Latent Variable Augmentation for Approximate Bayesian Inference

Applications for Gaussian Processes

vorgelegt von
Msc. Physik
Théo Galy-Fajou
geb. in Castres, France
ORCID: 0000-0002-3528-3536

an der Fakultät IV - Elektrotechnik und Informatik
der Technischen Universität Berlin

zur Erlangung des akademischen Grades

Doktor der Naturwissenschaften
-Dr. rer. nat.-

genehmigte Dissertation

Promotionsausschuss:

Vorsitzender: Prof. Dr. Marc Toussaint
Gutachter: Prof. Dr. Manfred Opper
Gutachter: Dr. Mark van der Wilk
Gutachter: Dr. Arno Solin
Tag der wissenschaftlichen Aussprache: 07. Juli 2022

Berlin 2022

Zusammenfassung

Die Inferenz auf probabilistische Modelle kann selbst bei scheinbar einfachen Problemen eine Herausforderung darstellen. Bei der Arbeit mit nicht-konjugierten Bayes'schen Modellen benötigen wir Näherungsmethoden wie Variationsinferenz oder Sampling, die jeweils ihre Tücken und Grenzen haben. So stellen beispielsweise stark schwanzlastige Verteilungen eine Herausforderung für Sampling-Methoden dar, und stark korrelierte Variablen werden für viele Inferenzalgorithmen schnell zu einem Engpass. Anstatt einen weiteren hochmodernen Sampler oder Optimierer zu entwickeln, konzentrieren wir uns darauf, Modelle so umzuinterpretieren, dass Standard-Inferenzalgorithmen wie blockiertes Gibbs-Sampling, die normalerweise auf trivialere Modelle beschränkt sind, die beste Wahl werden. Im ersten Teil leiten wir Modellerweiterungen für verschiedene Gauß'sche Prozessmodelle wie Klassifikation und Mehrklassenklassifikation ab. Wir konzentrieren uns auf die Auswirkungen auf die Inferenz und entwickeln eine Verallgemeinerung für eine bestimmte Klasse von Likelihoods. Wir zeigen, dass die Augmentierungen mit den Daten skalierbar sind und alle bestehenden Methoden in Bezug auf Geschwindigkeit und Stabilität übertreffen. Der zweite Teil konzentriert sich auf Approximationen, die auf einer Gaußschen Variationsverteilung basieren. Wir zeigen, dass wir durch die Parametrisierung der Gauß-Verteilung durch eine Menge von Partikeln anstelle ihrer Parameter teure Berechnungen vermeiden, die Flexibilität des Modells erhöhen und theoretische Konvergenzgrenzen nachweisen können. Zusätzlich zu den veröffentlichten Arbeiten diskutieren wir die Auswirkungen dieser verschiedenen Erweiterungen, einschließlich ihrer Grenzen. Wir geben auch einen Ausblick auf neue Forschungsrichtungen, einschließlich konkreter Fortschritte. Insbesondere zeigen wir Wege auf, wie die in den vorgestellten Arbeiten aufgeworfenen Probleme kompensiert werden können, und stellen neue Augmentationsmodelle und neue Inferenzansätze vor, die mit augmentierten Modellen kompatibel sind.

Abstract

Performing inference on probabilistic models can represent a challenge even in seemingly simple problems. When working with non-conjugate Bayesian models, we need approximate methods such as variational inference or sampling, each with its pitfalls and limits. For instance, heavy-tailed distributions represent a challenge for sampling methods, and strongly correlated variables quickly become a bottleneck for many inference algorithms. Instead of developing yet another new state-of-the-art sampler or optimizer, we focus on reinterpreting models such that standard inference algorithms like blocked Gibbs sampling, usually restricted to more trivial models, become the best choice. In the first part, we derive model augmentations for different Gaussian Process models such as classification and multi-class classification. We focus on the effects on inference and develop a generalization for a given class of likelihoods. We show that augmentations are scalable with data and outperform all existing methods in terms of speed and stability. The second part focuses on approximations based on a Gaussian variational distribution. We show that by parametrizing the Gaussian distribution by a set of particles instead of its parameters, we avoid expensive computations, increase the model flexibility, and prove theoretical convergence bounds. In addition to the published papers, we discuss the impact of these different augmentations, including their limitations. We also expose outlooks on new research directions, including concrete advances. In particular, we present ways to compensate for issues raised in the presented papers and present new augmentation models and new inference approaches compatible with augmented models.

Dedié à Manou.

Acknowledgements

I would like to thank Ena for her unconditional love and support since the beginning, and especially her help to not lose myself into work.
Professor Opper for sharing his immense wisdom and knowledge, bearing with by stubbornness and for believing in me.
My parents for supporting me in everything I have ever started.
"Les filous", for keeping me entertained at all times.
My main co-author and tutor Florian who taught me so much before and during my Ph.D.
The Julia community, from whom I learned so much and for their indeflectible help during hard programming times.
And of course all the people I shared lunch and good times with at the university.

Table of Contents

Acronyms

$\mathcal{GP}$ Gaussian Process

$\mathcal{GP}$**s** Gaussian Processes

MCMC Markov Chain Monte Carlo

VI Variational Inference

VFE Variational Free Energy

ELBO Evidence Lower BOund

KL Kullback-Leibler

MF Mean-Field

BMF Blocked Mean-Field

CAVI Coordinate Ascent Variational Inference

HMC Hamiltonian Monte Carlo

MH Metropolis-Hastings

ML Machine Learning

VGA Variational Gaussian Approximation

MGF Moment Generating Function

pdf probability distribution function

iid independent and identically distributed

NUTS No-U-turn sampling

ABI Approximate Bayesian Inference

1 Introduction

Machine Learning (ML) is a wide field of research with plenty of successful applications [29]. Some problems have specific requirements; for example, computing the probability of a prediction is essential for decision-making algorithms. One of the best ways to incorporate uncertainty in ML models is through the lens of probability theory. Probabilistic ML defines quantities of interest as random variables and considers data-generative processes as stochastic. We can produce more robust models and get more faithful to reality by accounting for the intrinsic measurement uncertainty and unknown random processes. Additionally, stochastic models return probabilistic predictions, allowing answers like "I don't know."

1.1 Bayesian Machine Learning

In the Bayesian paradigm, parameters of ML models are **random variables** defined by **probability distributions** instead of point estimates. Bayesian models allow modeling uncertainty in a principled way and prevent overfitting in the low-data regime. We set a **prior distribution** over the variables of interest representing our original belief. After observing data, we update our belief about our model parameters to the **posterior distribution**. A typical example is in medicine, where data is scarce, but the predictive outcome can have dramatic effects (diagnosis, prognosis). Providing uncertainties helps the practitioner make a better decision given the model predictions.

Generally, Bayesian models have a higher computational cost: a probability distribution contains more information than a point estimate and requires more parameters. Calculus with random variables is a difficult art, and finding analytical solutions happens almost exclusively for trivial models. Approximation methods allow working with more complex models at the cost of a potential bias or inaccuracies. **Approximate Bayesian Inference (ABI)** focuses on these algorithms finding a similar solution to the true posterior.

The research in ABI goes in many directions, but some main ones are: How to compute a highly accurate posterior approximation as efficiently as possible? How can it scale to large amounts of data and parameters? What are the guarantees of such algorithms? This thesis

aims to partially answer these questions for some given setups, mainly through a focus on representations.

1.2 The underestimated power of representations choices

The leading thread of this thesis is **model representation**, alternatively called **model parameterization**, and its use for solving problems more efficiently and faster without compromising prediction quality.

When defining probabilistic models, one needs to define relations between variables (observed and latent) and choose appropriate distributions to represent those. Some modeling choices are equivalent conceptually but have drastic differences in inference. A neat example, presented in Gorinova et al. [18], is the so-called Neal's funnel [39]. There are two equivalent representations, called centered and non-centered, shown respectively in Figure 1.1 and 1.2, where one leads to an inference nightmare while the other is a nice and easy isotropic Gaussian distribution.

$$\begin{aligned} z &\sim \mathcal{N}(0, 3) \\ x &\sim \mathcal{N}(0, \exp(z/2)) \end{aligned} \tag{1.1}$$

$$\begin{aligned} \tilde{z} &\sim \mathcal{N}(0, 1), \quad z = 3\tilde{z} \\ \tilde{x} &\sim \mathcal{N}(0, 1), \quad x = \exp(z/2)\tilde{x} \end{aligned} \tag{1.2}$$

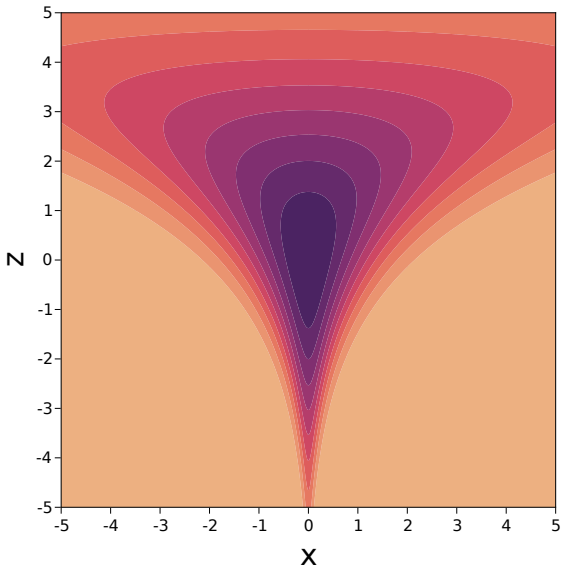


Figure 1.1: Neal's funnel - Centered representation

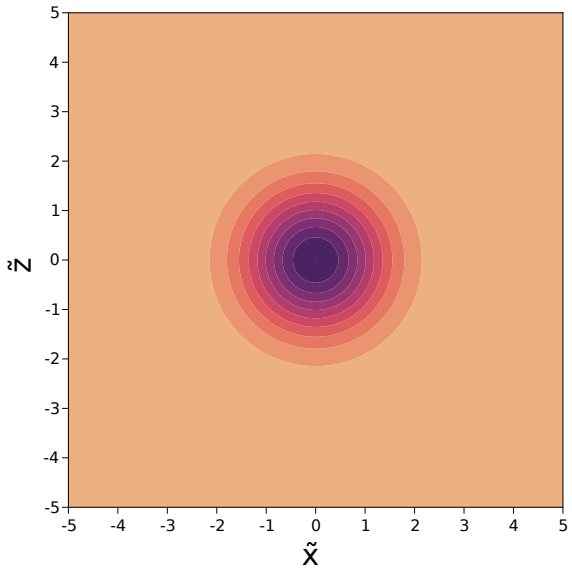


Figure 1.2: Neal's funnel - Non-centered representation

While both parameterizations are the same, the distribution geometry of $p(x, z)$ is less favorable to inference. x and z are strongly correlated for small z, and the density function is highly non-smooth. These constraints matter when running a sampling chain or fitting a variational distribution.

The use of different model representations has an often underestimated effect and is mainly considered "tricks." For example, when working with Gaussian Processess, it is generally preferable to use the so-called "whitened" representation, which corresponds to the non-centered representation of Neal's funnel (Figure 1.2). The different segments of this thesis show that finding better representations can confidently make inference easier, faster, and significantly

more stable. The first part will use basic inference methods by representing likelihoods as (hierarchical) mixtures. Rewriting distributions as scale mixtures, defined later in Section 2.3.3, has a lot of advantages and interesting properties. The scale mixture representation involves augmenting the model with new latent variables, making inference easier while keeping the original model recoverable. This augmentation procedure brings the maybe counter-intuitive view that **adding more variables simplifies the problem**. The last work of the thesis focuses on the representation of the variational Gaussian approximation. We avoid computational bottlenecks and add flexibility by representing the distribution with particles instead of using the mean and the covariance.

1.3 Gaussian Processes

The techniques mentioned above apply to many probabilistic models; however, we focus on Gaussian-based models, and more particularly Gaussian Processes ($\mathcal{GP}$**s**) [46]. A $\mathcal{GP}$ is a strong non-parametric tool to approximate functions using probabilistic methods. They were initially applied to regression problems with Gaussian noise, like the original **kriging problem** [9]. However, they are also used as prior over latent functions for more complex problems like classification, ordinal regression, and more. Compared to other general function approximators like neural networks, they have the advantage of providing uncertainty on the prediction they make. Most importantly, as their name suggests, they are based on Gaussian distributions, making them the best candidates for the presented work on augmentation. A full technical introduction to basic $\mathcal{GP}$**s** and its extensions is given in Section 2.2.

1.4 Open-source projects

All the works presented in this thesis, as well as additional tools, are backed-up by user-friendly packages in Julia [4]. Throughout my time as a Ph.D. student, I have developed numerous Julia packages and was involved in the JuliaGaussianProcesses organisation to develop a flexible, efficient and easy-to-use framework to work with $\mathcal{GP}$**s** from the very low-end to high-end interfaces through a series of packages: `KernelFunctions.jl` [16], `AbstractGPs.jl` [61], `ApproximateGPs.jl`, `InducingPoints.jl` and `GPLikelihoods.jl`. The particular strength of our work is the one-to-one mapping between theory and code. For example to define the posterior for some given data, the code looks like:

```
f = GP(mean_prior, kernel) # define an infinite-dimensional prior
fx = f(X, noise) # create a realization on the data X
fpost = posterior(fx, y) # create the posterior given the observations y
rand(fpost(x_test)) # sample from the predictive posterior of some test data
```

Here, each computational object represents exactly its mathematical equivalent.

The work of this thesis is represented as well with the package `AugmentedGPLikelihoods.jl`, which provide all the necessary tools to work with augmentations.

Julia's advantage is its strong interoperability capacity. This allows to use the augmentation work on more specialized implementations such as temporal $\mathcal{GP}$s with a concrete example given in `TemporalGPs.jl` (see `examples/augmented_inference.jl`).

Independently, I also developed `AugmentedGaussianProcesses.jl` [14] as a stand-alone $\mathcal{GP}$ package providing the augmentations techniques presented in the thesis, additional likelihoods, and standard inference approaches.

1.5 Thesis Outline

This thesis is constructed as follows:

- Chapter 2 introduces in detail all the common concepts of Bayesian inference and $\mathcal{GP}$s. There are introductions to these concepts in each published article, but this chapter dives more into the background theory. Bayesian inference, especially, is properly introduced, focusing on variational inference and sampling.
- Chapter 3 contains the paper *Efficient Gaussian Process Classification Using Pólya-Gamma Data Augmentation*, which was the first variable augmentation we explored.
- Chapter 4 introduces the paper *Multi-Class Gaussian Process Classification Made Conjugate: Efficient Inference via Data Augmentation*. This paper brings new augmentation concepts to a more complex problem: multi-class classification.
- Chapter 5 presents the paper *Automated Augmented Conjugate Inference for Non-conjugate Gaussian Process Models*. This work presents a generic way to identify augmentations in likelihoods and introduces a better understanding of the concepts behind it.
- Chapter 6 introduces the paper *Flexible and Efficient Inference with Particles for the Variational Gaussian Approximation* a completely different way of performing variational inference with a Gaussian distribution by using a continuous flow and particles.
- Chapter 7 discusses the different papers presented as well as some concrete outlooks on how to explore new models and new generalizations.
- Chapter 8 finishes this thesis with a general conclusion.
- The Appendix A also contains an additional workshop paper which does not fit the narrative of this thesis

  For all papers, a simplified view of the Contributor Roles Taxonomy (CReditT) details the contributions of each author.

2

Background

To fully comprehend the papers to be presented, we present a general overview of the needed concepts. A short introduction to the basic theory of Gaussian Processes as well as their extension to large datasets using inducing points [53] is given in Chapters 3, 4 and 5. However, this chapter presents a more thorough and basic description. Additionally, this chapter dives more into the basics of probabilistic Bayesian modeling, variational inference, and sampling methods.

2.1 Probabilistic Bayesian Modeling

Bayes' theorem is one of the simplest theorems in probability theory, and its proof fits in one line, yet its implications are immeasurably[1] important.

Let us give a very general modeling setting that we will follow for the rest of this chapter. Given a set of observed variables $\boldsymbol{X}$, a set of latent (unobserved) variables $\boldsymbol{\theta}$ with a **prior** distribution $p(\boldsymbol{\theta})$, and a **likelihood** function $p(\boldsymbol{X}|\boldsymbol{\theta})$, we obtain the **posterior** distribution $p(\boldsymbol{\theta}|\boldsymbol{X})$ via Bayes' theorem:

$$p(\boldsymbol{\theta}|\boldsymbol{X}) = \frac{p(\boldsymbol{X}|\boldsymbol{\theta})p(\boldsymbol{\theta})}{p(\boldsymbol{X})} = \frac{p(\boldsymbol{X}|\boldsymbol{\theta})p(\boldsymbol{\theta})}{\int p(\boldsymbol{X}|\boldsymbol{\theta})p(\boldsymbol{\theta})d\boldsymbol{\theta}}. \tag{2.1}$$

$p(\boldsymbol{X})$ represents the so-called evidence and can be used to compare different models (the dependency on the used model is implicit here). The posterior allows us to obtain a distribution of the latent variables with its uncertainty given the prior $p(\boldsymbol{\theta})$ and the observed data $\boldsymbol{X}$. The posterior is used for computing all kinds of expectations of the form $\mathbb{E}_{p(\boldsymbol{\theta}|\boldsymbol{X})}[f(\boldsymbol{\theta})] = \int f(\boldsymbol{\theta})p(\boldsymbol{\theta}|\boldsymbol{X})d\boldsymbol{\theta}$. Expected values of interest can be statistics of the posterior like the mean ($\mathbb{E}_{p(\boldsymbol{\theta}|\boldsymbol{X})}[\boldsymbol{\theta}]$) or predictive distribution of new data points $p(\boldsymbol{x}'|\boldsymbol{X}) = \mathbb{E}_{p(\boldsymbol{\theta}|\boldsymbol{X})}[p(\boldsymbol{x}'|\boldsymbol{\theta})]$.

[1] Pun intended.

Let's take the simple example of linear logistic regression, a discriminative model. Given an input $\boldsymbol{x} \in \mathbb{R}^D$ and a binary label $y \in \{0, 1\}$, we model the process as:

$$y \sim \text{Bernoulli}\left(\sigma(\boldsymbol{\theta}^\top \boldsymbol{x})\right),$$

where Bernoulli is the Bernoulli distribution, $\boldsymbol{\theta} \in \mathbb{R}^D$ is a vector of weights (our latent variable), and $\sigma : \mathbb{R} \to [0, 1]$ is the logistic function $\sigma(x) = \frac{1}{1+\exp(-x)}$. The likelihood function is given by:

$$p(y_i|\boldsymbol{\theta}, \boldsymbol{x}_i) = \sigma\left(\boldsymbol{\theta}^\top \boldsymbol{x}_i\right)^{y_i} \sigma\left(-\boldsymbol{\theta}^\top \boldsymbol{x}_i\right)^{1-y_i}.$$

Now let's suppose that we have N pairs of input $\boldsymbol{x}_i$ and label y_i, that we assume to be independent and identically distributed (iid), we get a training set $\boldsymbol{X} = \{\boldsymbol{x}_1, \ldots, \boldsymbol{x}_N\}$, $\boldsymbol{y} = \{y_1, \ldots, y_N\}$. With a prior distribution $p(\boldsymbol{\theta})$ on $\boldsymbol{\theta}$, we build the posterior as $p(\boldsymbol{\theta}|\boldsymbol{y}, \boldsymbol{X}) \propto p(\boldsymbol{\theta}) \prod_{i=1}^N p(y_i|\boldsymbol{\theta}, \boldsymbol{x}_i)$. We can then compute the predictive distribution for a new data input $\boldsymbol{x}^*$:

$$p(y^*|\boldsymbol{x}^*, \boldsymbol{y}, \boldsymbol{X}) = \int p(y^*, \boldsymbol{\theta}|\boldsymbol{x}^* \boldsymbol{y}, \boldsymbol{X}) d\boldsymbol{\theta} = \int p(y^*|\boldsymbol{\theta}, \boldsymbol{x}^*) p(\boldsymbol{\theta}|\boldsymbol{y}, \boldsymbol{X}) d\boldsymbol{\theta}. \tag{2.2}$$

Note that the last term of Equation (2.2) directly involves the posterior distribution $p(\boldsymbol{\theta}|\boldsymbol{y}, \boldsymbol{X})$. To solve this integral, we must either know the posterior distribution and compute the integral numerically (or analytically) or sample from the posterior and estimate the integral using Monte Carlo integration.

2.1.1 Posterior computations

Given a prior $p(\boldsymbol{\theta})$ and a likelihood $p(\boldsymbol{X}|\boldsymbol{\theta})$, computing the posterior distribution function (2.1) in closed-form requires the integral[2] $p(\boldsymbol{X}) = \int p(\boldsymbol{X}|\boldsymbol{\theta}) p(\boldsymbol{\theta}) d\boldsymbol{\theta}$. For most non-trivial models, this integral is intractable, and approximations to the posterior are needed. Such methods are introduced in Section 2.3.

However, in specific settings, computing the posterior in closed-form is possible. When the prior is said to be **conjugate** to the likelihood, the posterior is of the same probability distribution family as the prior and is analytically tractable [49]. It is worth emphasizing this seemingly trivial case since it will be exploited in Section 2.3.3. For a general example, we consider a likelihood part of the **exponential family**:

$$p(\boldsymbol{x}|\boldsymbol{\theta}) = h(\boldsymbol{x}) \exp(\boldsymbol{\eta}(\boldsymbol{\theta})^\top T(\boldsymbol{x}) - A(\boldsymbol{\theta})), \tag{2.3}$$

where $\boldsymbol{\theta}$ are the **distribution parameters**, $h(\boldsymbol{x})$ is the **base measure**, $\boldsymbol{\eta}(\boldsymbol{\theta})$ corresponds to the **natural parameters**, $T(\boldsymbol{x})$ are the **sufficient statistics** and $A(\boldsymbol{\theta})$ is the **log-partition**.

Formally, a **conjugate prior** to the likelihood (2.3) is defined as:

$$p(\boldsymbol{\theta}|\boldsymbol{\alpha}) = h'(\boldsymbol{\theta}) \exp(\boldsymbol{\eta}'(\boldsymbol{\alpha})^\top T'(\boldsymbol{\theta}) - A'(\boldsymbol{\alpha})), \tag{2.4}$$

[2]Even if the integral is known, it might not be enough to compute some expectations or statistics.

where $T'(\boldsymbol{\theta}) = \{\boldsymbol{\eta}(\boldsymbol{\theta}), A(\boldsymbol{\theta})\}$ and where $\boldsymbol{\alpha}$ represents the prior distribution parameters. Given a factorizable likelihood $p(\boldsymbol{X}|\boldsymbol{\theta}) = \prod_{i=1}^{N} p(\boldsymbol{x}_i|\boldsymbol{\theta})$, the posterior will be proportional to

$$p(\boldsymbol{\theta}|\boldsymbol{X}) \propto h'(\boldsymbol{\theta}) \exp\left(\left(\{\sum_{i=1}^{N} T(\boldsymbol{x}_i), N\} + \boldsymbol{\eta}'(\boldsymbol{\alpha})\right)^{\top} T'(\boldsymbol{\theta})\right). \tag{2.5}$$

Note that the only dependence on $\boldsymbol{X}$ is via the sufficient statistics $T(\boldsymbol{x})$.

Conjugate models are very practical as the posterior can be found in one step, but are very constraining in the choice of the prior. They tend to be considered too simple for many applications.

If the prior is not conjugate of the likelihood, an alternative is to look for **conditional conjugacy**. A parameter θ_i with a **conditionally conjugate prior** will have a full conditional distribution of the same family. The full conditional distribution is defined as $p(\theta_i|\boldsymbol{X}, \boldsymbol{\theta}_{/i})$ where $\boldsymbol{\theta}_{/i} = \{\theta_1, \ldots, \theta_{i-1}, \theta_{i+1}, \ldots \theta_D\}$. This notion of full conditional also extends to blocks of variables.

2.2 Gaussian Processes

Gaussian Processes ($\mathcal{GP}$**s**) are a class of stochastic processes used as non-parametric probabilistic representations of functions. A $\mathcal{GP}$ is a stochastic process $\{f_t\}$, where the joint distribution on any finite collection of random variables $\{f_t\}$ follows a (multivariate) Gaussian distribution [46]. Since all the variables are Gaussian, we can perform all linear operations analytically, making them computationally attractive. We can also compute marginals exactly, and a product of Gaussian distributions of the same variable is still proportional to another Gaussian.

A $\mathcal{GP}$ is uniquely specified by its **mean function** $\mu_0(\boldsymbol{x})$ and **kernel function** (also called covariance function) $k(\boldsymbol{x}, \boldsymbol{x}')$. $\mu_0(\boldsymbol{x})$ can be any real-valued function while $k(\boldsymbol{x}, \boldsymbol{x}')$ needs to be a positive-definite function (also called Mercer kernels). A symmetric function $k : \mathcal{X} \times \mathcal{X} \to \mathbb{R}$ is positive-definite on $\mathcal{X}$ if $\boldsymbol{w}^{\top} K \boldsymbol{w}$ where $K_{ij} = K(\boldsymbol{x}_i, \boldsymbol{x}_j)$ for any $\boldsymbol{w} \in \mathbb{R}^N$ and any $\{\boldsymbol{x}_i, \ldots, \boldsymbol{x}_N\} \in \mathcal{X}$.

One of the interpretations of a $\mathcal{GP}(\mu_0,\ k)$ is as a prior on the function space. Given a random function f with a $\mathcal{GP}$ prior, we can project f into a finite space by evaluating it on a set of data inputs $\boldsymbol{X} = \{\boldsymbol{x}_1, \ldots, \boldsymbol{x}_N\}$ such that we obtain the finite-dimensional vector $\boldsymbol{f}$ where $f_i = f(\boldsymbol{x}_i)$. The prior on the projected $\mathcal{GP}$ on $\boldsymbol{X}$ is given by $\mathcal{N}(\boldsymbol{\mu}_0(\boldsymbol{X}), K_X)$ where $\boldsymbol{\mu}_0(\boldsymbol{X}) = \{\mu_0(\boldsymbol{x}_i)\}_{i=1}^{N}$ and $K \in \mathbb{R}^{N \times N}$ is the kernel matrix, defined by $K_{ij} = k(\boldsymbol{x}_i, \boldsymbol{x}_j)$.

2.2.1 Gaussian Process Regression

Given our prior $p(\boldsymbol{f}) = \mathcal{N}(\boldsymbol{f}|\boldsymbol{\mu}_0, \boldsymbol{K})$, we can add noisy observations $\boldsymbol{y} = \{y_i\}_{i=1}^{N}$ for each respective $\boldsymbol{x}_i$ and model the process as:

$$y_i = f(\boldsymbol{x}_i) + \epsilon_i, \tag{2.6}$$

where $\epsilon_i \sim \mathcal{N}(0, \sigma^2)$. This leads to the likelihood $p(y_i|f_i) = \mathcal{N}(y_i|f_i, \sigma^2)$. Fortunately, adding a zero-mean Gaussian variable to another gives another Gaussian variable with increased variance and the posterior for $\boldsymbol{f}$ is given by $p(\boldsymbol{f}|\boldsymbol{y}) = \mathcal{N}(\boldsymbol{f}|\boldsymbol{y}, \boldsymbol{K}_{\boldsymbol{X}} + \sigma^2 I)$. The predictive distribution

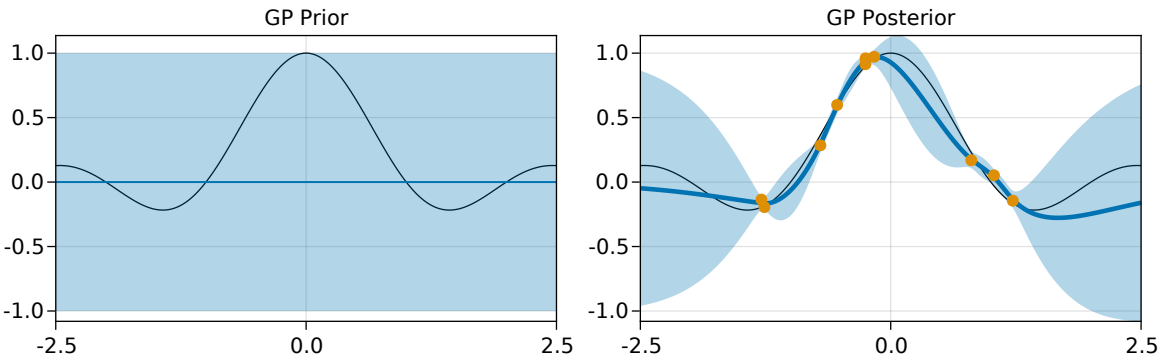


Figure 2.1: Illustration of the realization of a Gaussian Process. The black line is the true function f; the blue line is the mean of the prediction; the blue area represents the confidence interval of 2 standard deviations; the orange points represent observed data. Left: prediction on a grid given no observations. Right: prediction on a grid given a set of observations

of $f_* = f(\boldsymbol{x}_*)$ on a new input $\boldsymbol{x}_*$ can be evaluated by computing:

$$p(f_*|\boldsymbol{x}_*, \boldsymbol{X}, \boldsymbol{y}) = \int p(f_*|\boldsymbol{f}, \boldsymbol{x}_*)p(\boldsymbol{f}|\boldsymbol{X}, \boldsymbol{y})d\boldsymbol{f}. \tag{2.7}$$

This integral is analytically tractable and results in

$$p(f_*|\boldsymbol{x}_*, \boldsymbol{X}, \boldsymbol{y}) = \mathcal{N}(f_*|m_*, s_*), \tag{2.8}$$

where $m_* = K_{\boldsymbol{x}_*,\boldsymbol{X}}(K_{\boldsymbol{X}} + \sigma^2 I)^{-1}\boldsymbol{y}$ and $s_* = K_{\boldsymbol{x}_*} - K_{\boldsymbol{x}_*,\boldsymbol{X}}\left(K_{\boldsymbol{X}} + \sigma^2 I\right)^{-1} K_{\boldsymbol{X},\boldsymbol{x}_*}$, with $(K_{\boldsymbol{x}_*,\boldsymbol{X}})_i = k(\boldsymbol{x}_*, \boldsymbol{x}_i)$. The predictive distribution for f_* is Gaussian, with a known mean m_* and a measure of uncertainty given by the variance s_*. Note that s_* depends directly on $K_{\boldsymbol{x}_*,\boldsymbol{X}}$: if $\boldsymbol{x}_*$ is far from all points in $\boldsymbol{X}$ (in the sense of the distance used in the kernel k), then $K_{\boldsymbol{x}_*,\boldsymbol{X}}$ will be very small and the variance s_* maximized. The predictive uncertainty will be high when new inputs $\boldsymbol{x}_*$ are distant from the training data $\boldsymbol{X}$. A concrete example is shown on Figure 2.1.

