise of detecting when it seems feasible to do causal inference with

purely observational data; a query which is much more fundamental than the inference itself.

Acknowledgements

This project has received funding from the European Research Council (ERC) under the European Union's Horizon 2020 research and innovation programme (grant agreement no 757360). MJ and SH were supported in part by a research grant (15334) from VILLUM FONDEN.

A Experimental details

For the experimental setup in the paper, we here give the explicit and detailed description. See Algorithm 1 and 2 for notational help.

For our own method, we consistently use $m = 300$ and $M = 100$ (see Algorithm 2), meaning we generate 100 random bijections for each pair, and for each of these we subsample 300 times, c.f. Sec. 2. We estimate the underlying probability distribution by Gaussian kernel density estimation, with Silverman's rule of thumb for bandwidth [19]. This is a crude estimator for many pairs, but we leave it to future work to optimize this procedure of the algorithm; and to fairly compare on all pairs we choose it throughout.

From a practical perspective we note that by Proposition 2.2 we may restrict this search to strictly increasing functions.

Generation random increasing functions was done with the following setup: draw γ from an inverse Gamma distribution with both shape and scale parameters set to 5. Generate a Gaussian Process (GP) f with zero mean and covariance function $k(x, x') = \exp(-\frac{1}{2\gamma}\|x - x'\|_2^2)$. Then let $f(x_0) := \min_{x \in \text{supp}(X)} f(x)$ and set

$$F(x) := f(x_0) + \int_X (f(x) - f(x_0))\mathrm{d}x, \tag{17}$$

then F is an increasing function.

Based on a sample (C_X, C_Y), we estimate

$$p_x = \frac{1}{(mM)^2} \sum_{i=1}^{mM} \sum_{j=1}^{mM} \mathbf{1}\{c_{y_j} < c_{x_i}\}. \tag{18}$$

We note that, since the GP has zero mean, its integral (17) has a linear mean function.

We introduce a *confidence* in each decision, and this heuristic is near trivial when both algorithms return a probability p_x (we set $p_y = 1 - p_x$). Thus we define confidence of a decision d as

$$\text{conf}(d) := |p_x(d) - 0.5|. \tag{19}$$

We rank our decisions based on this: the higher the confidence, the more we believe in our decision. ANM and IGCI have other confidence scores [14].

When we state IGCI, we mean the slope-based estimator with uniform reference measure. For ANM, we applied the GP regression and the Hilbert-Schmidt Independence

Criterion [16]. Implementations are from the publicly available code given by Mooij et al. [14].

B Proof of Theorems

Theorem 2.1. *Let X have support on a compact and connected subset C of $\mathbb{R}$, and assume that $\mathbb{E}[Y|X = x]$ is a continuous differentiable function over C. Assume $\mathbb{E}Y^2 < \infty$. Let further (x_i, y_i), $i = 1, \ldots, N$, be iid samples from $\mathbb{P}_{(X,Y)}$. If we order, such that $x_{i+1} \geq x_i$ for all $i = 1, \ldots, N-1$, then it holds that*

$$\frac{1}{N-1}\sum_{i=1}^{N-1}(y_{i+1} - y_i)^2 \to 2\mathbb{E}\text{Var}(Y|X), \tag{20}$$

as $N \to \infty$.

Proof. Let $f(x) := \mathbb{E}[Y|X = x]$, and decompose for all i

$$y_i = f(x_i) + \big(y_i - f(x_i)\big) =: f(x_i) + \epsilon_i. \tag{21}$$

Then we see that

$$\begin{aligned}\sum_{i=1}^{N-1}\big(y_{i+1} - y_i\big)^2 = &\sum_{i=1}^{N-1}\big(f(x_{i+1}) - f(x_i)\big)^2 \\ &+ \sum_{i=1}^{N-1}\big(\epsilon_{i+1} - \epsilon_i\big)^2 \\ &+ 2\sum_{i=1}^{N-1}\big(\epsilon_{i+1} - \epsilon_i\big)\big(f(x_{i+1}) - f(x_i)\big),\end{aligned} \tag{22}$$

where the first and last terms tend to zero when scaled with $1/(N-1)$ due to Lemma B.1 (below) and the Cauchy-Schwartz inequality. Thus we are left with

$$\sum_{i=1}^{N-1}\big(\epsilon_{i+1} - \epsilon_i\big)^2 = \sum_{i=1}^{N-1}\epsilon_i^2 + \sum_{i=2}^{N}\epsilon_i^2 - 2\sum_{i=1}^{N-1}\epsilon_i\epsilon_{i+1}, \tag{23}$$