2.2.2 Non-Conjugate Gaussian Processes

A Gaussian prior is only conjugate to the mean parameter of a Gaussian likelihood. Therefore, the $\mathcal{GP}$ posterior obtained in the previous section is only tractable for homoscedastic[3] Gaussian likelihoods. For all other cases we talk about **non-conjugate $\mathcal{GP}$s**. Examples of non-conjugate $\mathcal{GP}$ problems are binary classification, regression using non-Gaussian noise such as Student-t or Laplace noise, or Poisson regression. Other examples, such as multi-class classification or heteroscedastic regression, can nrequire multiple latent $\mathcal{GP}$s. Figure 2.2 shows an example of 1-dimensional binary classification with a $\mathcal{GP}$ where the posterior was approximated using variational inference (see Section 2.3.2). Although the $\mathcal{GP}$ does not recover exactly the true process, most of it lies in the $\mathcal{GP}$'s 95% confidence interval (blue band).

Posteriors of non-conjugate problems are not analytically tractable, and one needs to resort to the approximation methods presented in Section 2.3. A strong focus of this thesis is to

[3]The noise variance is independent of the input

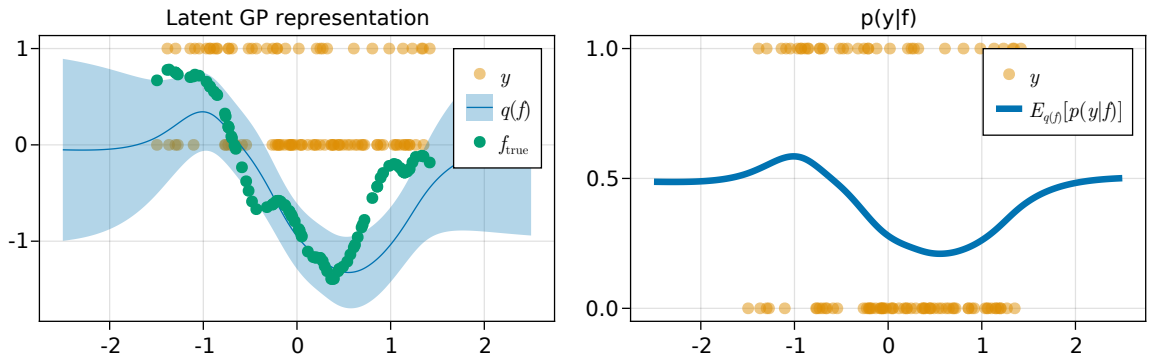


Figure 2.2: Illustration of a latent Gaussian process used for a binary classification problem. The Bernoulli likelihood is linked to the latent $\mathcal{GP}$ via the logistic function. On the left is shown the optimal variational posterior $q(f)$ in blue, compared to the true generation of f in green. Similar to Figure 2.1, the blue band represents one standard deviation. On the right, we show the expected predictive probability for y given the variational posterior $q(f)$ in blue.

take these non-conjugate likelihoods and find a representation where inference is simplified and basic methods can be used.

2.2.3 Sparse Gaussian Processes

One of the largest drawbacks of $\mathcal{GP}$**s**, regardless of the conjugacy of the likelihood, is the scalability with the number of observed samples. When computing the predictive mean and covariance, the inverse matrix operation in Equation (2.8) has a computational complexity of $\mathcal{O}(N^3)$ where N is the number of samples. For one-dimensional inputs ($D = 1$), solutions exist for specific kernels using state-space models representation [55, 52], leading to an $\mathcal{O}(N)$ complexity. However, higher-dimensional problems require alternative solutions. The first approach to reduce the complexity was to use a Nyström approximation [62]. Csató and Opper [11] proposed to create an approximation of the posterior using a subset of the points only in the context of online learning. Snelson and Ghahramani [51] expanded this theory to the offline framework and Csató [10] followed by Titsias [53] developed an alternative approximation based on KL divergence where the "inducing points" are not necessarily a subset of the training data and do not even have to belong to the same domain [31, 56]. For a unified view on sparse $\mathcal{GP}$**s**, see Quinonero-Candela and Rasmussen [45] and Bui et al. [7].

The works of thesis relying on inducing points are based on Titsias' approach [53]: The sparse approximation is made by defining a set of inducing points location $Z = \{\boldsymbol{z}_i\}_{i=1}^M$ and the realization of a $\mathcal{GP}$ u on them: $\boldsymbol{u}$ where $u_i = u(\boldsymbol{z}_i)$. We proceed to use variational inference (see Section 2.3.2) and approximate the posterior $p(\boldsymbol{u}, \boldsymbol{f}|\boldsymbol{y})$ by the variational distribution

$$q(\boldsymbol{u}, \boldsymbol{f}) = q(\boldsymbol{u}) \prod_{i=1}^{N} p(f_i|\boldsymbol{u}), \tag{2.9}$$

minimizing $\mathrm{KL}\left(q(\boldsymbol{u}, \boldsymbol{f})||p(\boldsymbol{u}, \boldsymbol{f}|\boldsymbol{y})\right)$. The assumption used is that all components of the random vector $\boldsymbol{f}$ are independent of each other given the random vector $\boldsymbol{u}$. It is a strong assumption, but the inference and prediction complexity reduces to $\mathcal{O}(NM^2 + M^3)$, where N can be reduced to a smaller batch-size B with stochastic inference approaches [23, 21]. Given $q(\boldsymbol{u}) = \mathcal{N}(\boldsymbol{\mu}, \boldsymbol{\Sigma})$,

the predictive distribution of $f_* = f(\boldsymbol{x}_*)$ on a new input $\boldsymbol{x}_*$ is given by

$$\begin{aligned} p(f_*|\boldsymbol{y}, \boldsymbol{X}) &= \int p(f_*|\boldsymbol{u}) p(\boldsymbol{u}|\boldsymbol{y}, \boldsymbol{X}) d\boldsymbol{u} \\ &\approx \int p(f_*|\boldsymbol{u}) q(\boldsymbol{u}) d\boldsymbol{u} \\ &= p(f_*|m_*, s_*), \end{aligned}$$

where $m_* = K_{\boldsymbol{x}_*,\boldsymbol{Z}} K_{\boldsymbol{Z}}^{-1} \boldsymbol{\mu}$ and $s_* = K_{\boldsymbol{x}_*} - K_{\boldsymbol{x}_*,\boldsymbol{Z}} K_{\boldsymbol{Z}}^{-1} (I - \Sigma) K_{\boldsymbol{Z}}^{-1} K_{\boldsymbol{Z},\boldsymbol{x}_*}$.

2.3 Approximate Bayesian Inference

The posterior distribution in Equation (2.1) cannot be computed in closed-form for non-trivial problems such as the ones presented in Section 2.2.2 and 2.2.3. We can approximate the posterior to obtain a valuable estimator for predictions and expected values of interest. **Approximate Bayesian Inference** is a research field of its own, and this chapter will focus specifically on sampling and Variational Inference, the most popular approximate inference methods for $\mathcal{GP}$**s**.

2.3.1 Sampling

We can compute predictive estimates $\mathbb{E}_{p(\boldsymbol{\theta}|\boldsymbol{X})}[f(\boldsymbol{\theta})]$ with Monte Carlo integration:

$$\mathbb{E}_{p(\boldsymbol{\theta}|\boldsymbol{X})}[f(\boldsymbol{\theta})] \approx \frac{1}{N} \sum_{i=1}^{N} f(\boldsymbol{\theta}_i), \quad \boldsymbol{\theta}_i \sim p(\boldsymbol{\theta}|\boldsymbol{X}),$$

where the samples $\boldsymbol{\theta}_i$ are iid.

Even if the posterior distribution $p(\boldsymbol{\theta}|\boldsymbol{X})$ is not available in closed-form or has no direct sampler, there are many alternatives to draw samples from it. The advantage of sampling is its unbiasedness: one obtains exact expectations in the limit of infinitely many samples. Sampling is an art of its own, and the number of methods is too large to mention them all in this thesis. Therefore, the scope is restricted to methods popular with or tailored to $\mathcal{GP}$**s**. In particular, we restrict ourselves to Markov Chain Monte Carlo (MCMC) methods.

Markov Chain Monte Carlo and Metropolis-Hastings

Markov Chain Monte Carlo (MCMC) methods generate a chain of variables $\boldsymbol{\theta}^t$ with the Markov assumption: $\boldsymbol{\theta}^t$ depends only on $\boldsymbol{\theta}^{t-1}$ and where the stationary distribution of $\boldsymbol{\theta}^t$ is the same as the target distribution $\pi(\boldsymbol{\theta})$ (for our use case the posterior $p(\boldsymbol{\theta}|\boldsymbol{X})$). MCMC methods require a transition probability $t(\boldsymbol{\theta}^{t+1}|\boldsymbol{\theta})$ which leaves the target stationary distribution invariant, i.e. $\pi(\boldsymbol{\theta}) = \int t(\boldsymbol{\theta}|\boldsymbol{\theta}')\pi(\boldsymbol{\theta}')d\boldsymbol{\theta}'$. Other properties such as detailed balance and ergodicity need to be satisfied as well [6, 42].

One of the most common algorithms to run a Markov Chain on a distribution $\pi(\boldsymbol{\theta})$ is the Metropolis-Hastings (MH) algorithm. The MH algorithm consists in having a proposal distribution $q(\boldsymbol{\theta}'|\boldsymbol{\theta})$ suggesting a new sample. Each proposed sample $\boldsymbol{\theta}'$ is randomly accepted or rejected with probability $p(\text{accept}) = \frac{\pi(\boldsymbol{\theta}')}{\pi(\boldsymbol{\theta})} \frac{q(\boldsymbol{\theta}|\boldsymbol{\theta}')}{q(\boldsymbol{\theta}'|\boldsymbol{\theta})} = A$. The choice of the proposal distribution q is the key to producing "good" chains with a high acceptance rate and a good exploration of $\boldsymbol{\theta}$'s parameter space. Next are presented some categories of choice for the proposal q.

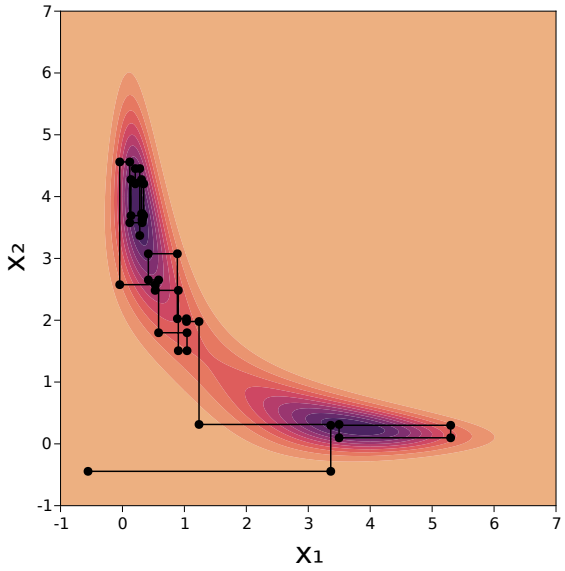


Figure 2.3: 20 steps of the Gibbs sampler trajectory on the Rosenbrock distribution in 2 dimensions.

Gibbs Sampling

Gibbs sampling is a particular MCMC method where we sample each component of the random vector one after another. The proposal distribution for each component is given by the full conditional $p(\theta_i|\boldsymbol{x}, \boldsymbol{\theta}_{/i})$, where $\boldsymbol{\theta}_{/i} = \{\theta_1, \dots \theta_{i-1}, \theta_{i+1}, \dots \theta_D\}$. The most prominent feature of Gibbs sampling is its acceptance probability, guaranteed to be 1:

$$\begin{aligned} A =& \frac{p(\theta_i^{t+1}, \boldsymbol{\theta}_{/i}^t|\boldsymbol{x})}{p(\theta_i^t, \boldsymbol{\theta}_{/i}^t|\boldsymbol{x})} \frac{p(\theta_i^t|\boldsymbol{x}, \boldsymbol{\theta}_{/i}^t)}{p(\theta_i^{t+1}|\boldsymbol{x}, \boldsymbol{\theta}_{/i}^t)} \\ =& \frac{p(\theta_i^{t+1}|\boldsymbol{x}, \boldsymbol{\theta}_{/i}^t)}{p(\theta_i^t|\boldsymbol{x}, \boldsymbol{\theta}_{/i}^t)} \frac{p(\boldsymbol{\theta}_{/i}^t|\boldsymbol{x})}{p(\boldsymbol{\theta}_{/i}^t|\boldsymbol{x})} \frac{p(\theta_i^t|\boldsymbol{x}, \boldsymbol{\theta}_{/i}^t)}{p(\theta_i^{t+1}|\boldsymbol{x}, \boldsymbol{\theta}_{/i}^t)} = 1. \end{aligned}$$

At every step, all proposed samples are therefore guaranteed to be accepted.

We illustrate the path of the sampler on a two-dimensional bimodal example in Figure 2.3.

The Gibbs sampling approach is a conundrum. On the one hand, sampling each component using the full conditional is easy since it only involves drawing a scalar. However, building a sampler for each full conditional at each step can be slow and costly. The sampler can also get stuck or move very slowly if the components are highly correlated with another. We can solve these drawbacks by using additional techniques like the **blocked Gibbs sampler** [28] where we sample groups of variables jointly, or **collapsed Gibbs sampling** [35] where we marginalize out some variables from the full conditional distributions. But blocked or collapsed updates are not always available and require heavier sampling machinery.

The augmentations proposed in this thesis allow using both the blocked and collapsed version by deriving the blocked full conditionals for each group of variables analytically. Experiments show that the correlations are very low between each group of variables, and that the sampler converges to the stationary distribution very fast.

Hamilton/Hybrid Monte Carlo

Hamiltonian Monte Carlo (HMC) or Hybrid Monte Carlo [13, 40, 3] is a MCMC method that uses Hamiltonian dynamics to make a new proposal. We augment $\boldsymbol{\theta}^t$ with an extra momentum $\boldsymbol{p}^t$ sampled randomly for every proposal from $\mathcal{N}(0, M)$ where M is the mass matrix. Next we run the Hamiltonian dynamics based on the Hamiltonian $H(\boldsymbol{\theta}, \boldsymbol{p}) = -\log \pi(\boldsymbol{\theta}) + \frac{1}{2}\boldsymbol{p}^\top M \boldsymbol{p}$ over L leapfrog steps with step size Δt. The proposal at time $L\Delta t$ is accepted or rejected based on the acceptance rate:

$$A = \min\left(1, \frac{\exp(-H(\boldsymbol{\theta}(L\Delta t), \boldsymbol{p}(L\Delta t)))}{\exp(-H(\boldsymbol{\theta}(0), \boldsymbol{p}(0)))}\right)$$

Hamiltonian dynamics normally keep the Hamiltonian invariant. However, symplectic (volume preserving) integrators, like the leapfrog method, only keep H approximately invariant [40]. The global error on H grows as $\mathcal{O}(L(\Delta t)^2)$. We get high acceptance rates while the dynamics lower the correlation between each sample by exploring the parameter space more freely than a basic random walk. We can tune the HMC algorithm parameters M, L, and Δt by drawing a series of adaptive samples and by adjusting to the local geometry of the potential function $-\log \pi(\boldsymbol{\theta})$. Figure 2.4 illustrates the sampling process with the Hamiltonian dynamics paths drawn with gray lines.

HMC is very popular due to its plug-and-play characteristics but suffers from different issues. It is gradient-based and can not sample discrete variables. The integration of the Hamiltonian dynamics requires $2L$ gradients per proposal. This computational cost can be prohibitively expensive for high-dimensional problems or for target distributions with costly computations.

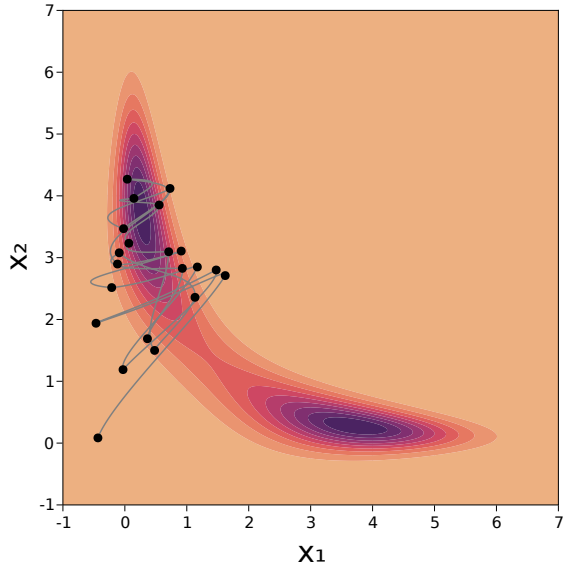


Figure 2.4: Illustration of the HMC sampler (gray lines are the Hamiltonian dynamics)

Other samplers

There are other solid choices for sampling from $\mathcal{GP}$s. For example, elliptical slice sampling Murray et al. [38] is particularly well-fitted for Gaussian priors. The No-U-turn sampling (NUTS) algorithm [25] is an extension of HMC where L is chosen automatically. We run the path

integration with both $\boldsymbol{p}$ and $-\boldsymbol{p}$ until one of the particles goes backward or if one of the Hamiltonian estimates becomes too inaccurate[4]. The proposal is finally sampled randomly from both paths. NUTS is good at avoiding oscillatory dynamics and is particularly strong for quadratic problems, which appear regularly in $\mathcal{GP}$**s** problems. Finally, another orthogonal approach to sample from predictive distributions with a known $\mathcal{GP}$ posterior is pathwise sampling [63]. By taking a mix of random Fourier features, specific to a particular class of kernels, the sampling complexity can be reduced from $\mathcal{O}(N^3)$ to $\mathcal{O}(T^3)$ where N is the number of test inputs, and T is the chosen number of basis.

2.3.2 Variational Inference

Variational Inference (VI), also called Variational Bayes, consists in approximating the posterior $p(\boldsymbol{\theta}|\boldsymbol{X})$ with another distribution $q(\boldsymbol{\theta})$. Given a family of distributions $\mathcal{Q}$, parametrized by the variational parameters $\boldsymbol{\varphi}$, one aims to solve the following optimization problem:

$$\boldsymbol{\varphi}^* = \arg_{\boldsymbol{\varphi}} \min \mathcal{D}\left(q_{\boldsymbol{\varphi}}(\boldsymbol{\theta}), p(\boldsymbol{\theta}|\boldsymbol{x})\right), \tag{2.10}$$

where $\mathcal{D}$ is a dissimilarity measure between two distributions and $q_{\boldsymbol{\varphi}}$ is the distribution $q \in \mathcal{Q}$ parametrized by $\boldsymbol{\varphi}$. One of the most used dissimilarity measure is the reverse Kullback-Leibler (KL) divergence, defined for continuous distributions as:

$$\mathrm{KL}\left(q(\boldsymbol{x})||p(\boldsymbol{x})\right) = \int q(\boldsymbol{x}) \log \frac{q(\boldsymbol{x})}{p(\boldsymbol{x})} d\boldsymbol{x} \tag{2.11}$$

The objective of Equation (2.10) or (2.11) is generally not directly tractable when the normalizer is not known. Since $p(\boldsymbol{\theta}|\boldsymbol{x})$ involves the normalization constant $p(\boldsymbol{x})$, one resorts to a surrogate function, the Variational Free Energy (VFE) (or its negative counterpart the Evidence Lower BOund (ELBO)):

$$\begin{aligned}
\mathrm{KL}\left(q_{\boldsymbol{\varphi}}(\boldsymbol{\theta})||p(\boldsymbol{\theta}|\boldsymbol{x})\right) &= \int q_{\boldsymbol{\varphi}}(\boldsymbol{\theta}) \left(\log q_{\boldsymbol{\varphi}}(\boldsymbol{\theta}) - \log p(\boldsymbol{\theta}|\boldsymbol{x})\right) d\boldsymbol{\theta} \\
&= \int q_{\boldsymbol{\varphi}}(\boldsymbol{\theta}) \left(\log q_{\boldsymbol{\varphi}}(\boldsymbol{\theta}) - \log p(\boldsymbol{\theta}, \boldsymbol{x}) - \log p(\boldsymbol{x})\right) d\boldsymbol{\theta} \\
&= \underbrace{-\log p(\boldsymbol{x})}_{:=C} + \int q_{\boldsymbol{\varphi}}(\boldsymbol{\theta}) \left(\log q_{\boldsymbol{\varphi}}(\boldsymbol{\theta}) - \log p(\boldsymbol{x}|\boldsymbol{\theta}) - \log p(\boldsymbol{\theta})\right) d\boldsymbol{\theta} \\
&= C - \mathbb{E}_{q_{\boldsymbol{\varphi}}}\left[\log p(\boldsymbol{x}|\theta)\right] + \mathrm{KL}\left(q_{\boldsymbol{\varphi}}(\boldsymbol{\theta})||p(\boldsymbol{\theta})\right) = \mathcal{F}(\boldsymbol{\varphi}) + C.
\end{aligned} \tag{2.12}$$

By minimizing $\mathcal{F}(\boldsymbol{\varphi})$ instead of the KL divergence, we can expect to find a solution close to the optimum of the problem stated in Equation (2.10).

A standard way to find the $\boldsymbol{\varphi}^* = \arg_{\boldsymbol{\varphi}} \min \mathcal{F}(\boldsymbol{\varphi})$ is to perform gradient descent on the variational parameters $\boldsymbol{\varphi}$:

$$\boldsymbol{\varphi}^{t+1} = \boldsymbol{\varphi}^t - \epsilon^t \nabla_{\boldsymbol{\varphi}} \mathcal{F}(\boldsymbol{\varphi}^t), \tag{2.13}$$

where $\epsilon^t > 0$ is the learning rate.

[4]Technical details are skipped here.

Computing the gradient $\nabla_{\boldsymbol{\varphi}}\mathcal{F}(\boldsymbol{\varphi})$ can be non-trivial. It involves derivatives over expectations, but "tricks" like reparametrization [54] help to reduce the cost of these computations.

The choice of the family $\mathcal{Q}$ is a trade-off decision. A richer, more complex family might be able to approximate the posterior better, but computing the KL and optimizing the variational parameters will be increasingly difficult. A standard example for continuous variables is the Variational Gaussian Approximation (VGA)[5], where the variational distribution $q_{\boldsymbol{\varphi}}$ is a Gaussian, i.e. $\mathcal{Q} = \{q \sim \mathcal{N}(\boldsymbol{m}, S)\}$, and $\boldsymbol{\varphi} = \{\boldsymbol{m}, S\}$. Many expectations can be computed analytically under VGA, in particular when the prior on $\boldsymbol{\theta}$ is Gaussian as well. The Gaussian distribution is easily reparametrizable, and it is straightforward to sample from it. Many operations will be of the cost $\mathcal{O}(D^3)$ where D is the dimensionality of $\boldsymbol{\theta}$. Restricting $\mathcal{Q}$ further by constraining the covariance S can reduce this cost. For example, setting S to be diagonal will reduce the number of variational parameters and avoid inverse matrix operations.

Mean-Field Approximation

We need assumptions on $\mathcal{Q}$ to reduce the computational cost of variational inference and scale with high-dimensional $\boldsymbol{\theta}$. The Mean-Field (MF) assumption imposes that every component of $\boldsymbol{\theta}$ is independent of each other. A MF variational family can be specified as:

$$\mathcal{Q}_{MF} = \left\{ q = \prod_{i=1}^{D} q_{\boldsymbol{\varphi}_i}(\theta_i) \right\}, \tag{2.14}$$

where $\boldsymbol{\varphi}_i$ are the variational parameters for the variable θ_i. Under the MF approximation, the number of variational parameters grows linearly with the dimensionality of $\boldsymbol{\theta}$ instead of quadratically. Additionally, integrals in Equation (2.12) can become one-dimensional or sometimes analytically tractable (the KL for example), and therefore more easily solvable. However, MF can not capture potential posterior correlations between the components of $\boldsymbol{\theta}$.

An intermediate solution is to assume independence between blocks of variables instead, similarly to the blocked Gibbs sampler. Given $\mathcal{I} = \{1, 2, \ldots, D\}$, the set of indices of $\boldsymbol{\theta}$, we can build into K independent subsets $\mathcal{I}_k \subseteq \mathcal{I}$ such that $\mathcal{I} = \cup_{k=1}^{K} \mathcal{I}_k$ and $\mathcal{I}_i \cap \mathcal{I}_j = \emptyset$, iff $i \neq j$. The variational distribution based on this Blocked Mean-Field (BMF) approximation is then defined as

$$q_{\boldsymbol{\varphi}}^{BMF}(\boldsymbol{\theta}) = \prod_{k=1}^{K} q_{\boldsymbol{\varphi}_k}(\boldsymbol{\theta}_{\mathcal{I}_k}), \tag{2.15}$$

where $\boldsymbol{\varphi}_k$ are the variational parameters for the set of variable $\boldsymbol{\theta}_{\mathcal{I}_k}$. The BMF approximation can capture correlations inside blocks of variables but loses some of MF's computational attractiveness.

Coordinate Ascent VI

The Coordinate Ascent Variational Inference (CAVI)[6] approach is an alternative to the gradient descent approach of Equation (2.13). Instead of moving all parameters at once in the

[5] The VGA is explored in more details in Chapter 6.

[6] The word ascent is used since the scheme was originally derived using the negative VFE, i.e., the ELBO.

gradient direction, we are interested in finding the optimal solution for each set of variational parameters $\boldsymbol{\varphi}_i$ one after another by keeping the others fixed:

$$\boldsymbol{\varphi}_i^* = \arg_{\boldsymbol{\varphi}_i} \min \mathcal{F}(\boldsymbol{\varphi}_i, \boldsymbol{\varphi}_{/i}), \tag{2.16}$$

where $\boldsymbol{\varphi}_{/i} = \{\boldsymbol{\varphi}_j | j \neq i\}$. Using the BMF approximation, we can update blocks of variational parameters at once. The optimal $\boldsymbol{\varphi}_i^*$ can be found by solving:

$$\nabla_{\varphi_i} \mathcal{F}(\boldsymbol{\varphi})|_{\varphi_i = \varphi_i^*} = 0, \tag{2.17}$$

or performing a partial version of the gradient descent from Equation (2.13). The solution to Equation (2.16) is always given by

$$q_{\boldsymbol{\varphi}_i}^*(\boldsymbol{\theta}_i) \propto \exp\left(\mathbb{E}_{q_{\boldsymbol{\varphi}}(\boldsymbol{\theta}_{/i})}\left[\log p\left(\boldsymbol{\theta}_i | \boldsymbol{\theta}_{/i}, \boldsymbol{x}\right)\right]\right) \tag{2.18}$$

where $\boldsymbol{\theta}_{/i}$ represent the collection of variables $\boldsymbol{\theta}_{/i} = \{\theta_j | j \neq i\}$ [37]. Even when the expectation involved in Equation (2.18) is available in closed-form, the resulting distribution might not always normalizable, but we are usually only interested in the different moments of $q_{\boldsymbol{\varphi}_i}(\boldsymbol{\theta}_i)$.

Algorithm 1 summarizes the CAVI algorithm. The order of the updates does not matter as long as the variational parameters $\boldsymbol{\varphi}$ are initialized in their respective domain.

Algorithm 1 CAVI algorithm

while $|\mathcal{F}^{t+1} - \mathcal{F}^t| > \epsilon$ **do**
  for $i \in \{1, \ldots, D\}$ **do**
    $\boldsymbol{\varphi}_i^{t+1} = \arg_{\boldsymbol{\varphi}_i} \min \mathcal{F}(\boldsymbol{\varphi}_{1:(i-1)}^{t+1}, \boldsymbol{\varphi}_i, \boldsymbol{\varphi}_{(i+1):D}^{t})$,
  end for
end while

The CAVI and Gibbs sampling algorithms are very similar in nature. The observations on Gibbs sampling also apply: CAVI updates on a distribution with MF is easily computable but has slower convergence, while updates with the BMF approach are more complex to derive, avoid some MF pitfalls, and provide a richer distribution.

Natural Gradients One interesting aspect of CAVI, is that it implicitly uses **natural gradients** [1]. A natural gradient is a gradient preconditioned with the inverse Fisher information matrix defined as

$$\mathcal{I}_\theta = \mathbb{E}_{p(\boldsymbol{x}|\boldsymbol{\theta})}\left[\left(\nabla_{\boldsymbol{\theta}} \log p(\boldsymbol{x}|\boldsymbol{\theta})\right)\left(\nabla_{\boldsymbol{\theta}} \log p(\boldsymbol{x}|\boldsymbol{\theta})\right)^\top\right] = -\mathbb{E}_{p(\boldsymbol{x}|\boldsymbol{\theta})}\left[\mathbf{H}(\log p(\boldsymbol{x}|\boldsymbol{\theta}))\right], \tag{2.19}$$

where $\mathbf{H}(f)$ is the Hessian matrix of the function f. The Fisher information matrix is a Riemannian metric that gives the direction of the steepest descent with respect to the KL divergence. The natural gradient is given by :

$$\widetilde{\nabla}_{\boldsymbol{\varphi}} \mathcal{F}(\boldsymbol{\varphi}) = \mathcal{I}^{-1} \nabla_{\boldsymbol{\varphi}} \mathcal{F}(\boldsymbol{\varphi})$$

The natural gradient works in a metric that maximizes the change of the infinitesimal KL divergence between the given distribution and its target [48]. The updates of the CAVI

algorithm 1 for exponential distributions, can be interpreted as natural gradient ascent updates with learning rate 1 [60].

$$\boldsymbol{\varphi}^{t+1} = \boldsymbol{\varphi}^t + \mathcal{I}_\theta^{-1} \nabla_{\boldsymbol{\varphi}} \mathcal{F}(\boldsymbol{\varphi}^t)$$

When working with constrained parameters like the covariance matrix of the Gaussian variational distribution, a step with a high learning rate might overshoot out of the cone of positive-definite matrices. Salimbeni et al. [48] proposes a given schedule to compensate while Lin et al. [34] forces a trajectory on a geodesic. Both approaches are computationally expensive, while we get this feature automatically.

2.3.3 Scale mixtures and conditionally conjugate likelihoods

We base a large part of this work on mixtures and use scale mixtures in particular. A scale mixture is a continuous mixture of a distribution with a varying scale parameter. A textbook example is the Student-T distribution which is a Gaussian scale mixture with a Gamma prior on the variance:

$$T_\nu(x) = \int_0^\infty \mathcal{N}(x|0,\omega)\,\mathrm{Ga}\left(\omega|\frac{\nu}{2},\frac{\nu}{2}\right) d\omega,$$

where Ga is a Gamma distribution. Another example is the Laplace distribution which is also a Gaussian scale mixture:

$$\mathrm{La}(x|\beta) = \int_0^\infty \mathcal{N}(x|0,\omega)\mathrm{Exp}\left(\omega|\frac{1}{2b^2}\right) d\omega,$$

where Exp is the exponential distribution.

These representations appear when computing predictive distributions. For example, when performing Gaussian linear regression with a fixed weight $\boldsymbol{\theta}$ and a Gamma prior on the likelihood variance σ^2, the resulting posterior predictive distribution will be a Student-T distribution.

This thesis shows that we can use this connection the other way around. Certain likelihoods $p(\boldsymbol{x}|\boldsymbol{\theta})$ can be defined as scale mixtures $\int p(\boldsymbol{x}|\boldsymbol{\theta},\omega)p(\omega)d\omega$. We can "unmarginalize" the likelihood by adding the scale variable ω to our model. We augment $p(\boldsymbol{x}|\boldsymbol{\theta})$ to $p(\boldsymbol{x},\omega|\boldsymbol{\theta})$. For example, we can augment a Student-T likelihood into a Gaussian likelihood with a Gamma prior on the variance. The advantage of the augmented model is to produce conditionally conjugate likelihoods for all the model variables as the next chapters will show.

3

Efficient Gaussian Process Classification Using Pólya-Gamma Data Augmentation

Before my doctoral studies, I worked on extending the work of Henao et al. [20] on Bayesian support vector machines to $\mathcal{GP}$**s** as well as scaling them up to big data [59]. This paper is not included in this thesis as it did not get published during my Ph.D. The approach proposed by Henao et al. [20] was the first step on the road of our research on augmentations. A natural continuation was to explore the binary classification problem with the logit link.

This paper extends the work of Polson et al. [44] on augmenting with Pólya-Gamma variables to $\mathcal{GP}$**s** and sparse $\mathcal{GP}$**s**. The main contributions of this paper are to show that the augmented model outperforms other state-of-the-art methods for $\mathcal{GP}$**s** but also a derivation of a remarkable equivalence between the variational bound derived Jaakkola and Jordan [27] and the Pólya-Gamma augmentation.

Authors:
Florian Wenzel,[1,*] Théo Galy-Fajou,[2,*] Christian Donner,[2] Marius Kloft,[1,3] Manfred Opper[2]
*Equal Contribution, [1]TU Kaiserslautern, Germany, [2]TU Berlin, Germany, [3]University of Southern California, USA

Details:
Type: Conference article Submitted: September 2018
Accepted: December 2018
URL: `https://ojs.aaai.org//index.php/AAAI/article/view/4481`
Conference: AAAI 2019

Contributions:

For an explanation of the terms see the Contributor Roles Taxonomy (CReditT)

	F.W.	T.G-F.	C.D.	M.K.	M.O.
Conceptualization	✓	✓	✓		✓
Methodology	✓	✓			
Formal Analysis	✓	✓			✓
Implementation		✓			
Investigation	✓	✓			
Writing - Original Draft	✓	✓	✓		
Writing - Review & Editing	✓	✓	✓		✓
Supervision					✓
Funding Acquisition				✓	✓

Efficient Gaussian Process Classification Using Pólya-Gamma Data Augmentation

Florian Wenzel,[1,*] **Théo Galy-Fajou,**[2,*] **Christan Donner,**[2] **Marius Kloft,**[1,3] **Manfred Opper**[2]
[*]Contributed equally, [1]TU Kaiserslautern, Germany, [2]TU Berlin, Germany, [3]University of Southern California, USA
wenzelfl@hu-berlin.de, galy-fajou@tu-berlin.de, christian.donner@bccn-berlin.de,
kloft@cs.uni-kl.de, manfred.opper@tu-berlin.de

Abstract

We propose a scalable stochastic variational approach to GP classification building on Pólya-Gamma data augmentation and inducing points. Unlike former approaches, we obtain closed-form updates based on natural gradients that lead to efficient optimization. We evaluate the algorithm on real-world datasets containing up to 11 million data points and demonstrate that it is up to two orders of magnitude faster than the state-of-the-art while being competitive in terms of prediction performance.

1 Introduction

Gaussian processes (GPs) Rasmussen and Williams (2005) provide a popular Bayesian non-linear non-parametric method for regression and classification. Because of their ability of accurately adapting to data and thus achieving high prediction accuracy while providing well calibrated uncertainty estimates, GPs are a standard method in several application areas, including geospatial predictive modeling Stein (2012) and robotics Dragiev, Toussaint, and Gienger (2011).

However, recent trends in data availability in the sciences and technology have made it necessary to develop algorithms capable of processing massive data John Walker (2014). Currently, GP classification has limited applicability to big data. Naive inference typically scales cubic in the number of data points, and exact computation of posterior and marginal likelihood is intractable.

Nevertheless, the combination of so-called sparse Gaussian process techniques with approximate inference methods, such as expectation propagation (EP) or the variational approach, have enabled GP classification for datasets containing millions of data points Hernández-Lobato and Hernández-Lobato (2016); Salimbeni, Eleftheriadis, and Hensman (2018).

While these results are already impressive, we will show in this paper that a speedup of up to two orders magnitudes can be achieved. Our approach is based on considering an augmented version of the original GP classification model and replacing the ordinary (stochastic) gradients for optimization by more efficient *natural gradients*, which is the standard Euclidean gradient multiplied by the inverse Fisher information matrix. Natural gradients recently have been successfully used in a variety of variational inference problems Honkela et al. (2010); Wenzel et al. (2017); Jähnichen et al. (2018).

Unfortunately, an efficient computation of the natural gradient for the GP classification problem is not straight forward. The use of the probit link function in Dezfouli and Bonilla (2015); Hernández-Lobato and Hernández-Lobato (2016); Mandt et al. (2017); Salimbeni, Eleftheriadis, and Hensman (2018) leads to expectations in the variational objective functions that can only be computed by numerical quadrature, thus, preventing efficient optimization.

We derive a natural-gradient approach to variational inference in GP classification based on the *logit* link. We exploit that the corresponding likelihood has an auxiliary variable representation as a continuous mixture of Gaussians involving Pólya-Gamma random variables Polson, Scott, and Windle (2013).

Unlike former approaches, our natural gradient updates can be computed in closed-form. Moreover, they have the advantage that they correspond to block-coordinate ascent updates and, therefore, learning rates close to one can be chosen. This leads to a fast and stable algorithm which is simple to implement. Our main contributions are as follows:

- We present a Gaussian process classification model using a logit link function that is based on Pólya-Gamma data augmentation and inducing points for Gaussian process inference.
- We derive an efficient inference algorithm based on stochastic variational inference and natural gradients. All natural gradient updates are given in closed-form and do not rely on numerical quadrature methods or sampling approaches. Natural gradients have the advantage that they provide effective second-order optimization updates.
- In our experiments, we demonstrate that our approach drastically improves speed up to two orders of magnitude while being competitive in terms of prediction performance. We apply our method to massive real-world

datasets up to 11 million points and demonstrate superior scalability.

The paper is organized as follows. In section 2 we discuss related work. In section 3 we introduce our novel scalable GP classification model and in section 4 we present an efficient variational inference algorithm. Section 5 concludes with experiments. Our code is available via Github[1].

2 Background and Related Work

Gaussian process classification Hensman and Matthews (2015) consider Gaussian process classification with a probit inverse link function and suggest a variational Gaussian model that builds on inducing points. By employing automatic differentiation, Salimbeni, Eleftheriadis, and Hensman (2018) generalize this approach to use natural gradients in non-conjugate GP models. Khan and Nielsen (2018) consider natural gradient updates in the setting of variational inference with exponential families. Unlike our approach, these methods do not benefit from closed-form updates and have to resort to numerical approximations. Moreover, our approach has the advantage that a higher learning rate close to one can be chosen leading to updates that can be interpreted as block-coordinate ascent updates.

Izmailov, Novikov, and Kropotov (2018) use tensor train decomposition to allow for the training of GP models with billions of inducing points. The updates are not computed in closed-form and they do not use natural gradients.

Dezfouli and Bonilla (2015) propose a general automated variational inference approach for sparse GP models with non-conjugate likelihood. Since they follow a black box approach and do not exploit model specific properties they do not employ efficient optimization techniques.

Hernández-Lobato and Hernández-Lobato (2016) follow an expectation propagation approach based on inducing points and have a similar computational cost as Hensman and Matthews (2015).

Pólya-Gamma data augmentation Polson, Scott, and Windle (2013) introduced the idea of data augmentation in logistic models using the class of Pólya-Gamma distributions. This allows for exact inference via Gibbs sampling or approximate variational inference schemes Scott and Sun (2013).

Linderman, Johnson, and Adams (2015) extend this idea to multinomial models and discuss the application for Gaussian processes with multinomial observations but their approach does not scale to big datasets and they do not consider the concept of inducing points.

[1] https://github.com/theogf/AugmentedGaussianProcesses.jl

3 Model

The logit GP Classification model is defined as follows. Let $X = (\boldsymbol{x}_1, \dots, \boldsymbol{x}_n) \in \mathbb{R}^{d \times n}$ be the d-dimensional training points with labels $\boldsymbol{y} = (y_1, \dots, y_n) \in \{-1, 1\}^n$. The likelihood of the labels is

$$p(\boldsymbol{y}|\boldsymbol{f}, X) = \prod_{i=1}^{n} \sigma(y_i f(\boldsymbol{x}_i)), \tag{1}$$

where $\sigma(z) = (1+\exp(-z))^{-1}$ is the logit link function and f is the latent decision function. We place a GP prior over f and obtain the joint distribution of the labels and the latent GP

$$p(\boldsymbol{y}, \boldsymbol{f}|X) = p(\boldsymbol{y}|\boldsymbol{f}, X)p(\boldsymbol{f}|X), \tag{2}$$

where $p(\boldsymbol{f}|X) = \mathcal{N}(\boldsymbol{f}|\boldsymbol{0}, K_{nn})$ and K_{nn} denotes the kernel matrix evaluated at the training points X. For the sake of clarity we omit the conditioning on X in the following.

3.1 Pólya-Gamma data augmentation

Due to the analytically inconvenient form of the likelihood function, inference for logit GP classification is a challenging problem. We aim to remedy this issue by considering an augmented representation of the original model. Later we will see that the augmented model is indeed advantageous as it leads to efficient closed-form updates in our variational inference scheme.

Polson, Scott, and Windle (2013) introduced the class of Pólya-Gamma random variables and proposed a data augmentation strategy for inference in models with binomial likelihoods. The augmented model has the appealing property that the likelihood of the latent function $\boldsymbol{f}$ is proportional to a Gaussian density when conditioned on the augmented Pólya-Gamma variables. This allows for Gibbs sampling methods, where model parameters and Pólya-Gamma variables can be sampled alternately from the posterior Polson, Scott, and Windle (2013). Alternatively, the augmentation scheme can be utilized to derive an efficient approximate inference algorithm in the variational inference framework, which will be pursued here.

The Pólya-Gamma distribution is defined as follows. The random variable $\omega \sim \mathrm{PG}(b, 0)$, $b > 0$ is defined by the moment generating function

$$\mathbb{E}_{\mathrm{PG}(\omega|\, b,0)}[\exp(-\omega t)] = \frac{1}{\cosh^b(\sqrt{t/2})}. \tag{3}$$

It can be shown that this is the Laplace transform of an infinite convolution of gamma distributions. The definition is related to our problem by the fact that the logit link can be written in a form that involves the cosh function, namely $\sigma(z_i) = \exp(\frac{1}{2}z_i)(2\cosh(\frac{z_i}{2}))^{-1}$. In the following we derive a representation of the logit link in terms of Pólya-Gamma variables.

First, we define the general $\mathrm{PG}(b, c)$ class which is derived by an exponential tilting of the $\mathrm{PG}(b, 0)$ density, it is given by

$$\mathrm{PG}(\omega|\, b, c) \propto \exp(-\frac{c^2}{2}\omega)\mathrm{PG}(\omega|\, b, 0).$$

From the moment generating function (3) the first moment can be directly computed

$$\mathbb{E}_{PG(\omega|b,c)}[\omega] = \frac{b}{2c}\tanh\left(\frac{c}{2}\right).$$

For the subsequently presented variational algorithm these properties suffice and the full representation of the Pólya-Gamma density $\mathrm{PG}(\omega|b, c)$ is not required.

We now adapt the data augmentation strategy based on Pólya-Gamma variables for the GP classification model. To do this we write the non-conjugate logistic likelihood function (1) in terms of Pólya-Gamma variables

$$\begin{aligned}\sigma(z_i) = (1+\exp(-z_i))^{-1} &= \frac{\exp(\frac{1}{2}z_i)}{2\cosh(\frac{z_i}{2})} \\ &= \frac{1}{2}\int \exp\left(\frac{z_i}{2} - \frac{z_i^2}{2}\omega_i\right) p(\omega_i)\mathrm{d}\omega_i, \end{aligned} \tag{4}$$

where $p(\omega_i) = \mathrm{PG}(\omega_i|1, 0)$ and by making use of (3). For more details see Polson, Scott, and Windle (2013). Using this identity and substituting $z_i = y_i f(x_i)$ we augment the joint density (2) with Pólya-Gamma variables

$$p(\boldsymbol{y}, \boldsymbol{\omega}, \boldsymbol{f}) \propto \exp\left(\frac{1}{2}\boldsymbol{y}^\top \boldsymbol{f} - \frac{1}{2}\boldsymbol{f}^\top \Omega \boldsymbol{f}\right) p(\boldsymbol{f})p(\boldsymbol{\omega}), \tag{5}$$

where $\Omega = \mathrm{diag}(\boldsymbol{\omega})$ is the diagonal matrix of the Pólya-Gamma variables $\{\omega_i\}$. In contrast to the original model (2) the augmented model is conditionally conjugate forming the basis for deriving closed-form updates in section 4.

Interestingly, employing a structured mean-field variational inference approach (cf. section 4) to the plain Pólya-Gamma augmented model (5) leads to the same bound for GP classification derived by Gibbs and MacKay (2000). This is an interesting new perspective on this bound since they do not employ a data augmentation approach. We provide a proof in appendix A.5. Our approach goes beyond Gibbs and MacKay (2000) by providing a fully Bayesian perspective, including a sparse GP prior (section 3.2) in the model and proposing a scalable inference algorithm based on natural gradients (section 4).

3.2 Sparse Gaussian process

Inference in GP models typically has the computational complexity $\mathcal{O}(n^3)$. We obtain a scalable approximation of our model and focus on inducing point methods Snelson and Ghahramani (2006). We follow a similar approach as in Hensman and Matthews (2015) and reduce the complexity to $\mathcal{O}(m^3)$, where m is number of inducing points.

We augment the latent GP f with m additional input-output pairs $(Z_1, u_1), \ldots, (Z_m, u_m)$, termed as *inducing inputs* and *inducing variables*. The function values of the GP $\boldsymbol{f}$ and the inducing variables $\boldsymbol{u} = (u_1, \ldots, u_m)$ are connected via

$$\begin{aligned} p(\boldsymbol{f}|\boldsymbol{u}) &= \mathcal{N}\left(\boldsymbol{f}|K_{nm}K_{mm}^{-1}\boldsymbol{u}, \widetilde{K}\right) \\ p(\boldsymbol{u}) &= \mathcal{N}\left(\boldsymbol{u}|0, K_{mm}\right), \end{aligned} \tag{6}$$

where K_{mm} is the kernel matrix resulting from evaluating the kernel function between all inducing inputs, K_{nm} is the cross-kernel matrix between inducing inputs and training points and $\widetilde{K} = K_{nn} - K_{nm}K_{mm}^{-1}K_{mn}$. Including the inducing points in our model gives the augmented joint distribution

$$p(\boldsymbol{y}, \boldsymbol{\omega}, \boldsymbol{f}, \boldsymbol{u}) = p(\boldsymbol{y}|\boldsymbol{\omega}, \boldsymbol{f})p(\boldsymbol{\omega})p(\boldsymbol{f}|\boldsymbol{u})p(\boldsymbol{u}) \tag{7}$$

Note that the original model (2) can be recovered by marginalizing $\boldsymbol{\omega}$ and $\boldsymbol{u}$.

4 Inference

The goal of Bayesian inference is to compute the posterior of the latent model variables. Because this problem is intractable for the model at hand, we employ variational inference to map the inference problem to a feasible optimization problem. We first chose a family of tractable variational distributions and select the best candidate by minimizing the Kullback-Leibler divergence between the variational distribution and the posterior. This is equivalent to optimizing a lower bound on the marginal likelihood, known as evidence lower bound (ELBO) Jordan et al. (1999); Wainwright and Jordan (2008).

In the following we develop a stochastic variational inference (SVI) algorithm that enables stochastic optimization based on natural gradient updates which are given in closed-form.

4.1 Why use natural gradients?

Using the natural gradient over the standard Euclidean gradient is favorable since natural gradients are invariant to reparameterization of the variational family Amari and Nagaoka (2007); Martens (2017) and provide effective second-order optimization updates Amari (1998); Hoffman et al. (2013).

The superiority of using natural gradients in our approach can be explained by the following. We reformulate the GP classification model as an augmented model which is conditionally conjugate. When using a learning rate of one, the natural gradient updates correspond to block-coordinate ascent updates, i.e. in each iteration each parameter is set to its optimal value given the remaining parameters (see appendix A.4 and Hoffman et al. (2013)). In practice, we employ stochastic variational inference, i.e. we only use mini-batches of the data to obtain a noisy version of the natural gradient. In this setting, learning rates slightly less than one have to be chosen.

This is in contrast to former natural gradient based approaches, e.g. Salimbeni, Eleftheriadis, and Hensman (2018), that focus on the original non-conjugate GP classification model. Although they benefit from using natural gradients, they have the disadvantage that their updates do not correspond to coordinate-ascent updates. Thus, learning rates that are much smaller that one have to be used to assure convergence.

Therefore, in our approach, we can use much higher learning rates and optimization is faster and more stable which we demonstrate in the experiments.

4.2 Variational approximation

We aim to approximate the posterior of the inducing points $p(\boldsymbol{u}|\boldsymbol{y})$ and apply the methodology of variational inference to the marginal joint distribution $p(y, \omega, u) = p(\boldsymbol{y}|\boldsymbol{\omega}, \boldsymbol{u})p(\boldsymbol{\omega})p(\boldsymbol{u})$. Following a similar approach as Hensman and Matthews (2015), we apply Jensen's inequality to obtain a tractable lower bound on the log-likelihood of the labels

$$\begin{aligned}\log p(\boldsymbol{y}|\boldsymbol{\omega}, \boldsymbol{u}) &= \log \mathbb{E}_{p(f|u)}[p(\boldsymbol{y}|\boldsymbol{\omega}, f)] \\ &\geq \mathbb{E}_{p(f|u)}[\log p(\boldsymbol{y}|\boldsymbol{\omega}, f)]. \end{aligned} \tag{8}$$

By this inequality we construct a variational lower bound on the evidence

$$\begin{aligned}\log p(\boldsymbol{y}) &\geq \mathbb{E}_{q(\boldsymbol{u},\boldsymbol{\omega})}[\log p(\boldsymbol{y}|\boldsymbol{u}, \boldsymbol{\omega})] - \mathrm{KL}\left(q(\boldsymbol{u}, \boldsymbol{\omega})||p(\boldsymbol{u}, \boldsymbol{\omega})\right) \\ &\geq \mathbb{E}_{p(\boldsymbol{f}|\boldsymbol{u})q(\boldsymbol{u})q(\boldsymbol{\omega})}[\log p(\boldsymbol{y}|\boldsymbol{\omega}, \boldsymbol{f})] \\ &\quad - \mathrm{KL}\left(q(\boldsymbol{u}, \boldsymbol{\omega})||p(\boldsymbol{u}, \boldsymbol{\omega})\right) \\ &=: \mathcal{L},\end{aligned}$$

where the first inequality is the usual evidence lower bound (ELBO) in variational inference and the second inequality is due to (8).

We follow a structured mean-field approach Wainwright and Jordan (2008) and assume independence between the inducing variables u and Pólya-Gamma variables ω, yielding a variational distribution of the form $q(u, \omega) = q(u)q(\omega)$. Setting the functional derivative of $\mathcal{L}$ w.r.t. $q(u)$ and $q(\omega)$ to zero, respectively, results in the following consistency condition for the maximum,

$$q(\boldsymbol{u}, \boldsymbol{\omega}) = q(\boldsymbol{u}) \prod_i q(\omega_i), \tag{9}$$

with $q(\omega_i) = \mathrm{PG}(\omega_i|1, c_i)$ and $q(\boldsymbol{u}) = \mathcal{N}(\boldsymbol{u}|\boldsymbol{\mu}, \Sigma)$. Remarkably, we do not have to use the full Pólya-Gamma class $\mathrm{PG}(\omega_i|b_i, c_i)$, but instead consider the restricted class $b_i = 1$ since it already contains the optimal distribution.

We use (9) as variational family which is parameterized by the variational parameters $\{\boldsymbol{\mu}, \Sigma, \boldsymbol{c}\}$ and obtain a closed-form expression of the variational bound

$$\begin{aligned}&\mathcal{L}(\boldsymbol{c}, \boldsymbol{\mu}, \Sigma) \\ &= \mathbb{E}_{p(\boldsymbol{f}|\boldsymbol{u})q(\boldsymbol{u})q(\boldsymbol{\omega})}[\log p(\boldsymbol{y}|\boldsymbol{\omega}, \boldsymbol{f})] - \mathrm{KL}\left(q(\boldsymbol{u}, \boldsymbol{\omega})||p(\boldsymbol{u}, \boldsymbol{\omega})\right) \\ &\overset{\mathsf{c}}{=} \frac{1}{2}\Bigg(\log|\Sigma| - \log|K_{mm}|) - \mathrm{tr}(K_{mm}^{-1}\Sigma) - \boldsymbol{\mu}^\top K_{mm}^{-1}\boldsymbol{\mu} \\ &\quad + \sum_i \Big\{ y_i \boldsymbol{\kappa}_i \boldsymbol{\mu} - \theta_i \left(\widetilde{K}_{ii} - \boldsymbol{\kappa}_i \Sigma \boldsymbol{\kappa}_i^\top - \boldsymbol{\mu}^\top \boldsymbol{\kappa}_i^\top \boldsymbol{\kappa}_i \boldsymbol{\mu} \right) \\ &\quad + c_i^2 \theta_i - 2 \log\cosh\frac{c_i}{2} \Big\} \Bigg),\end{aligned} \tag{10}$$

where $\theta_i = \frac{1}{2c_i}\tanh\left(\frac{c_i}{2}\right)$ and $\boldsymbol{\kappa}_i = K_{im}K_{mm}^{-1}$. Remarkably, all intractable terms involving expectations of $\log \mathrm{PG}(\omega_i|1, 0)$ cancel out. Details are provided in appendix A.2.

4.3 Stochastic variational inference

Our algorithm alternates between updates of the local variational parameters $\boldsymbol{c}$ and global parameters $\boldsymbol{\mu}$ and Σ. In each iteration we update the parameters based on a mini-batch of the data $\mathcal{S} \subset \{1, ..., n\}$ of size $s = |\mathcal{S}|$.

We update the *local parameters* $\boldsymbol{c}_\mathcal{S}$ in the mini-batch $\mathcal{S}$ by employing coordinate ascent. To this end, we fix the global parameters and analytically compute the unique maximum of (10) w.r.t. the local parameters, leading to the updates

$$c_i = \sqrt{\widetilde{K}_{ii} + \boldsymbol{\kappa}_i \Sigma \boldsymbol{\kappa}_i^\top + \boldsymbol{\mu}^\top \boldsymbol{\kappa}_i^\top \boldsymbol{\kappa}_i \boldsymbol{\mu}} \tag{11}$$

for $i \in \mathcal{S}$.

We update the *global parameters* by employing stochastic optimization of the variational bound (10). The optimization is based on stochastic estimates of the natural gradients of the global parameters. We use the natural parameterization of the variational Gaussian distribution, i.e., the parameters $\boldsymbol{\eta}_1 := \Sigma^{-1}\boldsymbol{\mu}$ and $\eta_2 = -\frac{1}{2}\Sigma^{-1}$. Using the natural parameters results in simpler and more effective updates. The natural gradients based on the mini-batch $\mathcal{S}$ are given by

$$\begin{aligned}\widetilde{\nabla}_{\boldsymbol{\eta}_1}\mathcal{L}_\mathcal{S} &= \frac{n}{2s}\boldsymbol{\kappa}_\mathcal{S}^\top \boldsymbol{y}_\mathcal{S} - \boldsymbol{\eta}_1 \\ \widetilde{\nabla}_{\eta_2}\mathcal{L}_\mathcal{S} &= -\frac{1}{2}\left(K_{mm}^{-1} + \frac{n}{s}\boldsymbol{\kappa}_\mathcal{S}^\top \Theta_\mathcal{S} \boldsymbol{\kappa}_\mathcal{S} \right) - \eta_2,\end{aligned} \tag{12}$$

where $\Theta = \mathrm{diag}(\boldsymbol{\theta})$ and $\theta_i = \frac{1}{2c_i}\tanh\left(\frac{c_i}{2}\right)$. The factor $\frac{n}{s}$ is due to the rescaling of the mini-batches. The global parameters are updated according to a stochastic natural gradient ascent scheme. We employ the adaptive learning rate method described by Ranganath et al. (2013).

The natural gradient updates always lead to a positive definite covariance matrix[2] and in contrast to Hensman and Matthews (2015) our implementation does not require any assurance for positive-definiteness of the variational covariance matrix Σ. Details for the derivation of the updates can be found in appendix A.3. The complexity of each iteration in the inference scheme is $\mathcal{O}(m^3)$, due to the inversion of the matrix η_2.

[2] This follows directly since K_{mm} and Θ are positive definite.

On the quality of the approximation In other applications of variational inference to GP classification, one tries to approximate the posterior directly by a Gaussian $q^*(f)$ which minimizes the Kullback-Leibler divergence between the variational distribution and the true posterior Hensman and Matthews (2015). On the other hand, in our paper, we apply variational inference to the augmented model, looking for the best distribution that factorizes in the Pólya-Gamma variables ω_i and the original function f. This approach also yields a Gaussian approximation $q(f)$ as a factor in the optimal density. Of course $q(f)$ will be different from the âĂŸoptimalâĂŹ $q^*(f)$. We could however argue that asymptotically, in the limit of a large number of data, the predictions given by both densities may not be too different, as the posterior uncertainty for both densities should become small Opper and Archambeau (2009).