and the last term vanishes due to the iid assumption[3] and the fact that $\mathbb{E}\epsilon_i = 0$ for all i. Hence, as $N \to \infty$,

$$\begin{aligned}&\frac{1}{N-1}\sum_{i=1}^{N-1}\Big(y_i^2 + f(x_i)^2 - 2y_i f(x_i)\Big) \\ &\quad\to \text{Var}(Y) + \text{Var}(\mathbb{E}[Y|X]) - 2\text{Cov}(Y, \mathbb{E}[Y|X]) \\ &\quad= \text{Var}(Y) - \text{Var}(\mathbb{E}[Y|X]) \\ &\quad= \mathbb{E}[\text{Var}(Y|X)],\end{aligned}$$

by the law of total variance[4]. □

[3] Recall a sequence of iid variables, is still iid under any permutation.

[4] We assumed without loss of generality that $\mathbb{E}Y = 0$.

Proposition 2.2. *Let $a, b, c, d \in \mathbb{R}$ and $a, c \neq 0$. Assume X and Y are random variables with compact support. Then*

(1) If we have that $\mathbb{E}[Y|X = x] = ax + b$ and $\mathbb{E}[X|Y = y] = cy + d$, then $C_{X\to Y} = C_{Y\to X}$, in the limit of infinite data.

(2) We have $C_{aX+b\to cY+d} = C_{X\to Y}$.

Proof. Ad (1):

$$C_{X\to Y} \to \text{Corr}(\mathbb{E}[Y|X], Y)^2 = \text{Corr}(aX + b, Y)^2 = \text{Corr}(X, Y)^2,$$

and completely analogous for $C_{Y\to X}$.

Ad (2): Clearly $\{x_i\}_{i=1,\dots,N}$ and $\{ax_i + b\}_{i=1,\dots,N}$ have the same sorting when $a \neq 0$. $C_{X\to Y}$ is obviously invariant to scaling and translating in Y, since we standardize the variable. □

Proposition 2.3. *If X and Y are random variables with compact support, and there exists measurable f such that $Y = f(X)$, then $C_{X\to Y} \geq C_{Y\to X}$, in the limit of infinite data.*

Proof. If there is no noise, then Lemma B.1 suggests that $C_{X\to Y} \to 1$, which concludes the assertion in the limit. □

Lemma B.1. *Let X be a random variable with support on a compact and connected set $C \subset \mathbb{R}$ and let $f : \mathbb{R} \to \mathbb{R}$ be a continuously differentiable function over C. Let x_i be independent samples of X for $i = 1, \dots, N$. Then*

$$\frac{1}{N-1} \sum_{i=1}^{N-1} \Big(f(x_{i+1}) - f(x_i)\Big)^2 \to 0 \quad \text{as } N \to \infty, \tag{24}$$

where $x_{i+1} \geq x_i$ for all i.

Proof. For notation, we use $x_{(1)} \leq x_{(2)} \leq \dots \leq x_{(N)}$ for the sorted sample. We denote

$$K_N = \frac{1}{N-1} \sum_{i=1}^{N-1} \Big(f(x_{(i+1)}) - f(x_{(i)})\Big)^2.$$

Since f is continuously differentiable, there exists $M := \sup_{x\in C} f'(x)$, and by compactness there exists $a, b \in \mathbb{R}$, $a \leq b$, such that $C = [a, b]$. Then the bound, for any $N \geq 2$

$$S_N := M^2 \left((x_{(1)} - a)^2 + (b - x_{(N)})^2 + \sum_{i=1}^{N-1} \Big(x_{(i+1)} - x_{(i)}\Big)^2 \right) \geq K_N. \tag{25}$$

Hence it suffices to show that for any $\epsilon > 0$ there exists N_0, such that for all $N > N_0$, we have $S_N < \epsilon$. Naturally S_N is downwards bounded by 0, thus we may show that S_N is a strictly descending sequence. See that for any fixed N we have that $x_{N+1} \in [a, b]$,

either $x_{N+1} \in [a, x_{(1)})$, $x_{N+1} \in [x_{(N)}, b]$ or there exists some $j = 1, \ldots, N-1$ such that $x_{N+1} \in [x_{(j)}, x_{(j+1)})$. For the last case it holds that

$$(x_{(j+1)} - x_{(j)})^2 \geq (x_{(j+1)} - x_{N+1})^2 + (x_{N+1} - x_{(j)})^2,$$

and the cases $a \leq x_{N+1} < x_{(1)}$ and $x_{(N)} \leq x_{N+1} \leq b$ follows analogously. This shows that $S_N > S_{N+1}$. Now scale S_N with $\frac{1}{N-1}$ and observe that (25) still holds, hence $0 \leq K_N \leq \frac{S_N}{N-1} \leq \frac{S_2}{N-1} \to 0$, and the assertion follows. □

References

[1] Patrick Blöbaum, Dominik Janzing, Takashi Washio, Shohei Shimizu, and Bernhard Schölkopf. Cause-effect inference by comparing regression errors. In *International Conference on Artificial Intelligence and Statistics*, pages 900–909, 2018.