It would be interesting to see how the ELBOs of the two variational approaches, which both give a lower bound on the likelihood of the data, differ. Unfortunately, such a computation would require the knowledge of the optimal $q^*(f)$. However, we can obtain some estimate of this difference when we assume that we use the *same* Gaussian density $q(f)$ for both bounds as an approximation. In this case, we obtain

$$\mathcal{L}_{\text{orig}} - \mathcal{L}_{\text{augmented}} = \mathbb{E}_{q(f)}[\text{KL}\left(q(\omega)||p(\omega|f,y)\right)].$$

This lower bound on the gap is small if on average the variational approximation $q(\omega)$ is close to the posterior $p(\omega|f,y)$. For the sake of simplicity we consider here the non-sparse case, i.e. the inducing points equal the training points ($f = u$). However, it is straight-forward to extend the results also to the sparse case.

We empirically investigate the quality of our approximation in experiment 5.1.

Predictions The approximate posterior of the GP values and inducing variables is given by $q(\boldsymbol{f}, \boldsymbol{u}) = p(\boldsymbol{f}|\boldsymbol{u})q(\boldsymbol{u})$, where $q(\boldsymbol{u}) = \mathcal{N}(\boldsymbol{u}|\boldsymbol{\mu}, \Sigma)$ denotes the optimal variational distribution. To predict the latent function values f_* at a test point x_* we substitute our approximate posterior into the standard predictive distribution

$$\begin{aligned} p(f_*|y) &= \int p(f_*|\boldsymbol{f}, \boldsymbol{u})p(\boldsymbol{f}, \boldsymbol{u}|y)\mathrm{d}\boldsymbol{f}\mathrm{d}\boldsymbol{u} \\ &\approx \int p(f_*|\boldsymbol{f}, \boldsymbol{u})p(\boldsymbol{f}|\boldsymbol{u})q(\boldsymbol{u})\mathrm{d}\boldsymbol{f}\mathrm{d}\boldsymbol{u} \\ &= \int p(f_*|\boldsymbol{u})q(\boldsymbol{u})\mathrm{d}\boldsymbol{u} = \mathcal{N}\left(f^*|\mu_*,\ \sigma_*^2\right), \end{aligned} \tag{13}$$

where the prediction mean is $\mu_* = K_{*m}K_{mm}^{-1}\boldsymbol{\mu}$ and the variance $\sigma_*^2 = K_{**} + K_{*m}K_{mm}^{-1}(\Sigma K_{mm}^{-1} - I)K_{m*}$. The matrix K_{*m} denotes the kernel matrix between the test point and the inducing points and K_{**} the kernel value of the test point. The distribution of the test labels is easily computed by applying the logit link function to (13),

$$p(y_* = 1|y) = \int \sigma(f_*)p(f_*|y)\mathrm{d}f_*. \tag{14}$$

This integral is analytically intractable but can be computed numerically by quadrature methods. This is adequate and fast since the integral is only one-dimensional.

Computing the mean and the variance of the predictive distribution has complexity $\mathcal{O}(m)$ and $\mathcal{O}(m^2)$, respectively.

Optimization of the hyperparameters We select the optimal kernel hyperparameters by maximizing the marginal likelihood $p(y|h)$, where h denotes the set of hyperparameters (this approach is called empirical Bayes Maritz and Lwin (1989)). We follow an approximate approach and optimize the fitted variational lower bound $\mathcal{L}(h)$ (10) as a function of h by alternating between optimization steps w.r.t. the variational parameters and the hyperparameters Mandt, Hoffman, and Blei (2016).

5 Experiments

We compare our proposed method, efficient Gaussian process classification (X-GPC), with the state-of-the-art methods SVGPC Salimbeni, Eleftheriadis, and Hensman (2018), provided in the package GPflow[3] Matthews et al. (2017), which builds on TensorFlow and the EP approach EPGPC by Hernández-Lobato and Hernández-Lobato (2016), implemented in R. All methods are applied to real-world datasets containing up to 11 million data points.

In all experiments a squared exponential covariance function with a common length scale parameter for each dimension, an amplitude parameter and an additive noise parameter is used. The kernel hyperparameters are initialized to the same values and optimized using Adam Kingma and Ba (2014), while inducing points location are initialized via k-means++ Arthur and Vassilvitskii (2007) and kept fixed during training. The SVI based methods, X-GPC and SVGPC, use an adaptive learning rate. All algorithms are run on a single CPU. We experiment on 12 datasets from the OpenML website and the UCI repository ranging from 768 to 11 million data points. In the first experiment (section 5.1), we examine the quality of the approximation provided by X-GPC. In the next experiment, we evaluate the prediction performance and run time of X-GPC and SVGPC and EPGPC on several real-world datasets. Finally, in 5.3, we examine the sensitivity of all methods to the number of inducing points.

5.1 Quality of the approximation

We empirically examine the quality of the variational approximation provided by our method. In Fig. 1, we compare the approximations to the true posterior obtained by employing an asymptotically correct Gibbs sampler Polson and Scott (2011); Linderman, Johnson, and Adams (2015). We compare the posterior mean and variance as well as the prediction probabilities with the ground truth. Since the Gibbs sampler

[3] We use GPflow version 1.2.0.

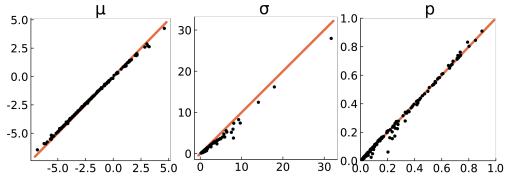


Figure 1: Posterior mean (μ), variance (σ) and predictive marginals (p) of the Diabetes dataset. Each plot shows the MCMC ground truth on the x-axis and the estimated value of our model on the y-axis. Our approximation is very close to the ground truth.

does not scale to large datasets we experiment on the small Diabetes dataset. In Fig. 1 we plot the approximated values vs. the ground truth. We find that our approximation is very close to the true posterior.

5.2 Numerical comparison

Dataset		**X-GPC**	SVGPC	EPGPC
aXa	Error	$\mathbf{0.17 \pm 0.07}$	$\mathbf{0.17 \pm 0.07}$	$\mathbf{0.17 \pm 0.07}$
$n = 36,974$	NLL	$\mathbf{0.29 \pm 0.13}$	0.36 ± 0.13	0.34 ± 0.13
$d = 123$	Time	$\mathbf{47 \pm 2.2}$	451 ± 7.8	214 ± 4.8
Bank Market.	Error	0.14 ± 0.12	$\mathbf{0.12 \pm 0.12}$	$\mathbf{0.12 \pm 0.13}$
$n = 45,211$	NLL	$\mathbf{0.27 \pm 0.22}$	0.31 ± 0.26	0.33 ± 0.20
$d = 43$	Time	$\mathbf{9 \pm 1.5}$	205 ± 6.6	46 ± 3.5
Click Pred.	Error	$\mathbf{0.17 \pm 0.00}$	$\mathbf{0.17 \pm 0.00}$	$\mathbf{0.17 \pm 0.01}$
$n = 399,482$	NLL	$\mathbf{0.39 \pm 0.07}$	0.46 ± 0.00	0.46 ± 0.01
$d = 12$	Time	$\mathbf{4.5 \pm 1.3}$	102 ± 3.0	8.1 ± 0.45
Cod RNA	Error	$\mathbf{0.04 \pm 0.00}$	$\mathbf{0.04 \pm 0.00}$	$\mathbf{0.04 \pm 0.00}$
$n = 343,564$	NLL	$\mathbf{0.11 \pm 0.03}$	0.13 ± 0.00	0.12 ± 0.00
$d = 8$	Time	$\mathbf{3.7 \pm 0.13}$	115 ± 4.3	869 ± 5.2
Diabetes	Error	$\mathbf{0.23 \pm 0.07}$	$\mathbf{0.23 \pm 0.06}$	0.24 ± 0.06
$n = 768$	NLL	$\mathbf{0.47 \pm 0.11}$	0.47 ± 0.10	0.48 ± 0.09
$d = 8$	Time	8.8 ± 0.12	150 ± 5.1	$\mathbf{8 \pm 0.45}$
Electricity	Error	$\mathbf{0.24 \pm 0.06}$	0.26 ± 0.06	0.26 ± 0.06
$n = 45,312$	NLL	$\mathbf{0.31 \pm 0.17}$	0.53 ± 0.08	0.53 ± 0.06
$d = 8$	Time	$\mathbf{8.2 \pm 0.48}$	356 ± 6.9	13.5 ± 1.50
German	Error	$\mathbf{0.25 \pm 0.12}$	$\mathbf{0.25 \pm 0.11}$	0.26 ± 0.13
$n = 1,000$	NLL	$\mathbf{0.44 \pm 0.17}$	0.51 ± 0.15	0.53 ± 0.11
$d = 20$	Time	17 ± 0.42	374 ± 7.3	$\mathbf{5.2 \pm 0.03}$
Higgs	Error	$\mathbf{0.33 \pm 0.01}$	0.45 ± 0.01	0.38 ± 0.01
$n = 11,000,000$	NLL	$\mathbf{0.55 \pm 0.13}$	0.69 ± 0.00	0.66 ± 0.00
$d = 28$	Time	$\mathbf{23 \pm 0.88}$	294 ± 54	8732 ± 867
IJCNN	Error	0.03 ± 0.01	0.06 ± 0.01	$\mathbf{0.02 \pm 0.01}$
$n = 141,691$	NLL	0.10 ± 0.03	0.15 ± 0.07	$\mathbf{0.09 \pm 0.04}$
$d = 22$	Time	$\mathbf{17 \pm 0.44}$	1033 ± 45	756 ± 8.6
Mnist	Error	0.14 ± 0.01	0.44 ± 0.13	$\mathbf{0.12 \pm 0.01}$
$n = 70,000$	NLL	$\mathbf{0.24 \pm 0.10}$	0.66 ± 0.11	0.27 ± 0.01
$d = 780$	Time	$\mathbf{200 \pm 5.5}$	991 ± 23	806 ± 5.2
Shuttle	Error	$\mathbf{0.01 \pm 0.01}$	$\mathbf{0.01 \pm 0.00}$	$\mathbf{0.01 \pm 0.01}$
$n = 58,000$	NLL	$\mathbf{0.07 \pm 0.01}$	$\mathbf{0.07 \pm 0.00}$	$\mathbf{0.07 \pm 0.01}$
$d = 9$	Time	$\mathbf{0.01 \pm 0.00}$	7.5 ± 0.7	100 ± 0.63
SUSY	Error	$\mathbf{0.21 \pm 0.00}$	0.22 ± 0.00	0.22 ± 0.00
$n = 5,000,000$	NLL	$\mathbf{0.31 \pm 0.10}$	0.49 ± 0.01	0.50 ± 0.00
$d = 18$	Time	$\mathbf{14 \pm 0.29}$	$10,000$	$10,000$
wXa	Error	$\mathbf{0.03 \pm 0.01}$	0.04 ± 0.01	$\mathbf{0.03 \pm 0.01}$
$n = 34,780$	NLL	0.27 ± 0.07	0.25 ± 0.07	$\mathbf{0.19 \pm 0.06}$
$d = 300$	Time	66 ± 16	612 ± 11	$\mathbf{1.4 \pm 0.10}$

Table 1: Average test prediction error, negative test log-likelihood (NLL) and time in seconds along with one standard deviation. Best values are highlighted.

We evaluate the prediction performance and run time of our method X-GPC and the competing methods SVGPC and EPGPC. We experiment on a variety of different datasets and report the resulting prediction error, negative test log-likelihood and run time for each method in table 1.

The experiments are conducted as follows. For each dataset we perform a 10-fold cross-validation and for datasets with more than 1 million points, we limit the test set to 100,000 points. We report the average prediction error, the negative test log-likelihood (14) and the run time along with one standard deviation. For all datasets, we use 100 inducing points and a mini-batch size of 100 points.

For X-GPC we find that the following simple convergence criterion on the global parameters leads to good results: a sliding window average being smaller than a threshold of 10^{-4} . Unfortunately, the original implementations of SVGPC and EPGPC do not include a convergence criterion. We find that the trajectories of the global parameters of SVGPC tend to be noisy, and using a convergence criterion on the global parameters often leads to poor results. To have a fair comparison, we therefore monitor the convergence of the prediction performance on a hold-out set and use a sliding window average of size 5 and threshold 10^{-3} as convergence criterion for all methods.

We observe that X-GPC is about one to two orders of magnitude faster than SVGPC and EPGPC on most datasets. Only on the dataset wXa, EPGPC is slightly faster than X-GPC. The prediction error is similar for all methods but X-GPC outperforms the competitors in terms of the test log-likelihood on most datasets (aXa, Bank Marketing, Click Prediction, Cod RNA, Diabetes, Electricity, German, Higgs, Mnist, SUSY). This means that the confidence levels in the predictions are better calibrated for X-GPC, i.e. when predicting a wrong label SVGPC and EPGPC tend to be more confident than X-GPC.

Performance as a function of time Since all considered methods are based on an optimization schemes, there is a trade-off between the run time of the algorithm and the prediction performance. We make this trade-off transparent by plotting the prediction performance as function of time on each dataset. For each method we monitor on a 10-fold cross-validation the average negative test log-likelihood and prediction error on a hold-out test set as a function of time.

The results are displayed in Fig. 2 for three selected datasets, while the results for the remaining datasets are deferred to appendix A.1. For all datasets we observe that after a few iterations X-GPC is already close to the optimum due to its efficient closed form natural gradient updates. Both the prediction error and test log-likelihood converge around one to two orders of magnitude faster for X-GPC than for SVGPC and EPGPC. Moreover, the performance curves tend to be noisier for SVGPC than for X-GPC and EPGPC. For the datasets HIGGS and IJCNN, EPGPC lead to slightly better final prediction performance, but with the cost of a runtime being up to 4 orders of magnitude slower than X-GPC (approx. 28 hours vs. 9 and 435 seconds, respectively).

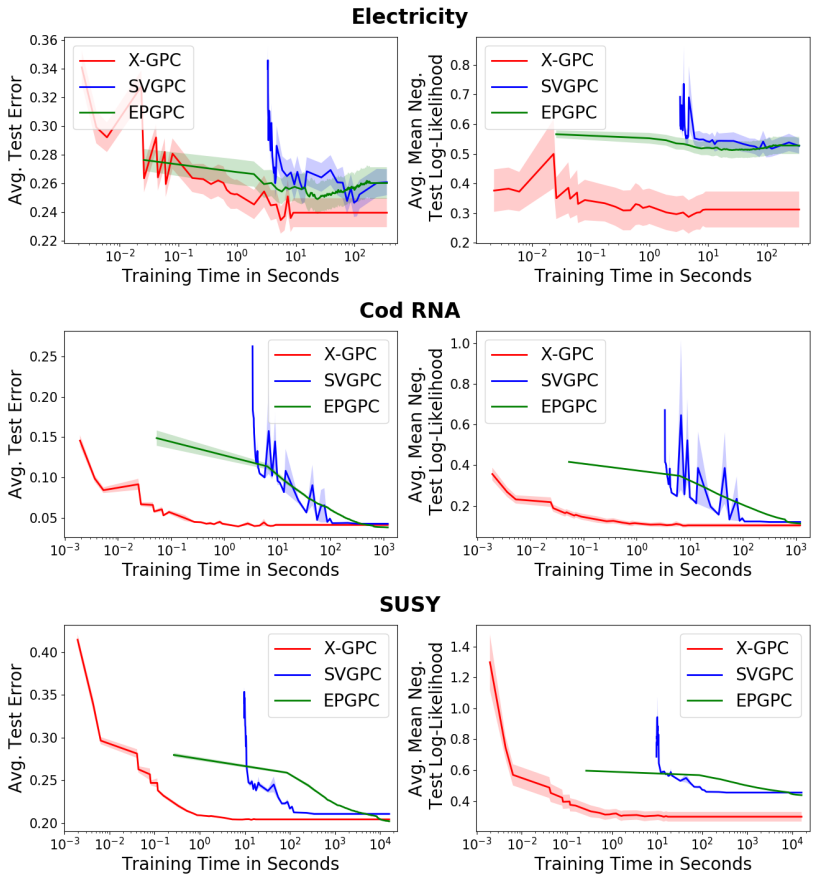


Figure 2: Average negative test log-likelihood and average test prediction error as a function of training time (seconds in a $\log_{10}$ scale) on the datasets Electricity (45,312 points), Cod RNA (343,564 points) and SUSY (5 million points). X-GPC (proposed) reaches values close to the optimum after only a few iterations, whereas SVGPC and EPGPC are one to two orders of magnitude slower.

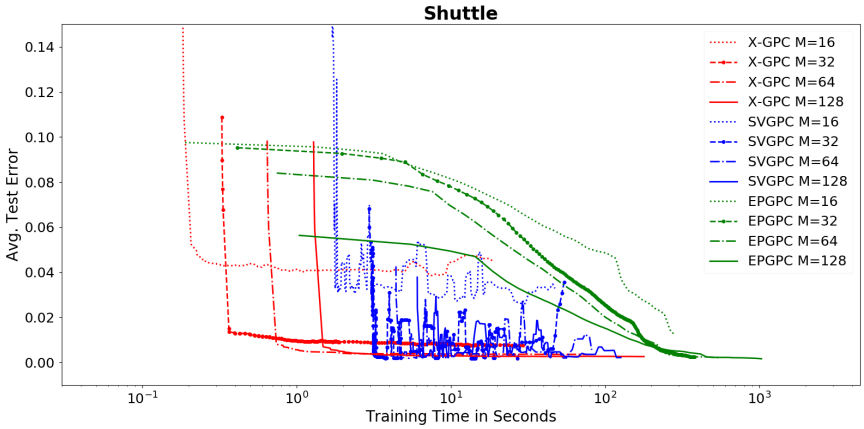


Figure 3: Prediction error as function of training time (on a $\log_{10}$ scale) for the Shuttle dataset. Different numbers of inducing points are considered, $M = 16, 32, 64, 128$. X-GPC (proposed) converges the fastest in all settings of different numbers of inducing points. Using only 32 inducing points is enought for obtaining allmost optimal prediction performance for all methods, but SVGPC becomes instable in settings of less than 128 inducing points.

All three methods are implemented in different programming frameworks: X-GPC in Julia, SVGPC in TensorFlow and EPGPC in R leading to different efficient implementations. However, we find that the main speed-up of our method is due to the efficient natural gradient updates and only marginally related to the usage of a different programming language. To check this we implemented EPGPC also in Julia and obtained similar runtimes. Since SVGPC is part of the highly optimized GPflow package we only used the original implementation.

5.3 Inducing points

We examine the effect of different numbers of inducing points on the prediction performance and run time. For all methods we compare different numbers of inducing points: $M = 16, 32, 64, 128$. For each setting, we perform a 10-fold cross validation on the Shuttle dataset and plot the mean prediction error as function of time. The results are displayed in Fig. 3. We observe that the higher the number of inducing points, the better the prediction performance, but the longer the run time. Throughout all settings of inducing points our method is consistently faster of around one to two orders of magnitude than the competitors. On the Shuttle dataset using only $M = 32$ inducing points is enough and can only be marginally improved by using more inducing point for all methods. However, the performance curves of SVGPC are instable when using less than 128 inducing points.

6 Conclusions

We proposed an efficient Gaussian process classification method that builds on Pólya-Gamma data augmentation and inducing points. The experimental evaluations shows that our method is up to two orders of magnitude faster than the state-of-the-art approach while being competitive in terms of prediction performance. Speed improvements are due to the Pólya-Gamma data augmentation approach that enables efficient second order optimization.

The presented work shows how data augmentation can speed up variational approximation of GPs. Our analysis may pave the way for using data augmentation to derive efficient stochastic variational algorithms also for variational Bayesian models other than GPs. Furthermore, future work may aim at extending the approach to multi-class and multi-label classification.

Acknowledgements We thank Stephan Mandt, James Hensman and Scott W. Linderman for fruitful discussions. This work was partly funded by the German Research Foundation (DFG) awards KL 2698/2-1 and GRK1589/2 and the by the Federal Ministry of Science and Education (BMBF) awards 031L0023A, 01IS18051A.

References

Amari, S., and Nagaoka, H. 2007. *Methods of Information Geometry*. American Mathematical Society.

Amari, S. 1998. Natural grad. works efficiently in learning. *Neural Computation*.

Arthur, D., and Vassilvitskii, S. 2007. k-means++: The advantages of careful seeding. In *Proceedings of the eighteenth annual ACM-SIAM symposium on Discrete algorithms*, 1027–1035. Society for Industrial and Applied Mathematics.

Dezfouli, A., and Bonilla, E. V. 2015. Scalable inference for gaussian process models with black-box likelihoods. In *NIPS*, 1414–1422.

Dragiev, S.; Toussaint, M.; and Gienger, M. 2011. Gaussian process implicit surfaces for shape estimation and grasping. In *Robotics and Automation (ICRA)*, 2845–2850.

Gibbs, M. N., and MacKay, D. J. C. 2000. Variational Gaussian process classifiers. *IEEE Transactions on Neural Networks* 11(6):1458–1464.

Hensman, J., and Matthews, A. 2015. Scalable Variational Gaussian Process Classification. In *AISTATS*.

Hernández-Lobato, D., and Hernández-Lobato, J. M. 2016. Scalable gaussian process classification via expectation propagation. In *AISTATS*.

Hoffman, M. D.; Blei, D. M.; Wang, C.; and Paisley, J. 2013. Stochastic Variational Inference. *Journal of Machine Learning Research*.

Honkela, A.; Raiko, T.; Kuusela, M.; Tornio, M.; and Karhunen, J. 2010. Approximate riemannian conjugate gradient learning for fixed-form variational bayes. *Journal of Machine Learning Research* 11.

Izmailov, P.; Novikov, A.; and Kropotov, D. 2018. Scalable gaussian processes with billions of inducing inputs via tensor train decomposition. In *AISTATS*, 726–735.

Jähnichen, P.; Wenzel, F.; Kloft, M.; and Mandt, S. 2018. Scalable generalized dynamic topic models. In *AISTATS*.

John Walker, S. 2014. *Big data: A revolution that will transform how we live, work, and think*. Taylor & Francis.

Jordan, M. I.; Ghahramani, Z.; Jaakkola, T. S.; and Saul, L. K. 1999. An Introduction to Variational Methods for Graphical Models. *Machine Learning*.

Khan, M. E., and Nielsen, D. 2018. Fast yet simple natural-gradient descent for variational inference in complex models. *Arxiv Preprint*.

Kingma, D. P., and Ba, J. 2014. Adam: A method for stochastic optimization. *CoRR* abs/1412.6980.

Linderman, S. W.; Johnson, M. J.; and Adams, R. P. 2015. Dependent multinomial models made easy: Stick-breaking with the polya-gamma augmentation. In *NIPS*.

Mandt, S.; Wenzel, F.; Nakajima, S.; Cunningham, J. P.; Lippert, C.; and Kloft, M. 2017. Sparse Probit Linear Mixed Model. *Machine Learning Journal*.

Mandt, S.; Hoffman, M.; and Blei, D. 2016. A Variational Analysis of Stochastic Gradient Algorithms. *ICML*.

Maritz, J., and Lwin, T. 1989. Empirical Bayes Methods with Applications. *Monographs on Statistics and Applied Probability.*

Martens, J. 2017. New insights and perspectives on the natural gradient method. *Arxiv Preprint*.

Matthews, A. G. d. G.; van der Wilk, M.; Nickson, T.; Fujii, K.; Boukouvalas, A.; León-Villagrá, P.; Ghahramani, Z.; and Hensman, J. 2017. GPflow: A Gaussian process library using TensorFlow. *Journal of Machine Learning Research*.

Opper, M., and Archambeau, C. 2009. The variational gaussian approximation revisited. *Neural Comput.* 21(3):786–792.

Polson, N. G., and Scott, S. L. 2011. Data augmentation for support vector machines. *Bayesian Anal.*

Polson, N. G.; Scott, J. G.; and Windle, J. 2013. Bayesian inference for logistic models using pólya–gamma latent variables. *Journal of the American Statistical Association* 108(504):1339–1349.

Ranganath, R.; Wang, C.; Blei, D. M.; and Xing, E. P. 2013. An Adaptive Learning Rate for Stochastic Variational Inference. *ICML*.

Rasmussen, C. E., and Williams, C. K. I. 2005. *Gaussian Processes for Machine Learning (Adaptive Computation and Machine Learning)*. The MIT Press.

Salimbeni, H.; Eleftheriadis, S.; and Hensman, J. 2018. Natural gradients in practice: Non-conjugate variational inference in gaussian process models. In *AISTATS*.

Scott, J. G., and Sun, L. 2013. Expectation-maximization for logistic regression. *arXiv preprint arXiv:1306.0040*.

Snelson, E., and Ghahramani, Z. 2006. Sparse GPs using Pseudo-inputs. *NIPS*.

Stein, M. L. 2012. *Interpolation of spatial data: some theory for kriging*. Springer Science & Business Media.

Wainwright, M. J., and Jordan, M. I. 2008. Graphical models, exponential families, and variational inference. *Found. Trends Mach. Learn.* 1–305.

Wenzel, F.; Galy-Fajou, T.; Deutsch, M.; and Kloft, M. 2017. Bayesian nonlinear support vector machines for big data. In *Proceedings of the European Conference on Machine Learning and Principles and Practice of Knowledge Discovery in Databases*.

A Appendix

A.1 Additional performance plots

We show all time vs. prediction performance plots for the datasets presented in table 1 in section section 5.2 which could not be included in the main paper due to space limitations.

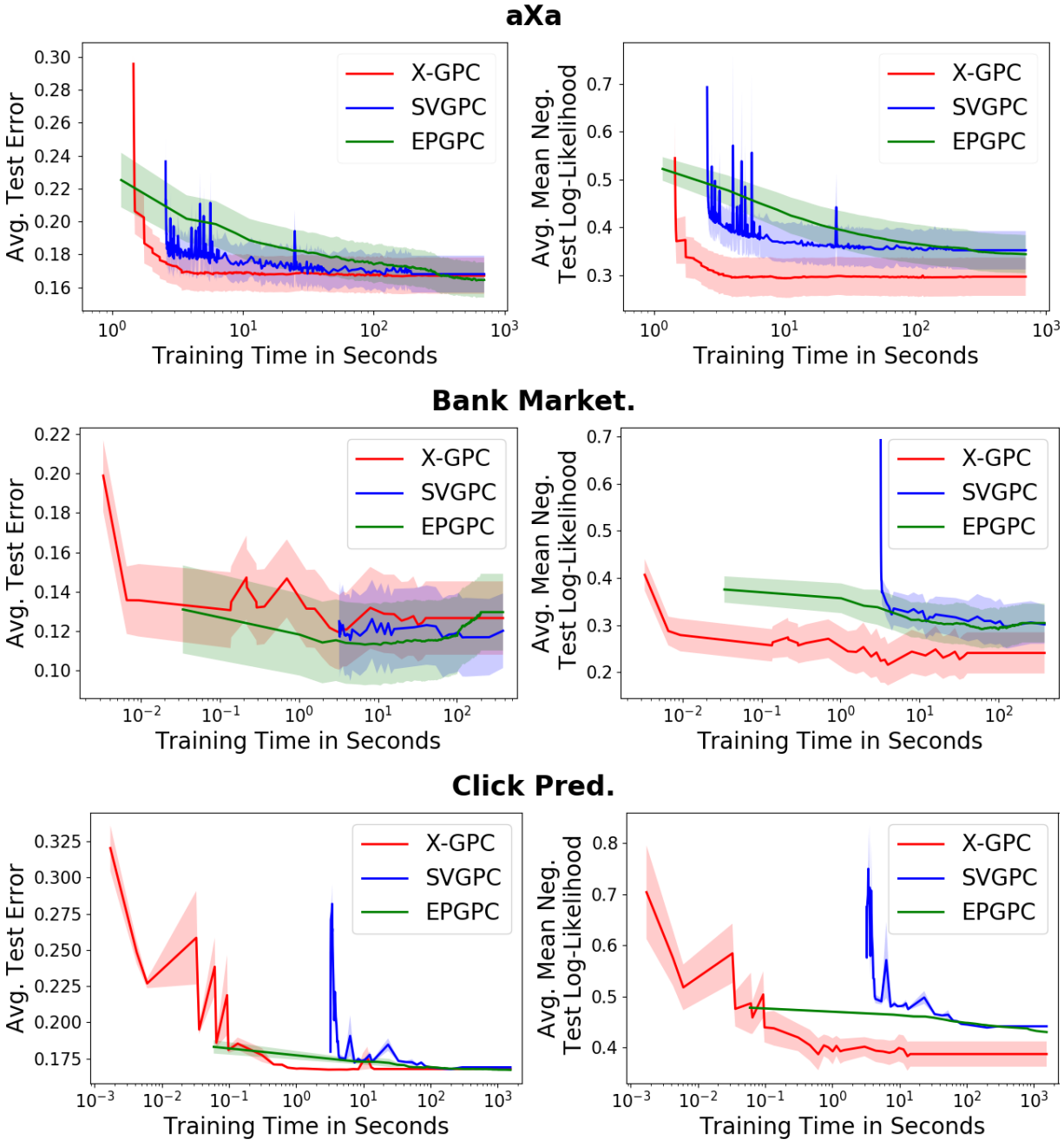


Figure 4: Average negative test log-likelihood and average test prediction error as function of training time measured in seconds (on a $\log_{10}$ scale).

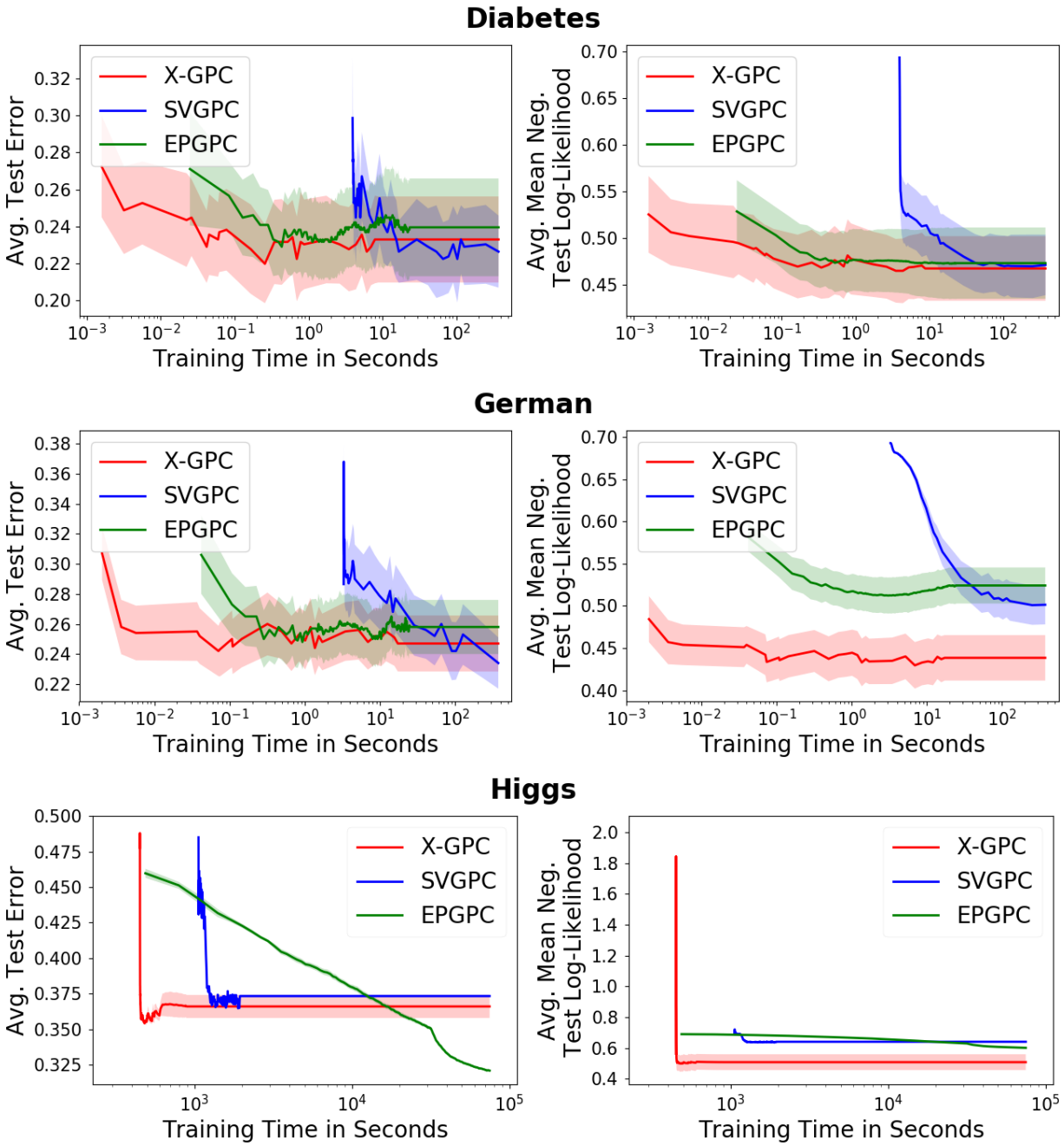


Figure 5: Average negative test log-likelihood and average test prediction error as function of training time measured in seconds (on a $\log_{10}$ scale). For the dataset Higgs, EPGPC exceeded the time budget of 10^5 seconds ($\approx$ 28 h).

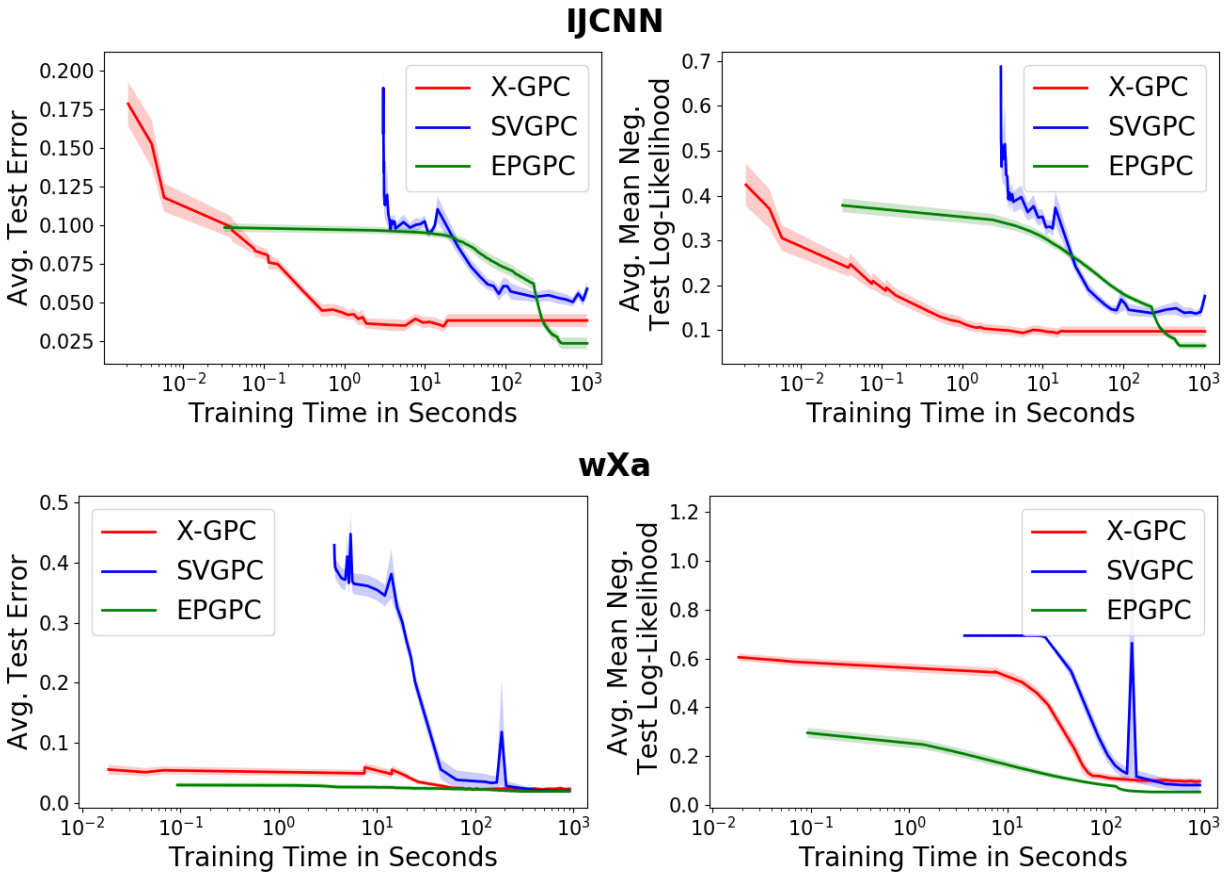


Figure 6: Average negative test log-likelihood and average test prediction error as function of training time measured in seconds (on a $\log_{10}$ scale).

A.2 Variational bound

We provide details of the derivation of the variational bound (10) which is defined as

$$\mathcal{L}(\boldsymbol{c}, \boldsymbol{\mu}, \Sigma) = \mathbb{E}_{p(\boldsymbol{f}|\boldsymbol{u})q(\boldsymbol{u})q(\boldsymbol{\omega})}[\log p(\boldsymbol{y}|\boldsymbol{\omega}, \boldsymbol{f})] - \mathrm{KL}\left(q(\boldsymbol{u}, \boldsymbol{\omega})||p(\boldsymbol{u}, \boldsymbol{\omega})\right),$$

and the family of variational distributions

$$q(\boldsymbol{u}, \boldsymbol{\omega}) = q(\boldsymbol{u}) \prod_i q(\omega_i) = \mathcal{N}(\boldsymbol{u}|\boldsymbol{\mu}, \Sigma) \prod_i \mathrm{PG}(\omega_i|1, c_i).$$

Considering the likelihood term we obtain

$$\begin{aligned}\mathbb{E}_{p(f|u)}[\log p(\boldsymbol{y}|\boldsymbol{\omega}, f)] &\overset{c}{=} \frac{1}{2}\mathbb{E}_{p(f|u)}\left[\boldsymbol{y}^\top \boldsymbol{f} - \boldsymbol{f}^\top \Omega \boldsymbol{f}\right] \\ &= \frac{1}{2}\left(\boldsymbol{y}^\top K_{nm} K_{mm}^{-1} \boldsymbol{u} - \mathrm{tr}(\Omega \widetilde{K}) - \boldsymbol{u}^\top K_{mm}^{-1} K_{mn} \Omega K_{nm} K_{mm}^{-1} \boldsymbol{u}\right).\end{aligned}$$

Computing the expectations w.r.t. to variational distributions gives

$$
\begin{aligned}
&\mathbb{E}_{p(\boldsymbol{f}|\boldsymbol{u})q(\boldsymbol{u})q(\boldsymbol{\omega})}[\log p(\boldsymbol{y}|\boldsymbol{\omega},\boldsymbol{f})] \\
&\quad \overset{\mathsf{c}}{=} \frac{1}{2}\mathbb{E}_{q(\boldsymbol{u})q(\boldsymbol{\omega})}\left[\boldsymbol{y}^\top K_{nm}K_{mm}^{-1}\boldsymbol{u} - \mathrm{tr}(\Omega\widetilde{K}) - \boldsymbol{u}^\top K_{mm}^{-1}K_{mn}\Omega K_{nm}K_{mm}^{-1}\boldsymbol{u}\right] \\
&\quad = \frac{1}{2}\mathbb{E}_{q(\boldsymbol{u})}\left[\boldsymbol{y}^\top K_{nm}K_{mm}^{-1}\boldsymbol{u} - \mathrm{tr}(\Theta\widetilde{K}) - \boldsymbol{u}^\top K_{mm}^{-1}K_{mn}\Theta K_{nm}K_{mm}^{-1}\boldsymbol{u}\right] \\
&\quad = \frac{1}{2}\left[\boldsymbol{y}^\top K_{nm}K_{mm}^{-1}\boldsymbol{\mu} - \mathrm{tr}(\Theta\widetilde{K}) - \mathrm{tr}(K_{mm}^{-1}K_{mn}\Theta K_{nm}K_{mm}^{-1}\Sigma) - \boldsymbol{\mu}^\top K_{mm}^{-1}K_{mn}\Theta K_{nm}K_{mm}^{-1}\boldsymbol{\mu}\right] \\
&\quad = \frac{1}{2}\sum_i \left(y_i\boldsymbol{\kappa}_i\boldsymbol{\mu} - \theta_i\widetilde{K}_{ii} - \theta_i\boldsymbol{\kappa}_i\Sigma\boldsymbol{\kappa}_i^\top - \theta_i\boldsymbol{\mu}^\top\boldsymbol{\kappa}_i^\top\boldsymbol{\kappa}_i\boldsymbol{\mu}\right),
\end{aligned}
$$

where $\theta_i = \mathbb{E}_{p(\omega_i)}[\omega_i] = \frac{1}{2c_i}\tanh\left(\frac{c_i}{2}\right)$, $\Theta = \mathrm{diag}(\boldsymbol{\theta})$ and $\boldsymbol{\kappa}_i = K_{im}K_{mm}^{-1}$.

The Kullback-Leibler divergence between the Gaussian distributions $q(\boldsymbol{u})$ and $p(\boldsymbol{u})$ is easily computed

$$
\mathrm{KL}(q(\boldsymbol{u}))||p(\boldsymbol{u})) \overset{\mathsf{c}}{=} \frac{1}{2}\left(\mathrm{tr}\left(K_{mm}^{-1}\Sigma\right) + \boldsymbol{\mu}^\top K_{mm}^{-1}\boldsymbol{\mu} - \log|\Sigma| + \log|K_{mm}|\right).
$$

The Kullback-Leibler divergence regarding the Pólya-Gamma also can be computed in closed-form. Have $q(\omega_i) = \cosh\left(\frac{c_i}{2}\right)\exp\left(-\frac{c_i^2}{2}\omega_i\right)\mathrm{PG}(\omega_i|1,0)$ and $p(\omega_i) = \mathrm{PG}(\omega_i|1,0)$ we obtain

$$
\begin{aligned}
\mathrm{KL}(q(\boldsymbol{\omega}))||p(\boldsymbol{\omega})) &= \mathbb{E}_{q(\boldsymbol{\omega})}\left[\log q(\boldsymbol{\omega}) - \log p(\boldsymbol{\omega})\right] \\
&= \sum_i \left(\mathbb{E}_{q(\omega_i)}\left[\log\left(\cosh\left(\frac{c_i}{2}\right)\exp\left(-\frac{c_i^2}{2}\omega_i\right)\mathrm{PG}(\omega_i|1,0)\right)\right] - \mathbb{E}_{q(\omega_i)}\left[\log\mathrm{PG}(\omega_i|1,0)\right]\right) \\
&= \sum_i \left(\log\cosh\frac{c_i}{2} - \frac{c_i}{4}\tanh\left(\frac{c_i}{2}\right) + \mathbb{E}_{q(\omega_i)}\left[\log\mathrm{PG}(\omega_i|1,0)\right] - \mathbb{E}_{q(\omega_i)}\left[\log\mathrm{PG}(\omega_i|1,0)\right]\right) \\
&= \sum_i \left(\log\cosh\frac{c_i}{2} - \frac{c_i}{4}\tanh\left(\frac{c_i}{2}\right)\right).
\end{aligned}
$$

Remarkably, the intractable expectations cancel out which would not have been the case if we assumed $\mathrm{PG}(\omega_i|b_i,c_i)$ as variational family. In section 4.2 we have shown that the restricted family $b_i = 1$ contains the optimal distribution.

Summing all terms results in the final lower bound

$$
\begin{aligned}
\mathcal{L}(\boldsymbol{c},\boldsymbol{\mu},\Sigma) \overset{\mathsf{c}}{=} \frac{1}{2}\Bigg(&\log|\Sigma| - \log|K_{mm}|) - \mathrm{tr}(K_{mm}^{-1}\Sigma) - \boldsymbol{\mu}^\top K_{mm}^{-1}\boldsymbol{\mu} + \\
&\sum_i \left\{y_i\boldsymbol{\kappa}_i\boldsymbol{\mu} - \theta_i\widetilde{K}_{ii} - \theta_i\boldsymbol{\kappa}_i\Sigma\boldsymbol{\kappa}_i^\top - \theta_i\boldsymbol{\mu}^\top\boldsymbol{\kappa}_i^\top\boldsymbol{\kappa}_i\boldsymbol{\mu} + c_i^2\theta_i - 2\log\cosh\frac{c_i}{2}\right\}\Bigg).
\end{aligned}
$$

A.3 Variational updates

Local parameters The derivative of the variational bound (10) w.r.t. the local parameter c_i is

$$
\begin{aligned}
\frac{\mathrm{d}\mathcal{L}}{\mathrm{d}c_i} &= \frac{1}{2}\frac{\mathrm{d}}{\mathrm{d}c_i}\left\{\theta_i\left(-\widetilde{K}_{ii} - \boldsymbol{\kappa}_i\Sigma\boldsymbol{\kappa}_i^\top - \boldsymbol{\mu}^\top\boldsymbol{\kappa}_i^\top\boldsymbol{\kappa}_i\boldsymbol{\mu} + c_i^2\right) - 2\log\cosh\frac{c_i}{2}\right\} \\
&= \frac{1}{2}\frac{\mathrm{d}}{\mathrm{d}c_i}\left\{\frac{1}{2c_i}\tanh\left(\frac{c_i}{2}\right)\left(-\widetilde{K}_{ii} - \boldsymbol{\kappa}_i\Sigma\boldsymbol{\kappa}_i^\top - \boldsymbol{\mu}^\top\boldsymbol{\kappa}_i^\top\boldsymbol{\kappa}_i\boldsymbol{\mu} + c_i^2\right) - 2\log\cosh\frac{c_i}{2}\right\} \\
&= \frac{\mathrm{d}}{\mathrm{d}c_i}\left\{\frac{1}{4c_i}\tanh\left(\frac{c_i}{2}\right)\left(\underbrace{-\widetilde{K}_{ii} - \boldsymbol{\kappa}_i\Sigma\boldsymbol{\kappa}_i^\top - \boldsymbol{\mu}^\top\boldsymbol{\kappa}_i^\top\boldsymbol{\kappa}_i\boldsymbol{\mu}}_{:=-A_i}\right) + \frac{c_i}{4}\tanh\left(\frac{c_i}{2}\right) - \log\cosh\frac{c_i}{2}\right\} \\
&= \left(\frac{A_i}{4c_i^2} - \frac{1}{4}\right)\tanh\left(\frac{c_i}{2}\right) - \frac{1}{2}\left(\frac{A_i}{4c_i} - \frac{c_i}{4}\right)\left(1 - \tanh^2(\frac{c_i}{2})\right) \\
&= U(c_i)\left(\frac{c_i}{2}\left(1 - \tanh^2(\frac{c_i}{2})\right) - \tanh\left(\frac{c_i}{2}\right)\right),
\end{aligned}
$$

where $U(c_i) = \frac{\Sigma_{ii}+\mu_i^2}{4c_i^2} - \frac{1}{4}$.

The gradient equals zero in two case. First, in the case $U(c_i) = 0$ which leads to[4]

$$c_i = \sqrt{\widetilde{K}_{ii} + \boldsymbol{\kappa}_i \Sigma \boldsymbol{\kappa}_i^\top + \boldsymbol{\mu}^\top \boldsymbol{\kappa}_i^\top \boldsymbol{\kappa}_i \boldsymbol{\mu}},$$

which is always valid since $\boldsymbol{\kappa}$, Σ and $\widetilde{K}$ are definite positive matrices. The second consists of the right hand side of the product being zero which leads to $c_i = 0$. The second derivative reveals that the first case always corresponds to a maximum and the second case to a minimum.

Global parameters We first compute the Euclidean gradients of the variational bound (10) w.r.t. the global parameters $\boldsymbol{\mu}$ and Σ. We obtain

$$\begin{aligned} \frac{d\mathcal{L}}{d\boldsymbol{\mu}} &= \frac{1}{2}\frac{d}{d\boldsymbol{\mu}}\left(-\boldsymbol{\mu}^\top K_{mm}^{-1}\boldsymbol{\mu} + \boldsymbol{y}^\top \boldsymbol{\kappa}\boldsymbol{\mu} - \boldsymbol{\mu}^\top \boldsymbol{\kappa}^\top \Theta \boldsymbol{\kappa}\boldsymbol{\mu}\right) \\ &= \frac{1}{2}\left(-2K_{mm}^{-1}\boldsymbol{\mu} + \boldsymbol{\kappa}^\top \boldsymbol{y} - 2\boldsymbol{\kappa}^\top \Theta \boldsymbol{\kappa}\boldsymbol{\mu}\right) \\ &= -\left(K_{mm}^{-1} + \boldsymbol{\kappa}^\top \Theta \boldsymbol{\kappa}\right)\boldsymbol{\mu} + \frac{1}{2}\boldsymbol{\kappa}^\top \boldsymbol{y}, \end{aligned} \tag{15}$$

and

$$\begin{aligned} \frac{d\mathcal{L}}{d\Sigma} &= \frac{1}{2}\frac{d}{d\Sigma}\left(\log|\Sigma| - \mathrm{tr}(K_{mm}^{-1}\Sigma) - \mathrm{tr}(\boldsymbol{\kappa}^\top \Theta \boldsymbol{\kappa}\Sigma)\right) \\ &= \frac{1}{2}\left(\Sigma^{-1} - K_{mm}^{-1} - \boldsymbol{\kappa}^\top \Theta \boldsymbol{\kappa}\right). \end{aligned} \tag{16}$$

We now compute the natural gradients w.r.t. natural parameterization of the variational Gaussian distribution, i.e the parameters $\boldsymbol{\eta}_1 := \Sigma^{-1}\boldsymbol{\mu}$ and $\eta_2 = -\frac{1}{2}\Sigma^{-1}$. For a Gaussian distribution, properties of the Fisher information matrix expose the simplification that the natural gradient w.r.t. the natural parameters can be expressed in terms of the Euclidean gradient w.r.t. the mean and covariance parameters. It holds that

$$\widetilde{\nabla}_{(\boldsymbol{\eta}_1,\eta_2)}\mathcal{L}(\eta) = \left(\nabla_\mu \mathcal{L}(\eta) - 2\nabla_\Sigma \mathcal{L}(\eta)\mu,\ \nabla_\Sigma \mathcal{L}(\eta)\right), \tag{17}$$

where $\widetilde{\nabla}$ denotes the natural gradient and ∇ the Euclidean gradient. Substituting the Euclidean gradients (16) and (15) in to equation (17) we obtain the natural gradients

$$\begin{aligned} \widetilde{\nabla}_{\eta_2}\mathcal{L} &= \frac{1}{2}\left(-2\eta_2 - K_{mm}^{-1} - \boldsymbol{\kappa}^\top \Theta \boldsymbol{\kappa}\right) \\ &= -\eta_2 - \frac{1}{2}\left(K_{mm}^{-1} + \boldsymbol{\kappa}^\top \Theta \boldsymbol{\kappa}\right) \end{aligned}$$

and

$$\begin{aligned} \widetilde{\nabla}_{\boldsymbol{\eta}_1}\mathcal{L} &= -\left(K_{mm}^{-1} + \boldsymbol{\kappa}^\top \Theta \boldsymbol{\kappa}\right)\left(-\frac{1}{2}\eta_2^{-1}\boldsymbol{\eta}_1\right) + \frac{1}{2}\boldsymbol{\kappa}^\top \boldsymbol{y} - 2\left(-\eta_2 - \frac{1}{2}\left(K_{mm}^{-1} + \boldsymbol{\kappa}^\top \Theta \boldsymbol{\kappa}\right)\right)\left(-\frac{1}{2}\eta_2^{-1}\boldsymbol{\eta}_1\right) \\ &= \frac{1}{2}\boldsymbol{\kappa}^\top \boldsymbol{y} - \boldsymbol{\eta}_1. \end{aligned}$$

A.4 Natural gradient and coordinate ascent updates

If the full conditional distributions and the corresponding variational distribution belong to the same exponential family it is known in variational inference that "we can compute the natural gradient by computing the coordinate updates in parallel and subtracting the current setting of the parameter" Hoffman et al. (2013). In our setting it is not clear if this relation holds since we do not consider the classic ELBO but a lower bound on it due to (8). Interestingly, the lower bound (8) does not break this property and our natural gradient updates correspond to coordinate ascent updates as we show in the following. Setting the Euclidean gradients and (15) to zero and using the natural parameterization gives

$$\eta_2 = -\frac{1}{2}\Sigma^{-1} = -\frac{1}{2}\left(K_{mm}^{-1} + \boldsymbol{\kappa}^\top \Theta \boldsymbol{\kappa}\right). \tag{18}$$

Setting (16) to zero yields

$$\boldsymbol{\mu} = \frac{1}{2}\left(K_{mm}^{-1} + \boldsymbol{\kappa}^\top \Theta \boldsymbol{\kappa}\right)^{-1}\boldsymbol{\kappa}^\top \boldsymbol{y}.$$

Substituting the update from above (18) and using natural parameterization results in

$$\boldsymbol{\eta}_1 = \frac{1}{2}\boldsymbol{\kappa}^\top \boldsymbol{y}.$$

This shows that using learning rate one in our natural gradient ascent scheme corresponds to employing coordinate ascent updates in the Euclidean parameter space.

[4] We omit the negative solution since $\mathrm{PG}(b, c) = \mathrm{PG}(b, -c)$.

A.5 Variational bound by Gibbs and MacKay

When using the full GP representation in our model and not the sparse approximation, the bound in our model is equal to the bound used by Gibbs and MacKay (2000). We provide a proof in the following.

Applying our variational inference approach to the joint distribution (5) gives the variational bound

$$\begin{aligned}\log p(\boldsymbol{y}|\,\boldsymbol{f}) &\geq \mathbb{E}_{q(\omega)}\left[\log p(\boldsymbol{y}|\,\boldsymbol{f},\boldsymbol{\omega})\right] - \mathrm{KL}(q(\boldsymbol{\omega})|p(\boldsymbol{\omega})) \\ &= \mathbb{E}_{q(\omega)}\left[\frac{1}{2}\boldsymbol{y}^\top\boldsymbol{f} - \frac{1}{2}\boldsymbol{f}^\top\Omega\boldsymbol{f}\right] - n\log(2) - \mathrm{KL}(q(\boldsymbol{\omega})|p(\boldsymbol{\omega})) \\ &= \frac{1}{2}\boldsymbol{y}^\top\boldsymbol{f} - \frac{1}{2}\boldsymbol{f}^\top\Theta\boldsymbol{f} - n\log(2) + \sum_{i=1}^{n}\left(\frac{c_i^2}{2}\theta_i - \log\cosh(c_i/2)\right).\end{aligned}$$

Gibbs and MacKay (2000) employ the following inequality on logit link

$$\sigma(z) \geq \sigma(c)\exp\left(\frac{z-c}{2} - \frac{\sigma(c)-1/2}{2c}(z^2-c^2)\right).$$

Using this bound in the setting of GP classification yields the following lower bound on the log-likelihood,

$$\begin{aligned}\log p(\boldsymbol{y}|\,\boldsymbol{f}) &= \sum_{i=1}^{n}\log\sigma(y_i f_i) \\ &\geq \sum_{i=1}^{n}\left(\log\sigma(c_i) + \frac{y_i f_i - c_i}{2} - \frac{\sigma(c_i)-1/2}{2c_i}((y_i f_i)^2 - c_i^2)\right) \\ &= \sum_{i=1}^{n}\left(-\log\cosh(c_i/2) - \log(2) + \frac{y_i f_i}{2} - \frac{\sigma(c_i)-1/2}{2c_i}(f_i^2 - c_i^2)\right) \\ &= \sum_{i=1}^{n}\left(-\log\cosh(c_i/2) - \log(2) + \frac{y_i f_i}{2} - \frac{1}{4c_i}\tanh(c_i/2)(f_i^2 - c_i^2)\right) \\ &= \sum_{i=1}^{n}\left(-\log\cosh(c_i/2) - \log(2) + \frac{y_i f_i}{2} - \frac{1}{2}\theta_i(f_i^2 - c_i^2)\right) \\ &= \frac{1}{2}\boldsymbol{y}^\top\boldsymbol{f} - \frac{1}{2}\boldsymbol{f}^\top\Theta\boldsymbol{f} - n\log(2) + \sum_{i=1}^{n}\left(\frac{c_i^2}{2}\theta_i - \log\cosh(c_i/2)\right),\end{aligned}$$

where we made use of the fact that $\sigma(x) - 1/2 = \tanh(x/2)/2$. This concludes the proof.

4

Multi-Class Gaussian Process Classification Made Conjugate: Efficient Inference via Data Augmentation

After the binary classification problem, a natural extension is the multi-class classification setting. By drawing inspiration from Donner and Opper [12], we use new augmentations methods to circumvent the problem of a much more complex likelihood function involving multiple latent $\mathcal{GP}$**s**. More specifically, we introduce a new link, the logistic-softmax function. We turn the model into a fully conditionally-conjugate model with three successive augmentations. A thorough analysis is made to compare this new model with other links and approaches, including standard choices like the softmax link.

Note that an extensive discussion about this model is given in Chapter 7 with potential model extensions and solutions to some problems faced in the paper.

Authors:

Théo Galy-Fajou,[1,*], Florian Wenzel,[1,*], Christian Donner,[1] Manfred Opper[1]

[*]Equal Contribution, [1]TU Berlin, Germany, [2]TU Kaiserslautern, Germany

Details:

Type: Conference article Submitted: January 2019

Accepted: May 2019

URL: `http://auai.org/uai2019/proceedings/papers/264.pdf`

Conference: UAI 2019

Contributions:

For an explanation of the terms see the Contributor Roles Taxonomy (CReditT)

	T.G-F.	F.W.	C.D.	M.O.
Conceptualization	✓		✓	✓
Methodology	✓			
Formal Analysis	✓	✓	✓	✓
Implementation	✓			
Investigation	✓			
Writing - Original Draft	✓	✓	✓	
Writing - Review & Editing	✓	✓	✓	✓
Supervision				✓
Funding Acquisition				✓

Multi-Class Gaussian Process Classification Made Conjugate: Efficient Inference via Data Augmentation

Théo Galy-Fajou *
TU Berlin
Germany

Florian Wenzel *
TU Kaiserslautern
Germany

Christian Donner
TU Berlin
Germany

Manfred Opper
TU Berlin
Germany

Abstract

We propose a new scalable multi-class Gaussian process classification approach building on a novel modified softmax likelihood function. The new likelihood has two benefits: it leads to well-calibrated uncertainty estimates and allows for an efficient latent variable augmentation. The augmented model has the advantage that it is conditionally conjugate leading to a fast variational inference method via block coordinate ascent updates. Previous approaches suffered from a trade-off between uncertainty calibration and speed. Our experiments show that our method leads to well-calibrated uncertainty estimates and competitive predictive performance while being up to two orders faster than the state of the art.