[2] Judea Pearl. *Causality: models, reasoning, and inference*. Cambridge University Press, 2009.

[3] Patrik O. Hoyer, Dominik Janzing, Joris M Mooij, Jonas Peters, and Bernhard Schölkopf. Nonlinear causal discovery with additive noise models. In D. Koller, D. Schuurmans, Y. Bengio, and L. Bottou, editors, *Advances in Neural Information Processing Systems 21*, pages 689–696. Curran Associates, Inc., 2009.

[4] Shohei Shimizu, Patrik O. Hoyer, Aapo Hyvärinen, and Antti Kerminen. A linear non-gaussian acyclic model for causal discovery. *Journal of Machine Learning Research*, 7:2003–2030, 2006.

[5] Kun Zhang and Aapo Hyvärinen. Distinguishing causes from effects using nonlinear acyclic causal models. In Isabelle Guyon, Dominik Janzing, and Bernhard Schölkopf, editors, *Proceedings of Workshop on Causality: Objectives and Assessment at NIPS 2008*, volume 6 of *Proceedings of Machine Learning Research*, pages 157–164, Whistler, Canada, 12 Dec 2010. PMLR.

[6] Eleni Sgouritsa, Dominik Janzing, Philipp Hennig, and Bernhard Schölkopf. Inference of cause and effect with unsupervised inverse regression. In *Proceedings of the 18th International Conference on Artificial Intelligence and Statistics (AISTATS)*, volume 38, pages 847–855, 2015.

[7] Dominik Janzing, Joris Mooij, Kun Zhang, Jan Lemeire, Jakob Zscheischler, Povilas Daniusis, Bastian Steudel, and Bernhard Schölkopf. Information-geometric approach to inferring causal directions. *Artificial Intelligence*, (182-183):1–31, 2012.

[8] Joris Mooij, Oliver Stegle, Dominik Janzing, Kun Zhang, and Bernhard Schölkopf. Probabilistic latent variable models for distinguishing between cause and effect. In *Advances in Neural Information Processing Systems*, volume 23, pages 1687–1695, 01 2010.

[9] Neil Lawrence. Probabilistic non-linear principal component analysis with gaussian process latent variable models. *Journal of machine learning research*, 6(Nov): 1783–1816, 2005.

[10] Jonas Peters, Dominik Janzing, and Bernhard Schölkopf. *Elements of Causal Inference*. MIT Press, 2017.

[11] P Daniusis, D Janzing, J Mooij, J Zscheischler, B Steudel, K Zhang, and B Schölkopf. Inferring deterministic causal relations. In *26th Conference on Uncertainty in Artificial Intelligence (UAI 2010)*, pages 143–150. AUAI Press, 2010.

[12] Richard Durrett. *Stochastic Calculus: A Practical Introduction*. CRC Press, 1996.

[13] Peter Hall, J. W. Kay, and D. M. Titterington. Asymptotically optimal difference-based estimation of variance in nonparametric regression. *Biometrika*, 77(3): 521–528, 1990.

[14] Joris M. Mooij, Jonas Peters, Dominik Janzing, Jakob Zscheischler, and Bernhard Schölkopf. Distinguishing cause from effect using observational data: Methods and benchmarks. *Journal of Machine Learning Research*, (17):1–102, 2016.

[15] Jean Jacod and Philip Protter. *Probability Essentials*. Springer, New York, 2000.

[16] Arthur Gretton, Olivier Bousquet, Alex Smola, and Bernhard Schölkopf. Measuring statistical dependence with hilbert-schmidt norms. *Algorithmic Learning Theory*, pages 63–78, 2005.

[17] Peter Hall and J. S. Marron. On variance estimation in nonparametric regression. *Biometrika*, 77(2):415–419, 1990.

[18] Dengyong Zhou, Jason Weston, Arthur Gretton, Olivier Bousquet, and Bernhard Schölkopf. Ranking on data manifolds. In S. Thrun, L. K. Saul, and B. Schölkopf, editors, *Advances in Neural Information Processing Systems 16*, pages 169–176. MIT Press, 2004.

[19] Bernard Walter Silverman. *Density Estimation for Statistics and Data Analysis*. London: Chapman & Hall/CRC, 1986.