1 INTRODUCTION

In real-world decision making systems, it is important that classification methods do not only provide accurate predictions, but also indicate when they are likely to be incorrect. Calibrated confidence estimates are important in many application domains such as self driving cars (Bojarski et al., 2016), medical diagnosis (Caruana et al., 2015) and speech recognition (Xiong et al., 2016).

In multi-class classification tasks, modern deep neural networks achieve state-of-the-art accuracies but often suffer from bad calibration (Guo et al., 2017). Gaussian process (GP) models provide an attractive alternative approach to multi-class classification problems.

Due to the Bayesian treatment of uncertainty, GPs have the advantage of leading to well-calibrated uncertainty estimates (Williams and Barber, 1998; Rasmussen and Williams, 2005). Furthermore, GP models become more expressive as the number of data points grows and allow for incorporating prior knowledge by using different kernel functions. However, inference in multi-class GP classification models is challenging.

In the easier setting of *binary* classification, GPs can be applied to big datasets using variational inference methods (Hensman and Matthews, 2015; Wenzel et al., 2019). This is possible because the expectation of generic log-likelihoods in the variational objective (the so-called ELBO) over the variational distribution (typically a Gaussian) reduces to univariate integrals which can be performed in an efficient way by using numerical quadrature methods. The optimization of the variational objective can then be achieved by stochastic gradient methods involving mini-batches. A further speedup of such methods is possible by the application of natural gradient techniques (Salimbeni et al., 2018).

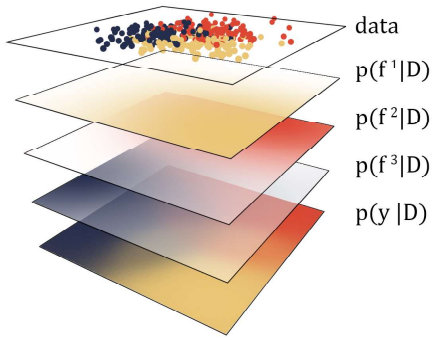


Figure 1: In a GP multi-class classification model, each class density is modeled by an individual GP $p(f^c|D)$. For predictions $p(y|D)$, the latent GPs are marginalized out.

The *multi-class* problem is more complicated because it involves not only one latent GP, but one GP for each class. In the common multi-class likelihoods, as e.g. the

*Equal contribution. Contact: galy-fajou@tu-berlin.de.

softmax function, the GPs are coupled. This leads to complicated multivariate integrals which make a direct application of variational inference techniques intractable. Previous inference methods for the softmax model rely on approximations and do not scale (Williams and Barber, 1998; Chai, 2012).

To tackle this issue, Hernández-Lobato et al. (2011) propose an alternative to the softmax, the *robust-max* likelihood. This likelihood simplifies the problem by focusing mainly on the maximal latent GP and discarding information of the other less likely classes. The model is robust against outliers and often yields good classification accuracy. However, it sacrifices the gradual response of the traditional softmax for an all-or-nothing criterion leading to bad uncertainty quantification.

In problems with well separated classes and a few outliers, the robust-max likelihood is an excellent choice, while in problems with overlapping classes a gradual classification criterion is more desirable (Xiong et al., 2010). In this work, we introduce a novel likelihood, the *logistic softmax* likelihood, which combines the best of both worlds. It has a gradual classification criterion similar to the traditional softmax, but on the other hand also enables fast inference.

We propose an augmentation approach that renders the model conditionally conjugate. Inference in the augmented model is much easier. We derive a fast *variational inference* algorithm based on closed-form updates. Our inference approach is faster and more stable than the state of the art since it uses efficient block coordinate ascent updates and does not rely on sampling.

Alternatively, the conditionally conjugate form of the augmented model directly leads to another inference strategy. If we are willing to pay more computation time, we obtain *exact samples* from the true posterior by a Gibbs sampling scheme. Our main contributions are as follows:

- We introduce a new multi-class GP classification model building on a modification of the softmax likelihood function. By applying a variable augmentation approach, we render the model conditionally conjugate.
- We propose an efficient stochastic variational inference scheme which is based on block coordinate-ascent updates. Unlike in previous work, all updates are given in closed-form and do not rely on numerical quadrature methods or sampling.
- Our method scales to datasets with many data points and a large number of classes. The experiments show that our method is faster than the state-of-the-art while leading to competitive prediction performance.
- We solve the calibration issue of the robust-max likelihood as our model leads to much better uncertainty quantification.

The paper is structured as follows. Section 2 introduces the problem of multi-class GP classification and reviews related work. In Section 3 we introduce the new model and present a data augmentation strategy that renders the model conditionally conjugate. In Section 4 we present an efficient inference algorithm. We show experimental results in Section 5. Finally, Section 6 concludes and lays out future research directions. Our code is included in a Julia package[1].

2 BACKGROUND AND RELATED WORK

We begin our review by introducing the multi-class GP classification model. Related work can be grouped into approaches that consider alternative likelihood functions or apply data augmentation strategies.

Multi-class GP classification. We consider a dataset of N data points $\boldsymbol{X} = (\boldsymbol{x}_1, \dots, \boldsymbol{x}_N)$ with labels $\boldsymbol{y} = (y_1, \dots, y_N)$, where $y_i \in \{1, \dots, C\}$ and C is the total number of classes. The multi-class GP classification model consists of a latent GP prior for each class $\boldsymbol{f} = (f^1, \dots, f^C)$, where $f^c \sim \mathrm{GP}(0, k^c)$ and k^c is the corresponding kernel function. The labels are modeled by a categorical likelihood

$$p(y_i = k | \boldsymbol{x}_i, \boldsymbol{f}_i) = g^k(\boldsymbol{f}(\boldsymbol{x}_i)), \tag{1}$$

where $g^k(f)$ is a function that maps the real vector of the GP values to a probability vector.

The most common way to form a categorical likelihood is through the softmax transformation

$$p(y_i = k | \boldsymbol{f}_i) = \frac{\exp\left(f_i^k\right)}{\sum_{c=1}^{C} \exp\left(f_i^c\right)}, \tag{2}$$

where we use the shorthand $f_i^c = f^c(x_i)$ and for the sake of clarity we omit the conditioning on x_i.

There have been several early works addressing multi-class GP classification with a softmax likelihood (Williams and Barber, 1998; Kim and Ghahramani, 2006; Chai, 2012; Riihimäki et al., 2013). Nevertheless, these methods do not scale well with the number of data points. Izmailov et al. (2018) use tensor train decomposition to use high numbers of inducing points but do not provide efficient closed-form updates.

[1] https://github.com/theogf/AugmentedGaussianProcesses.jl

The robust-max likelihood. Recently, there have been advances to scale multi-class GP classification to big datasets by changing the likelihood. Hernández-Lobato et al. (2011) propose the *robust-max* likelihood

$$p(y = k|\boldsymbol{f}) = (1 - \epsilon) \prod_{c \neq y}^{C} \Theta\left(f^k - f^c\right) + \frac{\epsilon}{C}, \quad (3)$$

where ϵ is the probability of a labeling error, and Θ is the Heaviside function. This likelihood simplifies the problem as it leads to a decoupling of the latent GPs.

Originally, the authors propose an expectation propagation (EP) based approach which only scales to small datasets. Hensman et al. (2015) and Salimbeni et al. (2018) scale this model to big datasets employing a variational inference approach but rely on numerical quadrature. As we show later, this likelihood has the big disadvantage of leading to poor confidence calibration.

The Heaviside likelihood. Villacampa-Calvo and Hernández-Lobato (2017) build on the Heaviside likelihood

$$p(y = k|\boldsymbol{f}) = \prod_{c \neq y}^{C} \Theta\left(f^k - f^c\right), \quad (4)$$

where Θ is again the Heaviside function. The authors propose a scalable expectation propagation approach but have to make approximations on the likelihood. The inference is still slow and the applicability to big datasets is limited.

Data augmentation. Other approaches consider probabilistic data augmentation. Wenzel et al. (2019) propose an augmentation approach for binary GP classification leading to a conditionally conjugate model, but are limited to the binary classification setting. Linderman et al. (2015) consider data augmentation for multinomial likelihoods but focus on sampling. The approach has the disadvantage of breaking the symmetry between the classes and is limited to small datasets. Polson et al. (2013) propose conditionally conjugate Pólya-Gamma augmentation for the softmax likelihood (extended by Češnovar and Štrumbelj (2017) to GPU support) which is suitable for sampling but cannot be used for obtaining an efficient variational inference algorithm since the ELBO is intractable. Girolami and Rogers (2006) propose an augmentation strategy to multinomial probit regression but does not scale. Ruiz et al. (2018) propose an augmentation approach for enabling subsampling of classes for parametric models with categorical likelihoods. The approach is limited to parametric models and cannot be applied to GP models.

3 MULTI-CLASS GAUSSIAN PROCESS CLASSIFICATION

We formulate a multi-class GP classification model which leads to well calibrated confidences and is amenable to fast inference. We define a new likelihood function, termed the *logistic-softmax* , which shares the good prediction properties of the softmax. But in addition, it has the advantage that it allows for a data augmentation approach which renders the model conditionally conjugate. The augmented posterior can then be efficiently approximated by a structured mean-field variational inference method resulting in a fast algorithm with closed-form updates.

3.1 THE LOGISTIC-SOFTMAX GP MODEL

We consider the multi-class GP classification model as described in eq. 1. Different functions g for mapping real vectors to probability vectors that have been considered in literature include the softmax (eq. 2), the multinomial probit (Albert and Chib, 1993), the robust-max likelihood (eq. 3) and the Heaviside likelihood (eq. 4).

In this work, we propose the *logistic-softmax :*

$$p(y_i = k|\boldsymbol{f}_i) = \frac{\sigma\left(f_i^k\right)}{\sum_{c=1}^{C} \sigma\left(f_i^c\right)}, \quad (5)$$

where $\sigma(z) = (1 + \exp(-z))^{-1}$ is the logistic function. Our likelihood is a modified version of the softmax likelihood which replaces the inner exponential functions by logistic functions. Alternatively, it can be interpreted as the standard softmax applied to a non-linearly transformed GP, i.e. $p(y_i|\boldsymbol{f}_i) = \text{softmax}(\log \sigma(\boldsymbol{f}_i))$. The likelihood reduces to the binary logistic likelihood for $C = 2$.

In the following section we derive a three steps augmentation scheme, where we (i) decouple the GP latent variables f_i^k in the denominator by the introduction of a set of auxiliary λ-variables, (ii) further simplify the model likelihood by introducing Poisson random variables, and finally (iii) use a Pólya–Gamma representation of the sigmoid function (Polson et al., 2013) to achieve the desired conjugate representation of the model.

3.2 CONJUGATE AUGMENTATION

We expand the logistic-softmax likelihood (5) by three data augmentation steps leading to a conditionally conjugate model. The final model is displayed in Figure 2. In the following we present the augmentations.

Augmentation 1: Gamma augmentation. To remedy the intractable normalizer term we make use of the integral

identity $\frac{1}{z} = \int_0^\infty \exp(-\lambda z) d\lambda$ and express the likelihood (5) as

$$p(y_i = k|\boldsymbol{f}_i) = \frac{\sigma\left(f_i^k\right)}{\sum_{c=1}^C \sigma\left(f_i^c\right))} = \sigma\left(f_i^k\right) \int_0^\infty \exp\left(-\lambda_i \sum_{c=1}^C \sigma\left(f_i^c\right)\right) d\lambda_i.$$

This augmentation is well known in the Gibbs sampling community to deal with intractable normalization constants (see e.g. Walker (2011)) but is not often used in the setting of variational inference. By interpreting λ_i as an additional latent variable we obtain the augmented likelihood

$$p(y_i = k|\boldsymbol{f}_i, \lambda_i) = \sigma(f_i^k) \prod_{c=1}^C \exp\left(-\lambda_i \sigma(f_i^c)\right), \quad (6)$$

and we impose the improper prior $p(\lambda_i) \propto \mathbb{1}_{[0,\infty)}(\lambda_i)$. The improper prior is not problematic since it leads to a proper complete conditional distribution as we will see in the end of the section.

Augmentation 2: Poisson augmentation. We rewrite the exponential factors in (6) based on the moment generation function of the Poisson distribution $\mathrm{Po}(\cdot|\lambda)$ which is

$$\exp(\lambda(z-1)) = \sum_{n=0}^\infty z^n \mathrm{Po}(n|\lambda).$$

Using $z = \sigma(-f)$ and the fact that $\sigma(f) = 1 - \sigma(-f)$ we rewrite the exponential factors as

$$\exp\left(-\lambda_i \sigma(f_i^c)\right) = \exp\left(\lambda_i(\sigma(-f_i^c) - 1)\right) = \sum_{n_i^c=0}^\infty (\sigma(-f_i^c))^{n_i^c} \mathrm{Po}(n_i^c|\lambda_i),$$

which leads to the augmented likelihood

$$p(y_i = k|\boldsymbol{f}_i, \lambda_i, \boldsymbol{n}_i) = \sigma(f_i^k) \prod_{c=1}^C (\sigma(-f_i^c))^{n_i^c}, \quad (7)$$

where $\boldsymbol{n}_i = (n_i^1, \ldots, n_i^C)$ and the augmented Poisson variables are distributed as $p(n_i^c|\lambda_i) = \mathrm{Po}(n_i^c|\lambda_i)$, see e.g. Donner and Opper (2017, 2018). Note that this augmentation is only possible since the transformation on f_i^c is bounded, hence the need for a modified likelihood.

Augmentation 3: Pólya-Gamma augmentation. In the last augmentation step, we aim for a Gaussian representation of the sigmoid function. The Pólya-Gamma representation (Polson et al., 2013) allows for rewriting the sigmoid function as a scale mixture of Gaussians

$$\sigma(z)^n = \int_0^\infty 2^{-n} \exp\left(\frac{nz}{2} - \frac{z^2}{2}\omega\right) \mathrm{PG}(\omega|n, 0), \quad (8)$$

where $\mathrm{PG}(\omega|n, b)$ is a Pólya-Gamma distribution. Pólya-Gamma variables are well suited for augmentations since the moments are known analytically and an efficient sampler exists (Polson et al., 2013). By applying this augmentation to (7) we obtain

$$p(y_i = k|\boldsymbol{f}_i, \lambda_i, \boldsymbol{n}_i, \boldsymbol{\omega}_i) = \prod_{c=1}^C 2^{-(y_i'^c + n_i^c)} \exp\left(\frac{(y_i'^c - n_i^c) f_i^c}{2} - \frac{(f_i^c)^2}{2}\omega_i^c\right), \quad (9)$$

where $\boldsymbol{\omega}_i = (\omega_i^1, \ldots, \omega_i^C)$ are Pólya-Gamma variables with distributions

$$p(\boldsymbol{\omega}_i|\boldsymbol{n}_i, y_i) = \prod_{c=1}^C \mathrm{PG}(\omega_i^c|y_i'^c + n_i^c, 0),$$

where $\boldsymbol{y}'$ is an $N \times C$-dimensional one-hot encoding of the labels , i.e. $y_i'^c$ is 1 if $y_i = c$, and 0 otherwise. Details are deferred to appendix A.1.

Realizing that (9) has a Gaussian form with respect to $\boldsymbol{f}_i$ we achieved our goal of a conjugate representation of the latent GPs. As we will show in the next paragraph the model is also conditionally conjugate for the augmented variables.

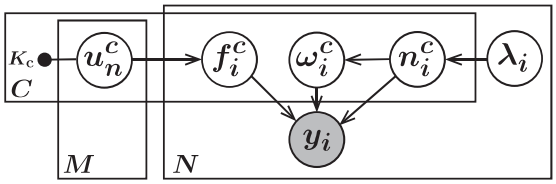


Figure 2: The final augmented model as presented in Section 3.2. Shaded circles represent observable variables, empty circles latent variables and dots hyperparameters.

The final model. The effort of the augmentations finally pays off as the final augmented model is now tractable and the complete conditional distributions are given in closed-form.

The complete conditionals of the GPs $\boldsymbol{f}^c$ are

$$p(\boldsymbol{f}^c \mid \boldsymbol{y}, \boldsymbol{\omega}^c, \boldsymbol{n}^c) = \mathcal{N}\left(\boldsymbol{f}^c \mid \frac{1}{2} A^c (\boldsymbol{y}'^c - \boldsymbol{n}^c), A^c\right),$$

where the conditional covariance matrix is given by $A^c = \left(\mathrm{diag}(\boldsymbol{\omega}^c) + K_c^{-1}\right)^{-1}$ and K_c is the kernel matrix of the

GP $\boldsymbol{f}^c$. For the conditional distribution of $\boldsymbol{\lambda}$ we get

$$p(\lambda_i \mid \boldsymbol{n}_i) = \mathrm{Ga}\left(\lambda_i \mid 1 + \sum_{c=1}^{C} n_i^c, C\right),$$

where $\mathrm{Ga}(\cdot|a, b)$ denotes a gamma distribution with shape parameter a and rate parameters b. The improper prior on λ_i does not impose an issue since the complete conditional distribution is proper.

For the Poisson variables $\boldsymbol{n}$, we get

$$p(n_i^c \mid f_i^c, \lambda_i) = \mathrm{Po}\left(n_i^c \mid \lambda_i \sigma(f_i^c)\right),$$

Finally, for the Pólya-Gamma variables $\boldsymbol{\omega}$ the complete conditional distributions are

$$p(\omega_i^c \mid n_i^c, f_i^c, y_i) = \mathrm{PG}\left(\omega_i^c \mid {y'}_i^c + n_i^c, |f_i^c|\right).$$

4 INFERENCE

We derive a variational approximation of the posterior of the augmented model (9). In the following we develop an efficient stochastic variational inference (SVI) algorithm that is based on closed-form block coordinate ascent updates. Our method allows both for subsampling of data points and of outcomes (classes) scaling to datasets with a large number of data points and a large number of classes.

4.1 VARIATIONAL APPROXIMATION

To scale our model to big datasets, we approximate the latent GPs $\boldsymbol{f}^c$ by *sparse GPs* building on *inducing points*. For each GP $\boldsymbol{f}^c$, we introduce M inducing points $\boldsymbol{u}^c$ and connect the GP values with the inducing points via the joint prior distribution $p(\boldsymbol{f}^c, \boldsymbol{u}^c)$ given in Titsias (2009). Details on variational sparse GP approximations can be found in Titsias (2009); Hensman et al. (2013).

We approximate the posterior distribution of the latent sparse GPs $\boldsymbol{u}$ and the augmented variables $\boldsymbol{\lambda}, \boldsymbol{n}, \boldsymbol{\omega}$ by assuming the following structure of the variational distribution $q(\boldsymbol{u}, \boldsymbol{\lambda}, \boldsymbol{n}, \boldsymbol{\omega}) = q(\boldsymbol{u}, \boldsymbol{\lambda}) q(\boldsymbol{n}, \boldsymbol{\omega})$. Note that the only assumption on the variational posterior is the decoupling of two groups of variables. Since our model is conditionally conjugate, the family of the optimal variational distribution can be easily determined by averaging the complete conditionals in log-space (Blei et al., 2017). From the above decoupling assumption, it follows that the optimal variational posterior has a factorizing form $q(\boldsymbol{u}, \boldsymbol{\lambda}, \boldsymbol{n}, \boldsymbol{\omega}) = q(\boldsymbol{u}) q(\boldsymbol{\lambda}) q(\boldsymbol{\omega}, \boldsymbol{n})$ and the factors are

$$q(\boldsymbol{u}) = \prod_c \mathcal{N}(\boldsymbol{u}^c | \boldsymbol{\mu}^c, \Sigma^c), \; q(\boldsymbol{\lambda}) = \prod_i \mathrm{Ga}(\lambda_i | \alpha_i, \beta_i),$$
$$q(\boldsymbol{\omega}, \boldsymbol{n}) = \prod_{i,c} \mathrm{PG}(\omega_i^c | {y'}_i^c + n_i^c, b_i^c) \mathrm{Po}(n_i^c | \gamma_i^c),$$

where $\boldsymbol{\mu}^c$, Σ^c, α_i, β_i, b_i^c, γ_i^c, for all $i \in \{1, \ldots, N\}$ and $c \in \{1, \ldots, C\}$ are the *variational parameters*. The variational parameters are optimized by a coordinate ascent scheme outlined in Section 4.2. Finally, the approximate posterior of the sparse GPs $q^*(\boldsymbol{u})$ can be used to obtain an approximate posterior of the original latent GPs $\boldsymbol{f}$ by $q^*(\boldsymbol{f}) := \int p(\boldsymbol{f}|\boldsymbol{u}) q(\boldsymbol{u}) d\boldsymbol{u}$ which is given in closed-form (see e.g., Hensman and Matthews, 2015).

4.2 INFERENCE METHOD

Building on the conditionally conjugate representation of our model deriving efficient variational parameter updates is straightforward. We implement the classic SVI algorithm described by Hoffman et al. (2013), which builds on block coordinate ascent updates. We iteratively optimize each factor of the variational distribution, while holding the others fixed. The variational parameters of each factor are directly set to the optimal value given the other parameters.

We compute the block coordinate ascent (CAVI) updates in closed-form by averaging the parameters of each complete conditional in log space (Blei et al., 2017) and details are deferred to appendix A.2. When using minibatches of the data, each global variational parameter (i.e. $\boldsymbol{\mu}^c$ and Σ^c) is updated using a convex combination of the old parameter and the CAVI update, which corresponds to a natural gradient ascent scheme (Hoffman et al., 2013). Remarkably, the negative ELBO in our augmented model is convex in the global parameters (see appendix A.5 for the proof). Therefore, our algorithm is ensured to converge to the global optimum (Hoffman et al., 2013). The inference algorithm is summarized in Alg. 1 and its complexity is $\mathcal{O}(CM^3)$.

Extreme classification. When the number of possible outcomes (classes) C is very large, using probabilistic multi-class models becomes generally computationally expensive as the likelihood (categorical distribution) scales linearly with the number of classes. Using large categorical distributions is a challenging problem (Ruiz et al., 2018; Titsias, 2016).

With a slight modification, our method can deal with an extreme classification setting (large number of classes). In our augmentation, the GPs in the normalizer term are decoupled and allow for subsampling of the classes. This reduces the complexity to $\mathcal{O}(M^3)$, i.e. being independent of the number of classes. We provide details in appendix A.3. This approach is especially useful when using shared hyperparameters among the class specific latent GPs.

Algorithm 1 Conjugate multi-class Gaussian process classification

Input: data $\boldsymbol{X}$,$\boldsymbol{y}$, minibatch size $|\mathcal{S}|$
Output: variational posterior GPs $p(u^c|\mu^c, \Sigma^c)$
Set the learning rate schedules ρ_t, ρ_t^h appropriately
Initialize all variational parameters and hyperparameters
Select M inducing points locations (e.g. kMeans)
for iteration $t = 1, 2, \ldots$ **do**
  `# Sample minibatch:`
  Sample a minibatch of the data $\mathcal{S} \subset \{1, \ldots, N\}$
  `# Local variational updates`
  for $i \in \mathcal{S}$ **do**
    Update $(\alpha_i, \boldsymbol{\gamma}_i)$ (Eq. 12,13)
    for each class c **do**
      Update b_i^c (Eq. 14)
    end for
  end for
  `# Global variational GP updates`
  for each class c **do**
    $\mu^c \leftarrow (1 - \rho_t)\mu^c + \rho_t \hat{\mu}^c$ (Eq. 15)
    $\Sigma^c \leftarrow (1 - \rho_t)\Sigma^c + \rho_t \hat{\Sigma}^c$ (Eq. 16)
  end for
  `# Hyperparameter updates`
  Gradient step $h \leftarrow h + \rho_t^h \nabla_h \mathcal{L}$
end for

Predictions. The posterior distribution of the latent function $p(f_\star^c|x_\star, \boldsymbol{y})$ at a new test point $x_\star$ is approximated by

$$q(f_\star^c|x^\star, \boldsymbol{y}) = \int p(f_\star^c|\boldsymbol{u}^c) q(\boldsymbol{u}^c) d\boldsymbol{u} = \mathcal{N}\left(f_\star^c|{\mu_\star}^c, {\sigma_\star^2}^c\right),$$

where the mean is $\mu_\star^c = {K_{\star m}}^c {K_{mm}^{-1}}^c \boldsymbol{\mu}^c$ and the variance $\sigma_\star^{2c} = {K_{\star\star}}^c + {K_{\star m}}^c {K_{mm}^{-1}}^c (\Sigma^c {K_{mm}^{-1}}^c - I) {K_{m\star}}^c$. The matrix $K_{\star m}$ denotes the kernel matrix between the test point and the inducing points and $K_{\star\star}$ the kernel value of the test point. The final approximate predictive distribution of a test label is

$$p(y = k|x_\star, \boldsymbol{y}) \approx \int p(y = k|\boldsymbol{f}_\star) \prod_{c=1}^{C} q(f_\star^c|x^\star, \boldsymbol{y}) d\boldsymbol{f}^\star,$$

where $p(y = k|\boldsymbol{f}_\star)$ is the logistic-softmax likelihood. This is a C-dimensional analytically intractable integral. We approximate it by Monte Carlo integration. For faster convergence, the random samples can be replaced by Quasi-Monte Carlo sequences (Owen, 1998; Buchholz et al., 2018). Finally, a point is classified by the highest predictive likelihood, $y_i^* = \arg\max_{c \in C} p(y_i = c \mid \boldsymbol{f})$.

Optimization of the hyperparameters. We select the optimal kernel hyperparameters by maximizing the marginal likelihood $p(y|h)$, where h denotes the set of hyperparameters (this approach is called empirical Bayes (Maritz and Lwin, 1989)). We follow an approximate approach and optimize the fitted variational lower bound $\mathcal{L}(h)$ as a function of h by alternating between optimization steps w.r.t. the variational parameters and the hyperparameters (Mandt et al., 2016).

4.3 GIBBS SAMPLING

Since our augmented model is conditionally conjugate we can directly derive a Gibbs sampling scheme. In order to sample from the *exact posterior*, we alternate between drawing a sample from each complete conditional distributions. The augmented variables are naturally marginalized out and asymptotically, the latent GP samples will be from the true posterior.

5 EXPERIMENTS

In this section we empirically answer the following questions:

- What is the effect of using the softmax, logistic-softmax, robust-max and Heaviside likelihood on predictive performance and calibration quality? (Section 5.1)
- How does the augmentation affect the predictive performance? (Section 5.2)
- How does our method perform compared to other state-of-the-art GP based multi-class classification methods? (Section 5.4)

In all experiments we use a squared exponential covariance function with automatic relevance determination (ARD): $k(\boldsymbol{x}, \boldsymbol{x'}) = \eta \exp\left(-\sum_{d=1}^{D} \frac{(x_d - x'_d)^2}{2l_d^2}\right)$, where we set the initial variance η to 1 and the length scales $\boldsymbol{l}$ are initialized to the median of the pairwise distance matrix of the data. The hyperparameters are optimized using Adam (Kingma and Ba, 2015). We use a collection of datasets from the LIBSVM repository[2]. Every dataset has been normalized to mean 0 and variance 1. For each method, we use 200 inducing points, unless stated otherwise. The initial inducing points locations are determined by the kmeans++ algorithm (Arthur and Vassilvitskii, 2007). We find that fixing the locations while training gives good results. We use a mini-batch size of 200 and all experiments are performed on a single CPU.

[2] https://www.csie.ntu.edu.tw/~cjlin/libsvmtools/datasets/multiclass.html

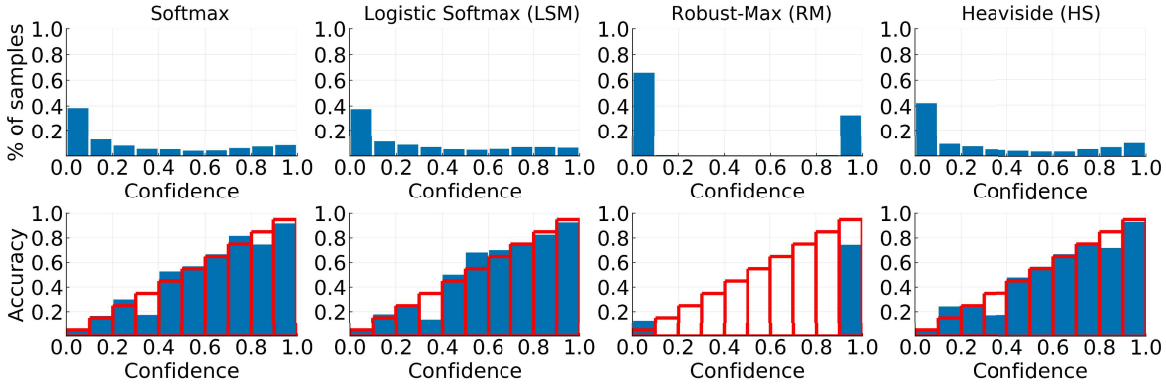


Figure 3: *Likelihood comparison:* Confidence histograms (top) and reliability diagrams (bottom) for four different likelihood models. The robust-max model always predicts with probability either close to one or close to zero leading to a poor confidence calibration.

5.1 LIKELIHOODS COMPARISON

We begin the experiments by investigating the effect of using different likelihood functions. We compare our novel logistic-softmax (eq. 5), the softmax (eq. 2), the robust-max (eq. 3) and the Heaviside likelihood (eq. 4). For each model we employ variational inference to obtain an approximate posterior. In this experiment, no augmentation is used and the gradients are estimated by sampling.

To investigate uncertainty calibration, we create seven different toy datasets of 500 points with three classes. The data is generated from a mixture of Gaussians model with different variances σ^2. For $\sigma^2 = 0$, the classes are sharply separated and for $\sigma^2 = 1$, the classes highly overlap and are almost indistinguishable.

See appendix A.4 for a visualization of the decision boundaries of the different methods. In Figure 4 we plot test error, negative log-likelihood and calibration error as function of the noise in the data. The (expected) calibration error is a summary statistic of calibration and is computed by the expectation between confidence and accuracy in the reliability diagram (c.f. Guo et al., 2017).

For datasets where the classes are sharply separated (small σ^2), all models perform similarly. But for datasets where classes overlap (high σ^2), the robust-max performs poorly due to bad uncertainty calibration.

In Figure 3 we show the confidence histograms and reliability diagrams for one dataset ($\sigma^2 = 0.5$). The diagrams are generated according to Naeini et al. (2015); Guo et al. (2017) – the reliability diagram displays the accuracy as function of confidence (a perfectly calibrated model would produce the identity function) and the confidence histogram shows the empirical distribution of the prediction confidence. The robust-max model fails to provide

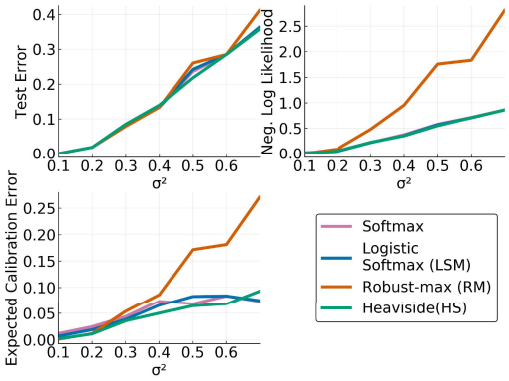


Figure 4: *Likelihood comparison:* The test error, negative log-likelihood and calibration error are plotted as function of the noise (σ^2) in the generated dataset. For highly overlapping classes (large σ^2), the robust-max likelihood yields poor calibration and bad log-likelihood values.

sensitive uncertainty estimates and only predicts with either probability close to zero or close to one. The softmax, logistic-softmax and Heaviside likelihood yield similar predictive performance and confidence calibration. However, as the following experiments show, our approach is much faster than the softmax and Heaviside model. It is the only scalable approach that leads to well calibrated confidences and the logistic-softmax can be used as an efficient replacement of the standard softmax.

5.2 EFFECT OF THE AUGMENTATION

We investigate the effect of the augmentation of the logistic-softmax model and its variational approximation. To this end we compare three different inference methods (1) variational inference for our augmented model (*Aug-*

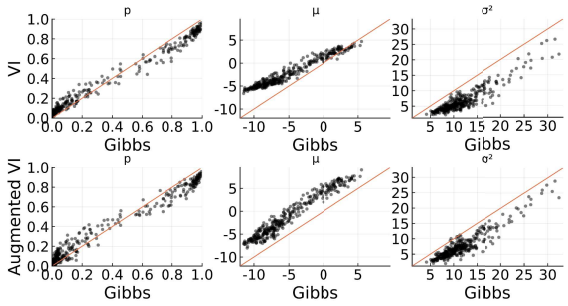


Figure 5: *Effect of the augmentation:* Comparison of the predictive marginals (p), posterior mean (μ) and posterior variance (σ^2) on a test set. Each plot shows the ground truth of the Gibbs sampler on the x-axis. On the y-axis the estimated values by variational inference without augmentation *VI* (top) and augmented variational inference *Augmented VI* are shown (bottom). Our efficient augmented VI method produces values very close to the less efficient VI method. Both methods slightly overestimate the mean (μ) and underestimate the variance (σ^2). However, for both methods the final predictions (p) are close to the ground truth.

mented VI), (2) variational inference without augmentation (approximating the posterior of the original model from section 3.1 using a variational Gaussian), where the gradients are computed via sampling (*VI*) and (3) Gibbs sampling (*Gibbs*), c.f. Section 4.3. After burn-in, the samples from the Gibbs sampler serve as ground truth since they come from the exact posterior. In this experiment we do not use the inducing point approximation and all hyperparameters are fixed. We apply all three methods on the dataset Wine (3 classes) and compare the predictive likelihood (p) and the mean (μ) and variance (σ^2) of the latent GPs on a test set. We compare each entry of the three-dimensional vectors p, μ, σ^2 with the ground truth and display the results for all classes $c = 1, 2, 3$ combined in Figure 5.

Variational inference in the augmented model results in an approximate posterior which is very close to the variational inference solution in the original model. Both methods lead to a similar slight approximation error of the posterior mean μ and variance σ^2 and give predictive marginals p close to the ground truth. The Gibbs sampling approach has a final prediction accuracy of 0.98, whereby both variational inference methods have a final accuracy of 0.96. We find that the augmentation approach can be used as a scalable alternative to standard variational inference.

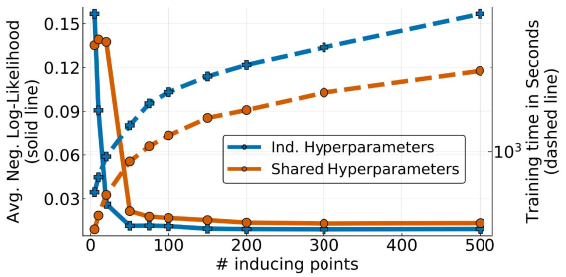


Figure 6: *Inducing points and hyperparameters:* The trade-off between predictive performance and run time is shown. Two versions of our method are used: individual hyperparameters for each GP (blue) and shared hyperparameters (orange). On the left y-axis we plot the negative log-likelihood (solid line) and on the right y-axis the training time (dashed line) as function of the number of inducing points.

5.3 HYPERPARAMETERS

In this experiment we answer two questions. What is the effect of the number of inducing points and what is the difference between using shared hyperparameters and individual hyperparameters for each latent GP? We train our model on the Shuttle dataset (58,000 points, 9 classes) for 200 epochs. We vary the number of inducing points from 5 to 400, and set the GP hyperparameters to be either shared or independent among classes.

In Figure 6 we display the trade-off between predictive performance and training time. We plot the negative log-likelihood (solid lines, y-axis left) and training time (dashed lines, y-axis right) as a function of the number of inducing points. If the number of inducing points is increased, the negative log-likelihood goes down and, oppositely, the training time goes up. We find that using only 200 inducing points already leads to near optimal predictive performance. Using independent hyperparameters over shared hyperparameters does not lead to a significant improvement of the predictive performance but implies a higher computational cost, especially for datasets with a large number of classes.

5.4 NUMERICAL COMPARISON

Finally, we evaluate the predictive performance and convergence speed of our method against other state-of-the-art multi-class GP classification approaches. We compare our logistic-softmax likelihood with augmentation based approach (LSM) against two competitors. First, the robust-max likelihood model (RM) by Hensman and Matthews (2015) which is provided in the package GPFlow (De G. Matthews et al., 2017) and trained by the natural

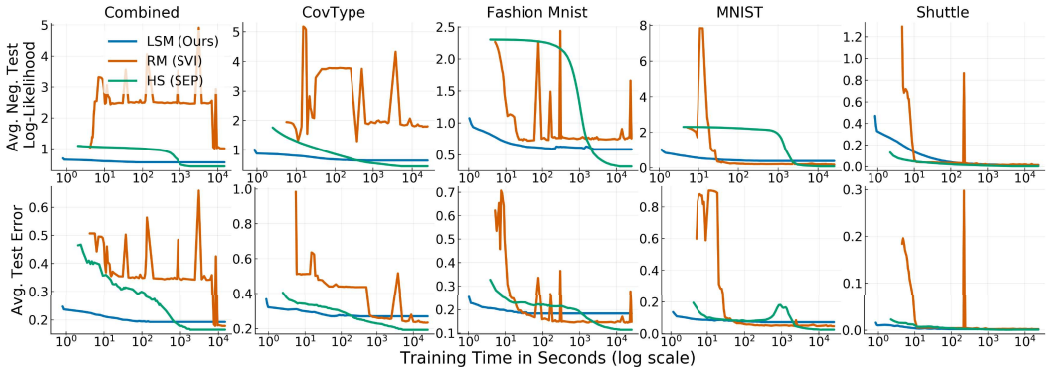


Figure 7: *Numerical comparison:* Prediction error and negative log-likelihood as a function of training time (seconds on a $\log_{10}$ scale). Our method (LSM) converges one to two orders of magnitudes faster than the Heaviside model (HS) and is around 10 times faster than the robust-max model (RM). RM yields poor negative log-likelihood values due to poor uncertainty calibration.

gradient method of Salimbeni et al. (2018) and second, the Heaviside likelihood model (HS) trained by a scalable EP method (Villacampa-Calvo and Hernández-Lobato, 2017). For all methods, the hyperparameters are initialized to the same values, and are optimized using Adam. We compare the methods on five different multi-class benchmark datasets: Combined (98,528 points, 50 features, 3 classes), CovType (581,000 points, 54 features, 7 classes), Fashion-MNIST (70,000 points, 784 features, 10 classes), MNIST (70,000 points, 784 features, 10 classes) and Shuttle (58,000 points, 9 features, 7 classes).

In Figure 7 we plot the test error and negative log-likelihood as functions of the training time for each dataset. We find that our method (LSM) is one to two orders of magnitude faster than the EP based method for the Heaviside model (HS) and around ten times faster than the SVI based method for the robust-max model (RM).

Furthermore, our method consistently beats RM in terms of negative log-likelihood due to the better calibrated uncertainty quantification. Only on the MNIST dataset RM reaches a slightly better log-likelihood. This dataset is easily separable and therefore, suits well to the robust-max likelihood assumptions. On most datasets, the EP based method (HS) leads to slightly better predictive log-likelihood values, but is demanding a much longer training time. In contrast to the log-likelihood, the pure prediction error is not very sensitive to uncertainty calibration. All three methods achieve similar prediction errors whereby HS is a bit better on some datasets.

Moreover, the optimization curves in Figure 7 show that our inference method is much more stable than the SVI approach for the RM model. This is due to our efficient coordinate ascent updates which are given in closed-form. The RM approach suffers from additional noise injected by approximating its gradients.

To summarize, our method is a good choice for fast inference on big datasets. It is particularly well fitted for datasets with overlapping classes where well calibrated uncertainty quantification is important. Due to the closed-form updates our method is more stable than the competitors.

6 CONCLUSION

We proposed an efficient Gaussian process multi-class classification method that builds on data augmentation. The augmented model is conditionally conjugate allowing for fast and stable variational inference based on closed-form updates. The experiments show that our approach leads to better confidence calibration than recent scalable multi-class GP classification methods. Additionally, we achieve competitive prediction performance while being faster than state-of-the-art. For small problems the proposed Gibbs sampler can be used which provides samples from the exact posterior.

The presented work shows how data augmentation can speed up inference in GP based models. Our approach may pave the way to similar augmentation strategies for other Bayesian models. Future work may aim at extending our approach to Bayesian neural networks (BNNs). Inference in BNNs is a hard problem. Exchanging the common softmax link functions with our proposed logistic-softmax may leads to a conditionally conjugate augmentation approach for BNNs. Typically, Gaussian priors are used for the weights of the network. In the augmented model the posterior of the weights would be given in closed-form. This might lead to an efficient inference algorithm.

Acknowledgements

We thank Stephan Mandt, Robert Bamler and Marius Kloft for discussions and feedback on the manuscript. We also thank Simon Danisch for helping with implementation details in Julia. This work was partly funded by the German Research Foundation (DFG) awards KL 2698/2-1 and GRK1589/2 and the by the Federal Ministry of Science and Education (BMBF) awards 031L0023A, 01IS18051A.

Bibliography

Albert, J. H. and Chib, S. (1993). Bayesian analysis of binary and polychotomous response data. *Journal of the American Statistical Association*, 88(422):669–679.

Arthur, D. and Vassilvitskii, S. (2007). k-means++: The advantages of careful seeding. In *Proceedings of the eighteenth annual ACM-SIAM symposium on Discrete algorithms*, pages 1027–1035. Society for Industrial and Applied Mathematics.

Blei, D. M., Kucukelbir, A., and McAuliffe, J. D. (2017). Variational inference: A review for statisticians. *Journal of the American Statistical Association*.

Bojarski, M., Testa, D. D., Dworakowski, D., Firner, B., Flepp, B., Goyal, P., Jackel, L. D., Monfort, M., Muller, U., Zhang, J., Zhang, X., Zhao, J., and Zieba, K. (2016). End to end learning for self-driving cars. *CoRR*, abs/1604.07316.

Buchholz, A., Wenzel, F., and Mandt, S. (2018). Quasi-monte carlo variational inference. In *International Conference on Machine Learning*, pages 667–676.

Caruana, R., Lou, Y., Gehrke, J., Koch, P., Sturm, M., and Elhadad, N. (2015). Intelligible models for healthcare: Predicting pneumonia risk and hospital 30-day readmission. In *KDD*, pages 1721–1730. ACM.

Češnovar, R. and Štrumbelj, E. (2017). Bayesian lasso and multinomial logistic regression on gpu. *PLOS ONE*, 12(6):1–17.

Chai, K. M. A. (2012). Variational multinomial logit gaussian process. *Journal of Machine Learning Research*, 13:1745–1808.

De G. Matthews, A. G., Van Der Wilk, M., Nickson, T., Fujii, K., Boukouvalas, A., León-Villagrá, P., Ghahramani, Z., and Hensman, J. (2017). Gpflow: A gaussian process library using tensorflow. *J. Mach. Learn. Res.*, 18(1):1299–1304.

Donner, C. and Opper, M. (2017). The inverse Ising problem in continuous time: A latent variable approach. *Physical Review E*, 96(6):062104.

Donner, C. and Opper, M. (2018). Efficient Bayesian Inference for a Gaussian Process Density Model. *Conference on Uncertainty in Artificial Intelligence (UAI)*, pages 1–10.

Girolami, M. and Rogers, S. (2006). Variational bayesian multinomial probit regression with gaussian process priors. *Neural Computation*, 18(8):1790–1817.

Guo, C., Pleiss, G., Sun, Y., and Weinberger, K. Q. (2017). On calibration of modern neural networks. In *ICML*, volume 70 of *Proceedings of Machine Learning Research*, pages 1321–1330. PMLR.

Hensman, J., Fusi, N., and Lawrence, N. D. (2013). Gaussian processes for big data. In *Conference on Uncertainty in Artificial Intellegence*.

Hensman, J. and Matthews, A. (2015). Scalable Variational Gaussian Process Classification. *AISTATS*.

Hensman, J., Matthews, A., Filippone, M., and Ghahramani, Z. (2015). MCMC for variationally sparse gaussian processes. *NIPS*.

Hernández-Lobato, D., Hernández-Lobato, J. M., and Dupont, P. (2011). Robust multi-class gaussian process classification. In *Advances in neural information processing systems*, pages 280–288.

Hoffman, M. D., Blei, D. M., Wang, C., and Paisley, J. (2013). Stochastic Variational Inference. *JMLR*.

Izmailov, P., Novikov, A., and Kropotov, D. (2018). Scalable gaussian processes with billions of inducing inputs via tensor train decomposition. *AISTATS*.

Kim, H.-C. and Ghahramani, Z. (2006). Bayesian gaussian process classification with the em-ep algorithm. *IEEE Trans. Pattern Anal. Mach. Intell.*, 28(12):1948–1959.

Kingma, D. P. and Ba, J. (2015). Adam: A method for stochastic optimization. *Proceedings of the 3rd International Conference on Learning Representations*.

Linderman, S. W., Johnson, M. J., and Adams, R. P. (2015). Dependent multinomial models made easy: Stick-breaking with the polya-gamma augmentation. *NIPS*.

Mandt, S., Hoffman, M., and Blei, D. (2016). A Variational Analysis of Stochastic Gradient Algorithms. *ICML*.

Maritz, J. and Lwin, T. (1989). Empirical Bayes Methods with Applications. *Monographs on Statistics and Applied Probability*.

Naeini, M. P., Cooper, G., and Hauskrecht, M. (2015). Obtaining well calibrated probabilities using bayesian

binning. In *Twenty-Ninth AAAI Conference on Artificial Intelligence*.

Owen, A. (1998). Monte Carlo extension of quasi-Monte Carlo. *1998 Winter Simulation Conference. Proceedings (Cat. No.98CH36274)*, 1(1):571–577.

Polson, N. G., Scott, J. G., and Windle, J. (2013). Bayesian inference for logistic models using pólya–gamma latent variables. *Journal of the American Statistical Association*, 108(504):1339–1349.

Rasmussen, C. E. and Williams, C. K. I. (2005). *Gaussian Processes for Machine Learning (Adaptive Computation and Machine Learning)*. The MIT Press.

Riihimäki, J., Jylänki, P., and Vehtari, A. (2013). Nested expectation propagation for gaussian process classification. *J. Mach. Learn. Res.*, 14(1):75–109.

Ruiz, F. J. R., Titsias, M. K., Dieng, A. B., and Blei, D. M. (2018). Augment and reduce: Stochastic inference for large categorical distributions. *ICML*.

Salimbeni, H., Eleftheriadis, S., and Hensman, J. (2018). Natural gradients in practice: Non-conjugate variational inference in gaussian process models. *AISTATS*.

Titsias, M. (2016). One-vs-each approximation to softmax for scalable estimation of probabilities. In *Advances in Neural Information Processing Systems*, pages 4161–4169.

Titsias, M. K. (2009). Variational learning of inducing variables in sparse gaussian processes. In *In Artificial Intelligence and Statistics 12*, pages 567–574.

Villacampa-Calvo, C. and Hernández-Lobato, D. (2017). Scalable multi-class gaussian process classification using expectation propagation. *ICML*.

Walker, S. G. (2011). Posterior sampling when the normalizing constant is unknown. *Communications in Statistics–Simulation and Computation®*, 40(5):784–792.

Wenzel, F., Galy-Fajou, T., Donner, C., Kloft, M., and Opper, M. (2019). Efficient gaussian process classification using polya-gamma data augmentation. *AAAI*.

Williams, C. K. I. and Barber, D. (1998). Bayesian classification with gaussian processes. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 20:1342–1351.

Xiong, H., Wu, J., and Liu, L. (2010). Classification with classoverlapping: A systematic study. In *Proceedings of the 1st International Conference on E-Business Intelligence (ICEBI2010),*. Atlantis Press.

Xiong, W., Droppo, J., Huang, X., Seide, F., Seltzer, M., Stolcke, A., Yu, D., and Zweig, G. (2016). Achieving human parity in conversational speech recognition. *CoRR*, abs/1610.05256.

5

Automated Augmented Conjugate Inference for Non-conjugate Gaussian Process Models

The larger question following the work on Pólya-Gamma variables and other augmentation works such as Nguyen and Wu [41] or Henao et al. [20] is: What likelihoods have a scale mixture representation?

This article, extending the work of Palmer [43] partially answers by finding a class of functions, the positive-definite radial functions, guaranteed to be interpretable as scale mixtures of Gaussians. The paper also provides an algorithm to directly infer the CAVI updates and Gibbs sampling algorithm from the likelihood.

Authors:
Théo Galy-Fajou,[1], Florian Wenzel,[2], Manfred Opper[1]
[1]TU Berlin, Germany, [2]Google Research

Details:
Type: Conference article Submitted: October 2019
Accepted: December 2019
URL: `https://proceedings.mlr.press/v108/galy-fajou20a.html`
Conference: AISTATS 2020

Contributions:

For an explanation of the terms see the Contributor Roles Taxonomy (CReditT)

	T.G-F.	F.W.	M.O.
Conceptualization	✓	✓	✓
Methodology	✓		
Formal Analysis	✓	✓	✓
Software	✓		
Investigation	✓		
Writing - Original Draft	✓	✓	
Writing - Review & Editing	✓	✓	✓
Supervision			✓
Funding Acquisition			✓

Automated Augmented Conjugate Inference for Non-conjugate Gaussian Process Models

Théo Galy-Fajou
Technical University of Berlin

Florian Wenzel
Google Research*

Manfred Opper
Technical University of Berlin

Abstract

We propose *automated augmented conjugate inference*, a new inference method for non-conjugate Gaussian processes (GP) models. Our method automatically constructs an auxiliary variable augmentation that renders the GP model conditionally conjugate. Building on the conjugate structure of the augmented model, we develop two inference methods. First, a fast and scalable stochastic variational inference method that uses efficient block coordinate ascent updates, which are computed in closed form. Second, an asymptotically correct Gibbs sampler that is useful for small datasets. Our experiments show that our method are up two orders of magnitude faster and more robust than existing state-of-the-art black-box methods.

1 INTRODUCTION

Developing automated yet efficient Bayesian inference methods for Gaussian process (GP) models is a challenging problem that has attracted considerable attention within the probabilisitic machine learning community (Salimbeni et al., 2018; Wenzel et al., 2019). A GP defines a distribution over functions and can be used as a flexible building block to develop expressive probabilistic models. By choosing an appropriate likelihood function on top of a latent GP, a variety of interesting models is obtained, which are successfully used in several application areas including robotics (Beckers et al., 2019), facial behavior analysis (Eleftheriadis et al., 2017) and electrical engineering (Pandit and Infield, 2018). For instance, using a logistic likelihood leads to a binary GP classification model, and using a Student-t likelihood can be used for robust regression.

The main challenge in these models is to infer the latent GP given a general non-Gaussian likelihood. Methods that are more generally applicable often treat the model as a black box and are based on sampling or numerical quadrature, thus, preventing efficient optimization (Hensman et al., 2015; Salimbeni et al., 2018). On the other side. a lot of methods focus on special cases of GP models (i.e. special likelihood functions) by exploiting model specific properties, e.g. binary classification (Polson et al., 2013).

In this work, we develop *automated augmented conjugate inference (*AACI*)*. AACI is an efficient inference framework, which is applicable to a large class of GP models that use a super-Gaussian likelihood[1]. It automatically exploits specific properties of the likelihood leading to an inference algorithm that is up to two orders of magnitudes faster than the state of the art.

Our approach builds on an auxiliary variable augmentation of the model: we add a latent variable to the model such that the original model is recovered when this variable is integrated out. We consider an augmentation that renders the model conditionally conjugate. In a conditionally conjugate model, all complete conditional distributions (the posterior distribution of one random variable given all the others), can be computed in closed form. Moreover, we show that inference in the augmented conditionally conjugate model is much easier than in the original model and demonstrate superior performance over the state of the art.

Building on the conditionally conjugate augmentation, AACI provides two options for inference: a scalable variational inference method based on efficient closed-form coordinate ascent updates and an exact Gibbs sampling method, which is useful on smaller datasets.

Our main contributions are as follows:

- We introduce AACI: an automated inference method for GP models with a super-Gaussian likelihood.
- We propose two inference modules: augmented variational inference, which scales to large datasets contain-

Proceedings of the 23rdInternational Conference on Artificial Intelligence and Statistics (AISTATS) 2020, Palermo, Italy. PMLR: Volume 108.

*Work done while at TU Berlin

[1]The definition of the family of super-Gaussian likelihoods is given in Section 3.

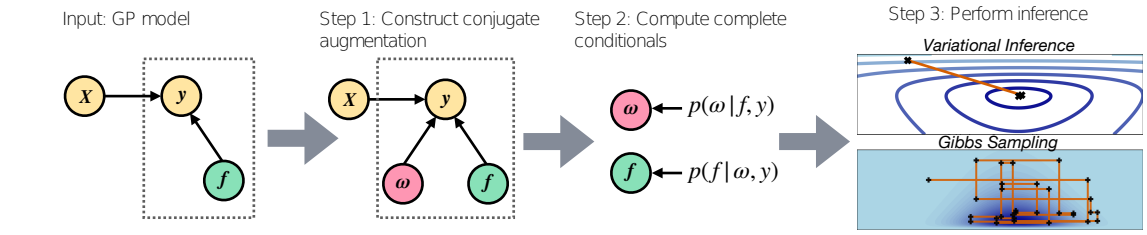


*Figure 1. Automated augmented conjugate inference (*AACI*)* performs automated efficient inference in non-conjugate Gaussian process models. In the first step, AACI translates the GP model into an augmented model that is conditionally conjugate. In the second step, the complete conditionals are computed in closed form. In the final step, AACI provides two options: (A) fast stochastic variational inference based on coordinate ascent updates, which easily scales to big datasets and (B) an asymptotically exact Gibbs sampler, which provides high quality samples from the true posterior but is limited to smaller datasets.

ing millions of instances and an exact Gibbs sampler, which is useful for small datasets.

- The experiments demonstrate that the augmented variational inference module of AACI outperforms the state of the art in terms of speed by up to two orders of magnitude while being competitive in terms of prediction performance. The Gibbs sampler module leads to a much better efficient sample size while still being up to ten times faster than Hamiltonian Monte Carlo.

The paper is structured as follows: Section 2 gives a high-level overview about our novel inference method AACI. In Section 3, we provide a detailed discussion of the algorithm and proof that our approach indeed leads to conditionally conjugate models. We discuss related work in Section 4 and show our experimental results in Section 5. Finally, Section 6 concludes and lays out future research directions. Our source code for the experiments is included in a gitgub repository[2].

2 AUTOMATED AUGMENTED CONJUGATE INFERENCE

Let $X = (\boldsymbol{x}_1, \ldots, \boldsymbol{x}_n)^\top \in \mathbb{R}^{n \times d}$ be a matrix of data points and $\boldsymbol{y} = (y_1, \ldots, y_n) \in \mathbb{R}^n$ the corresponding target values. The goal is to learn a mapping from the input points to the target values via a latent function f. We assume a prior GP distribution (with mean prior $\boldsymbol{\mu}_0$ and covariance function $k(x, x')$) on the latent function and the data labels $\boldsymbol{y} = (y_1, \ldots, y_n)$ are connected to f via a factorizable likelihood

$$p(f) = \mathrm{GP}(f|\boldsymbol{\mu}_0, k), \quad p(\boldsymbol{y}|f, X) = \prod_{i=1}^{n} p(y_i|f(\boldsymbol{x}_i)).$$

[2] https://github.com/theogf/AutoConjGP_Exp

The key inference challenge in the GP models is to compute the posterior distribution of the latent function

$$p(f|y) = \frac{p(y|f)p(f)}{\int p(y|f)p(f)\mathrm{d}y}.$$

This is a challenging problem. Inference in GP models scale cubically in the number of data points and is intractable for non-Gaussian likelihoods.

Ideally, we would like an efficient inference method that is not hand-tailored to a specific type of likelihood and hence allows for experimenting with different types of GP models on big datasets in a scalable manner. Thus, we need a flexible inference method that works for a large class of likelihoods, is fast and ideally does not involve inefficient black box approaches as approximating the objective by sampling.

2.1 Automated Augmented Conjugate Inference

We introduce the *automated augmented conjugate inference (*AACI*)* to achieve this goal. AACI accelerates training of GP models whose likelihood is in the family of super-Gaussian likelihood functions.

AACI translates the intractable non-conjugate model into an easier, conditionally conjugate model by adding auxiliary random variables to the model. Inference in conditionally conjugate models is a classic and well-studied problem (Bishop, 2006). Because of the special structure of conditionally conjugate models, many efficient inference methods exist (Wang and Blei, 2013). Based on the automatically constructed augmentation, we propose an efficient variational inference method using coordinate ascent updates and a Gibbs sampler.

The inference pipeline of AACI. AACI consists of three steps. In the first step, a conjugate augmentation of the model is constructed by adding auxiliary variables ω to the

model. Then, the complete conditional distributions of the latent function f and auxiliary variables ω are computed. In the final step, we provide two options to perform inference.

The *variational inference* (VI) module of AACI performs block coordinate ascent updates, computed in closed form. The updates are much more efficient than ordinary Euclidean gradient updates, which are used in most previous approaches. The *Gibbs sampling* module of AACI builds on the complete conditional distributions and provides exact samples from the true posterior. For each type of likelihood, the sampler is automatically constructed.

The inference pipeline of AACI is summarized in Fig. 1. In the following, we give an overview of how each module of our inference pipeline works and provide the details in Section 3.

(1) Augmenting the model. The first step of our inference framework constructs an *auxiliary variable augmentation* that renders the model *conditionally conjugate*. Our augmentation approach finds a Gaussian scale mixture representation of the intractable likelihood

$$p(y_i|f_i) = \int p(y_i|f_i,\omega_i)p(\omega_i)\mathrm{d}\omega, \tag{1}$$

where $p(y_i|f_i,\omega_i)$ is an unnormalized Gaussian distribution in f_i with precision ω_i and $p(\omega_i)$ is the prior distribution of the auxiliary variable. The construction of the distribution $p(\omega)$ is based on an inverse Laplace transformation and is discussed in Section 3.1.

Building on Eq. 1, we augment the GP model by a set of auxiliary variables $\boldsymbol{\omega} = (\omega_1, \dots, \omega_n)$ leading to the augmented joint distribution

$$p(\boldsymbol{y}, \boldsymbol{f}, \boldsymbol{\omega}) = \prod_i p(y_i|f_i,\omega_i)p(\omega_i)p(\boldsymbol{f}),. \tag{2}$$

The auxiliary variable augmentation is constructed in a way such that the augmented model is *conditionally conjugate*, i.e. the complete conditional distributions $p(\boldsymbol{\omega}|\boldsymbol{f},\boldsymbol{y})$ and $p(\boldsymbol{f}|\boldsymbol{\omega},\boldsymbol{y})$ are in the same family as their associated priors.

(2) Computing the complete conditionals. The complete conditionals of $\boldsymbol{f}$ and the auxiliary variables ω_i are computed in closed form and are given by

$$\begin{aligned} p(\boldsymbol{f}|\boldsymbol{y},\boldsymbol{\omega}) =& \mathcal{N}(\boldsymbol{f}|\boldsymbol{\mu},\boldsymbol{\Sigma}) \\ p(\omega_i|f_i,y_i) =& \pi_\varphi(\omega_i|c_i), \end{aligned}$$

where φ is a function determined by the type of the likelihood (see Eq. 4) and the parameters $\boldsymbol{\mu}, \boldsymbol{\Sigma}, c_i$ have closed-form expressions and are described in Section 3.2. The distribution family $\pi_\varphi(\omega|c)$ is derived by an exponential tilting of the prior distribution $p(\omega)$ and is discussed in Section 3.2.

(3a) Augmented variational inference. In step 3, AACI provides two options to perform inference. We first discuss the variational inference module, which approximates the posterior by a variational distribution and easily scales to big datasets.

We assume a mean-field variational distribution, where the latent GP $\boldsymbol{f}$ and the auxiliary variables $\boldsymbol{\omega}$ are decoupled, i.e. $q(\boldsymbol{f},\boldsymbol{\omega}) = q(\boldsymbol{f})q(\boldsymbol{\omega})$. The optimal variational distribution of $\boldsymbol{\omega}$ naturally factorizes, i.e. $q(\boldsymbol{\omega}) = \prod_i q(\omega_i)$. Following standard results (Bishop, 2006) the variational distributions can be iteratively optimized by the block-coordinate ascent updates:

$$\begin{aligned} q(\boldsymbol{f}) &\propto \exp\left(\mathbb{E}_{q(\boldsymbol{\omega})}\left[\log p(\boldsymbol{f}|\boldsymbol{\omega},\boldsymbol{y})\right]\right) \\ q(\omega_i) &\propto \exp\left(\mathbb{E}_{q(\boldsymbol{f})}\left[\log p(\omega_i|\boldsymbol{f},\boldsymbol{y})\right]\right). \end{aligned} \tag{3}$$

In Section 3.3, we show that these updates are given in closed form and can be computed efficiently without resorting to numerical methods. To scale to big datasets we employ SVI (Hoffman et al., 2013) and replace the original latent GP f by Titsias (2009) sparse approximation building on inducing points .

(3b) Exact inference via Gibbs sampling. Building on the conditionally conjugate augmentation, it is straightforward to derive a Gibbs sampler. In order to sample from the exact posterior, we alternate between drawing a sample from each complete conditional distribution

$$\begin{aligned} \boldsymbol{\omega}^t &\sim p(\boldsymbol{\omega}|\boldsymbol{f}^{t-1},\boldsymbol{y}), \\ \boldsymbol{f}^t &\sim p(\boldsymbol{f}|\boldsymbol{\omega}^t,\boldsymbol{y}). \end{aligned}$$

The augmented variables are naturally marginalized out and the latent GP samples $\{\boldsymbol{f}^t\}$ will be from the true posterior $p(\boldsymbol{y}|\boldsymbol{f})$. As we empirically show in Section 5.1, the Gibbs sampler leads to very fast mixing and outperforms standard Hamiltonian Monte Carlo sampling.

3 ALGORITHM DETAILS

Here we provide the details on the *automated augmented conjugate inference (*AACI*)* algorithm. We start by specifying the class of GP models that we consider in our framework. We then discuss the technical details of AACI and proof that the automatically constructed augmentation indeed leads to a conditionally conjugate model.

GP Models with a super-Gaussian likelihood. AACI can be applied to GP models, where the likelihood is within the class of super-Gaussian likelihoods. A super-Gaussian likelihood is of the form

$$p(\boldsymbol{y}|\boldsymbol{f};\theta) = C(\theta)e^{g(\boldsymbol{y};\theta)^\top \boldsymbol{f}}\varphi(||h(\boldsymbol{f},\boldsymbol{y})||_2^2), \tag{4}$$

where θ are hyperparameters of the likelihood, $C(\theta)$ is the normalizing constant, $g(\boldsymbol{y};\theta)$ is an arbitrary function, φ is

a *positive definite radial (*PDR*) function*[3], and h is a linear function in $\boldsymbol{f}$, such that we can write

$$||h(\boldsymbol{f},\boldsymbol{y})||_2^2 = \alpha(\boldsymbol{y},\theta) - \beta(\boldsymbol{y},\theta)^\top \boldsymbol{f} + \gamma(\boldsymbol{y},\theta)||\boldsymbol{f}||_2^2, \quad (5)$$

where α, β, γ are arbitrary functions. We omit θ in the later derivations for clarity.

Many interesting models are instances of super-Gaussian likelihood GP models. In Table 1, we present several likelihood functions with their corresponding parameter settings of the super-Gaussian likelihood as given in Eq. 4.

Constructing new likelihoods. Using Eq. 4, we can also construct novel likelihood functions based on existing kernel functions. In this paper we propose the Matern 3/2 likelihood.

3.1 Step 1: Conjugate augmentation

Given the likelihood of the model, AACI constructs a conditionally conjugate auxiliary variable augmentation as follows. We first define a family of distribution $\pi_\varphi(\omega|c)$, which will be useful for constructing the augmentation.

For the case $c = 0$, the distribution $\pi_\varphi(\omega|0)$ is defined by the inverse Laplace transform of $\varphi(\cdot)$,

$$\pi_\varphi(\omega|0) = \mathcal{L}^{-1}\left\{\varphi(\cdot)\right\}(\omega). \quad (6)$$

The inverse Laplace is the inverse mapping of the Laplace transformation and can be computed by the Bromwich integral formula[4] (Debnath and Bhatta, 2014) and it defines a valid density in our setting (see proof of Theorem 1). Remarkably, we will see that for the final updates of our algorithm, we do not need to compute the inverse Laplace transformation explicitly.

We generalize the base distribution $\pi_\varphi(\omega|0)$ by applying an exponential tilting:

$$\pi_\varphi(\omega|c) = \frac{e^{-c^2\omega}\pi_\varphi(\omega|0)}{\varphi(c^2)}, \quad (7)$$

where $c \in \mathbb{R}$.

Theorem 1. *A GP model with a super-Gaussian likelihood (of the form of Eq. 4) is rendered* ***conditionally conjugate*** *by the auxiliary variable augmentation* $p(\boldsymbol{y},\boldsymbol{f},\boldsymbol{\omega};\theta) = p(\boldsymbol{y}|\boldsymbol{f},\boldsymbol{\omega};\theta)p(\boldsymbol{f})p(\boldsymbol{\omega})$. *The augmented likelihood is*

$$p(\boldsymbol{y}|\boldsymbol{f},\boldsymbol{\omega};\theta) = C(\theta)\exp\left(g(\boldsymbol{y};\theta)^\top \boldsymbol{f} - ||h(\boldsymbol{f},\boldsymbol{y})||_2^2\boldsymbol{\omega}\right)$$

and the prior distribution of the auxiliary variables is

$$p(\boldsymbol{\omega}) = \pi_\varphi\left(\boldsymbol{\omega}|0\right).$$

Proof: We first apply Schoenberg's theorem (Schoenberg, 1938), which states that a function $\mathbb{R}^d \ni \boldsymbol{x} \to \varphi(\|\boldsymbol{x}\|_2^2)$ is a PDR function for any dimension $d > 0$ if and only if $\varphi(r)$ is a completely monotone function on the domain $r \geq 0$.

A completely monotone function $\varphi(\cdot)$ has the property that it is infinitely differentiable and its derivatives have an alternating sign (Bernstein et al., 1929), i.e.

$$(-1)^k\varphi^{(k)}(r) > 0, \quad r \in [0,+\infty),\ k = 0,1,2,\ldots. \quad (8)$$

As a direct consequence, $\varphi(\cdot)$ is a positive, decreasing, and convex function and the first derivative of $\varphi(\cdot)$ is a concave function.

Building on these properties, Widder (1946) states that we can rewrite $\varphi(\|h(f,y)\|_2^2)$ as a Gaussian scale-mixture

$$\varphi\left(\|h(f,y)\|_2^2\right) = \int_0^\infty e^{-\|h(f,y)\|_2^2\omega}d\mu(\omega), \quad (9)$$

with respect to a Borel measure $\mu(\omega)$. We apply the monotone convergence theorem (Yeh, 2006), which gives that $\mu(\omega)$ is even a probability measure iff $\lim_{r\to 0}\varphi(r) = 1$. Since we have a probability measure, we write $d\mu(\omega) = p(\omega)d\omega$ and which leads to the equality $\varphi(r) = \mathcal{L}\left\{p(\omega)\right\}(r)$, where $\mathcal{L}$ denotes the Laplace transformation. The inverse Laplace transformation gives the density of the auxiliary variable $p(\omega) = \mathcal{L}^{-1}\left\{\varphi(r)\right\}(\omega) = \pi_\varphi(\omega|0)$.

Therefore we can rewrite the super-Gaussian likelihood Eq. 4 as :

$$p(y|f) = C(\theta)\int_0^\infty e^{-g(y)f - \|h(f,y)\|_2^2\omega}p(\omega)d\omega. \quad (10)$$

Adding the auxiliary variable ω with prior $p(\omega)$ to the model, we obtain the augmented likelihood $p(\boldsymbol{y}|\boldsymbol{f},\boldsymbol{\omega};\theta) = C(\theta)\exp\left(g(\boldsymbol{y};\theta)^\top\boldsymbol{f} - ||h(\boldsymbol{f},\boldsymbol{y})||_2^2\boldsymbol{\omega}\right)$.

Since the function $g(\boldsymbol{y};\theta)^\top\boldsymbol{f} - ||h(\boldsymbol{f},\boldsymbol{y})||_2^2\boldsymbol{\omega}$ is by definition quadratic in $\boldsymbol{f}$ the augmented likelihood is proportional to an (unnormalized) Gaussian distribution in $\boldsymbol{f}$, hence, conditionally conjugate in $\boldsymbol{f}$.
For the augmented variable ω_i, the likelihood $p(y|\omega, f)$ act as an exponential tilting of $p(\omega)$ and the full conditional in ω will stay in the same family of distributions. QED.

3.2 Step 2: Complete Conditionals

Since the augmented model (Section 3.1) is conditionally conjugate, the complete conditional distribution are in the

[3] φ is a positive definite radial function if $\varphi(r)$ is completely monotone for all $r \geq 0$ and $\lim_{r\to 0}\varphi(r) = 1$.

[4] The inverse Laplace transformation of a function $\varphi(\cdot)$ can be computed by $\mathcal{L}^{-1}\left\{\varphi(\cdot)\right\}(\omega) = \lim_{T\to\infty}\frac{1}{2\pi i}\int_{b-iT}^{b+iT} e^{r\omega}\varphi(r)dr$, where b can be arbitrarily chosen but has to be larger than the real part of all singularities of φ.

Likelihood	Full form	$g(f,y)$	$h(f,y)$	$\varphi(r)$
Student-t	$\frac{\Gamma\left(\frac{\nu+1}{2}\right)}{\sqrt{\nu\pi}\sigma\Gamma\left(\frac{\nu}{2}\right)}\left(1+\frac{(y-f)^2}{\nu\sigma^2}\right)^{-\frac{\nu+1}{2}}$	0	$\frac{f-y}{\sigma}$	$\left(1+\frac{r}{\nu}\right)^{-\frac{\nu+1}{2}}$
Laplace	$\frac{1}{2\beta}\exp\left(-\frac{\lvert y-f\rvert}{\beta}\right)$	0	$f-y$	$\exp\left(-\frac{\sqrt{r}}{\beta}\right)$
Logistic	$\frac{1}{2}\exp\left(\frac{yf}{2}\right)\cosh^{-1}\left(\frac{\lvert yf\rvert}{2}\right)$	$\frac{yf}{2}$	$\frac{f}{2}$	$\cosh^{-1}\left(\sqrt{r}\right)$
Bayesian SVM	$\exp\left((yf-1)-\lvert 1-yf\rvert\right)$	yf	$1-yf$	$\exp(-\sqrt{r})$
Matern 3/2	$\frac{\sqrt{3}}{4\rho}(1+\frac{\sqrt{3}\lvert y-f\rvert}{\rho})\exp(-\frac{\sqrt{3}\lvert y-f\rvert}{\rho})$	0	$f-y$	$(1+\frac{\sqrt{3r}}{\rho})\exp(-\frac{\sqrt{3r}}{\rho})$

Table 1. Many interesting GP models are members of the super-Gaussian likelihood family introduced in Section 3. We display the full likelihood and the corresponding terms of the super-Gaussian likelihood as described in Eq. 4. Some models were already considered independently but our approach provides a unified view.

same family as their associated prior distributions and are given in closed form.

Theorem 2. *The **complete conditional distributions** of the augmented model presented in Section 3.1 are given by*

$$\begin{aligned} p(\omega_i|f_i,y_i) =& \pi_\varphi\left(\omega_i|\|h(f_i,y_i)\|_2\right), \\ p(\boldsymbol{f}|\boldsymbol{y},\boldsymbol{\omega}) =& \mathcal{N}\left(\boldsymbol{f}|\boldsymbol{\mu},\boldsymbol{\Sigma}\right), \end{aligned} \tag{11}$$

where $\boldsymbol{\Sigma} = \left(\operatorname{diag}\left(2\boldsymbol{\omega}\circ\gamma(\boldsymbol{y})\right)+K^{-1}\right)^{-1}$ *and* $\boldsymbol{\mu} = \boldsymbol{\Sigma}\left(g(\boldsymbol{y})+\boldsymbol{\omega}\circ\beta(\boldsymbol{y})+K^{-1}\boldsymbol{\mu}_0\right)$, $\circ$ *denotes the Hadamard product and the function* $h(\cdot)$ *is given by the form of likelihood (see Eq.5).*

The proof is given in Appendix A.1

3.3 Step 3: Efficient inference

In the final step of our inference pipeline, we leverage the conditionally conjugate structure of the augmented model and derive two inference methods. First, we propose a scalable stochastic variational inference (SVI) method that builds on efficient block coordinate ascent updates (CAVI) updates, computed in closed form. Second, we develop a Gibbs sampling scheme that generates samples from the exact posterior.

3.3.1 Augmented variational inference

We implement the classic stochastic variational inference (SVI) algorithm for conditionally conjugate models described by Hoffman et al. (2013), which builds on block coordinate ascent updates. The updates can be interpreted as natural gradient updates and are much more efficient than ordinary Euclidean gradient updates (Amari, 1998).

Variational approximation. We approximate the posterior distribution of the latent GP values by assuming a decoupling between $\boldsymbol{f}$ and $\boldsymbol{\omega}$. The family of the optimal variational distribution can be easily determined by averaging the complete conditionals in log-space, as given in Eq. 3 (see e.g. Blei et al., 2017). From the above decoupling assumption, it follows that the optimal variational posterior is in the variational family

$$q(\boldsymbol{f},\boldsymbol{\omega}) = q(\boldsymbol{f})\prod_{i=1}^{N} q(\omega_i), \tag{12}$$

where $q(\boldsymbol{f}) = \mathcal{N}\left(\boldsymbol{f}|\boldsymbol{m},\boldsymbol{S}\right)$ and $q(\omega_i) = \pi_\varphi(\omega_i|c_i)$ and $\boldsymbol{m},\boldsymbol{S}$ and $\boldsymbol{c}$ are the *variational parameters*.

Variational updates. We start with deriving the variational updates for the variational Gaussian distribution,

$$\begin{aligned} q(\boldsymbol{f}) &\propto \exp\left[\mathbb{E}_{q(\boldsymbol{\omega})}\left[\log p(\boldsymbol{f}|\boldsymbol{\omega},\boldsymbol{y})\right]\right] \\ &\propto \exp\left[\sum_i g(y_i)f_i - \|h(f_i,y_i)\|_2^2\mathbb{E}_{q(\omega_i)}\left[\omega_i\right]\right]p(\boldsymbol{f}) \end{aligned}$$

Computing the variational updates of $q(\boldsymbol{f})$ boils down to computing the first moment of $\boldsymbol{\omega}$. Remarkably, the moments of π_φ can be computed without computing the closed-form density of π_φ explicitly, i.e. without evaluating the inverse Laplace transformation of φ (Eq. 6).

The moments can be computed by differentiating the moment generating function, which is itself a Laplace transform. For our algorithm, we only need the first moment of ω, which is given by

$$\mathbb{E}_{q(\omega)}\left[\omega\right] = \left.\frac{d\mathcal{L}\left\{q(\omega)\right\}(-t)}{dt}\right|_{t=0} = -\frac{\varphi'(c^2)}{\varphi(c^2)} = \overline{\omega},$$

which can be cheaply computed via automatic differentiation.

The updates for the variational distribution of the auxiliary variables $q(\boldsymbol{\omega})$ are computed as follows.

$$\begin{aligned} q(\omega_i) &\propto \exp\left[-\mathbb{E}_{q(f_i)}\left[\|h(f_i,y_i)\|_2^2\right]\omega_i + \log p(\omega_i)\right] \\ &\propto \exp\left(-\mathbb{E}_{q(f_i)}\left[\|h(f_i,y_i)\|_2^2\right]\omega_i\right)p(\omega_i) \\ &= \pi_\varphi(\omega_i|\sqrt{\mathbb{E}_{q(f_i)}\left[h(f_i,y_i)^2\right]}). \end{aligned}$$

We get then the update $c_i = \sqrt{\mathbb{E}_{q(f_i)}\left[\|h(f_i, y_i)\|_2^2\right]}$, which can be easily computed in closed form since $\|h(f_i, y_i)\|_2^2$ is a quadratic function of f_i.

The coordinate ascent variational inference (CAVI) method is summarized in Algorithm 1.

Algorithm 1 Augmented Variational Inference

Input: Data $(\mathbf{X}, \boldsymbol{y})$, GP model $p(\boldsymbol{y}|\boldsymbol{f})$, kernel k
Output: Approximate posterior $q(f) = \mathcal{N}(f \mid \boldsymbol{m}, \boldsymbol{S})$
for iteration $t = 1, 2, \ldots,$ **do**
  `# Local updates:`
  for $i \in 1 : N$ **do**
    $c_i = \sqrt{\mathbb{E}_{q(f)}\left[h(f_i, y_i)^2\right]}$
    $\overline{\omega}_i = \mathbb{E}_{q(\omega_i)}\left[\omega_i\right] = -\varphi'(c_i^2)/\varphi(c_i^2)$
  end for
  `# Coordinate ascent updates (CAVI):`
  $\boldsymbol{S} \leftarrow \left(\operatorname{diag}\left(2\overline{\boldsymbol{\omega}} \circ \gamma(\boldsymbol{y})\right) + K^{-1}\right)^{-1}$
  $\boldsymbol{m} \leftarrow \boldsymbol{S}\left(K^{-1}\mu_0 + g(\boldsymbol{y}) + \overline{\boldsymbol{\omega}} \circ \beta(\boldsymbol{y})\right)$
end for

Sparse GP approximation. To scale our method to big datasets, we approximate the latent GP $\boldsymbol{f}$ by a *sparse Gaussian process* building on *inducing points*. We introduce M inducing points $\boldsymbol{u}$ and connect the GP values with the inducing points via the joint prior distribution $p(\boldsymbol{f}, \boldsymbol{u})$ given in Titsias (2009). The introduction of inducing points preserves conditional conjugacy and allows for mini-batch sampling of the data (stochastic variational inference). This scales the algorithm to big datasets and has the computational complexity $\mathcal{O}(M^3)$. The SVI version of our algorithm only slightly changes the updates that are presented in Algorithm 1. It is deferred to Appendix A.3.

3.3.2 Gibbs sampling

To sample from the exact posterior distribution, a Gibbs sampling scheme alternates between sampling from the complete conditional distributions. In the following we propose a sampling scheme for the distribution family $\pi_\varphi(\omega|c)$ that is automatically constructed given the PDR function of the likelihood $\varphi(\cdot)$

The distribution class π_φ is defined in Eq. 6 and is based on the inverse Laplace transform of $\varphi(\cdot)$. However there is no general approach to compute the inverse Laplace in closed form (Cohen, 2007). We circumvent this issue by proposing an algorithm that only evaluates the inverse Laplace transformation point-wise but does not need access to its full analytical form. We apply the method proposed by Ridout (2009), which build on the fact that the cumulative density function (CDF) $F_{\pi_\varphi(\omega|c)}(\cdot)$ can be computed via the inverse Laplace transform of a scaled (forward) Laplace transform,

$$F_{\pi_\varphi(\omega|c)}(x) = \mathcal{L}^{-1}\left\{\frac{\mathcal{L}\left\{\pi_\varphi(\omega|c)\right\}(s)}{s}\right\}(x) = \mathcal{L}^{-1}\left\{\frac{\varphi(s + c^2)}{s\varphi(c^2)}\right\}(x).$$

To generate samples from $\pi_\varphi(\omega|c)$, we first generate a uniform sample $u \sim \mathcal{U}[0, 1]$ and then push it through the inverse CDF, $\omega = F^{-1}_{\pi_\varphi(\omega|c)}(u)$ (Devroye, 1986) Finally, to compute the inverse CDF, we solve a fixed point problem using the modified Newton-Raphson method described by Ridout (2009). We solve the equation $F_{\varphi(c)}(\omega) = u$ by repeatedly setting $\omega \leftarrow \omega - F_{\varphi(c)}(\omega)/\pi_\varphi(\omega|c)$ until reaching convergence. We numerically approximate the (forward) CDF $F_{\varphi(c)}(\omega)$ by the cheap trapezoidal method introduced in Abate et al. (2000), which has error guarantees. The cost of this process is negligible against the matrix inversion for sampling $\boldsymbol{f}$. All steps are summarized in Algorithm 2.

Note that for some likelihood functions (e.g. the logistic likelihood function), the inverse Laplace transform can be derived analytically and the steps described above can be optimized by using an existing the sampler for the corresponding complete conditional distribution.

Algorithm 2 Gibbs Sampling

Input: Data $(\mathbf{X}, \boldsymbol{y})$, GP model $p(\boldsymbol{y}|\boldsymbol{f})$, kernel k
Output: Posterior samples $\{\boldsymbol{f}^t\} \sim p(\boldsymbol{f} \mid \boldsymbol{y})$
for sample index $t = 1, 2, \ldots,$ **do**
  `# Sample` $\boldsymbol{\omega} \sim p(\boldsymbol{\omega}|\boldsymbol{f}, \boldsymbol{y})$:
  for $i \in 1 : N$ **do**
    Compute $c_i = \|h(f_i, y_i)\|_2$
    Sample $u_i \sim \mathcal{U}[0, 1]$
    `# Compute inverse cdf` $\omega_i = F^{-1}_{\pi_\varphi(c_i)}(u_i)$:
    Initialize $\omega_i > 0$
    while $|\widetilde{F}_{\pi_\varphi(c_i)}(\omega_i) - u_i| > \epsilon$ **do**
      Approximate $\widetilde{F}_{\pi_\varphi}(\omega_i)$, $\widetilde{\pi}_\varphi(\omega_i|c_i)$(see Sec.3.3.2)
      $\omega_i \leftarrow \omega_i - \frac{\widetilde{F}_{\pi_\varphi(c_i)}(\omega_i)}{\widetilde{\pi}_\varphi(\omega_i|c_i)}$
    end while
  end for
  `# Sample` $\boldsymbol{f} \sim p(\boldsymbol{f}|\boldsymbol{\omega}, \boldsymbol{y})$:
  $\boldsymbol{\Sigma} = \left(\operatorname{diag}\left(2\boldsymbol{\omega} \circ \gamma(\boldsymbol{y})\right) + K^{-1}\right)^{-1}$
  $\boldsymbol{\mu} = \boldsymbol{\Sigma}\left(K^{-1}\mu_0 + g(\boldsymbol{y}) + \boldsymbol{\omega} \circ \beta(\boldsymbol{y})\right)$
  Sample $\boldsymbol{f}^t \sim \mathcal{N}\left(\boldsymbol{\mu}, \boldsymbol{\Sigma}\right)$
end for

4 RELATED WORK

Inference for non-conjugate likelihoods is not a new topic and there have been many works to deal efficiently with the problem.

Scale mixtures of normals. The Gaussian scale-mixture formulation is well known in statistics and have been explored more recently by Gneiting (1997, 1999). Palmer (2006); Palmer et al. (2006) started to generalize it for a machine learning use but did not explore the probability side of the augmentation.

Black-box variational inference. One of the most popular approach for variational inference in the recent years is to optimize the ELBO for an arbitrary model by computing gradients estimates via sampling or quadrature, e.g. Salimbeni et al. (2018); Mohamed et al. (2019). However these methods do not exploit the structure of the model and can be less efficient.

Sampling methods. Sampling is not a popular method for GP models since f is high-dimensional and the posterior is usually highly correlated (Lawrence et al., 2009). But as for many Bayesian models, Hamiltonian Monte Carlo is a good candidate (Titsias et al., 2008).

Likelihood approximation. Jaakkola and Jordan (2000) propose a variational approach purely based on optimization, using the partial convexity of the likelihood. Our method recovers their results, but coming from a probabilistic perspective. We show in Appendix A.5, the equivalence with their approach. Khan and Lin (2017) exploit existing partial conjugacy in the model and rely on the assumption that part of the joint posterior can be rewritten as an exponential family. Their approach is complementary to ours and could be combined for solving more complex models.

Use cases of the augmented model. Different applications of the augmentation technique for specific likelihoods have been explored in multiple papers: Jylänki et al. (2011) applied the augmentation on the Student-t likelihood with Gaussian Processes. Polson et al. (2013) developed an approach with the logistic likelihood, this work was further expanded by Wenzel et al. (2019) to big data. The augmentation done on the Bayesian Support Vector Machine of Polson et al. (2011) and scaled up by Wenzel et al. (2017), is similar to our method but is based on a different augmentation approach. Note that our method covers all these cases exactly but do not rely on any manual derivations.

5 EXPERIMENTS

In this section we answer the following questions empirically:

- How does the Gibbs sampling scheme compare to other sampling methods?
- What is lost in variational inference by approximating an additional variable?
- And what is the gain in speed?

We explore four different cases. We use three regression models with different likelihood functions: a Laplace likelihood, a Student-t likelihood, a new likelihood inspired by the Matern 3/2 kernel (Rasmussen, 2003) and one classification model with a logistic likelihood. All the mathematical details of these augmentations are deferred to the Appendix A.6. For the two first experiments we use a full GP without inducing points to have a cleaner analysis of the effect of the augmentation. For all experiments we use a squared exponential kernel with automatic relevance determination: $k(x, x') = \exp(-\sum_{d=1}^{D}(x_d - x'_d)^2/\theta_d^2)$. For the two first experiments we use datasets from the UCI repository (Dua and Graff, 2017) : the Boston housing dataset ($N = 506, D = 14$) for regression and the Heart dataset ($N = 303, D = 14$) for classification. For the last experiment we use the Protein dataset ($N = 45730, D = 9$) and the Airline dataset ($N = 190K, D = 7$) for regression and the Covtype dataset ($N = 581K, D = 54$) and the SUSY dataset ($N = 5$M, $D = 18$) for classification. We normalize the input features to mean 0 and variance 1.

Likelihood/Method		MH	HMC	**Gibbs**
Logistic	Time/Sample (s)	**0.001**	0.041	0.01
	Lag 1	0.996	0.53	**0.11**
	Gelman	1.38	**1.00**	**1.00**
Student-t	Time/Sample (s)	**0.003**	0.573	0.028
	Lag 1	1.0	0.857	**0.04**
	Gelman	1.51	1.00	**1.00**
Laplace	Time/Sample (s)	**0.002**	0.082	0.028
	Lag 1	0.995	0.931	**0.26**
	Gelman	1.44	1.01	**1.00**
Matern 3/2	Time/Sample (s)	**0.005**	0.15	0.029
	Lag 1	0.997	0.995	**0.05**
	Gelman	1.59	1.10	**1.00**

Table 2. Sampling time and diagnostics of Gibbs Sampling, naive Metropolis-Hastings and Hamiltonian Monte-Carlo. The Gelman test indicates the inter-chain correlation and should be close to 1.

5.1 Gibbs sampling mixing

Our approach leads to a Gibbs sampling algorithm that provides samples from the true posterior of the original model. We compare our method (*Gibbs*) with a naive Metropolis-Hasting algorithm (*MH*) and a Hamiltonian Monte Carlo (*HMC*) sampler (where ϵ and n_{step} are selected via a grid search, see appendix A.7) both implemented in Turing.jl (Ge et al., 2018), with a whitening transformation on the kernel matrix for better mixing. We draw 5 independent chains of 10000 samples for each algorithm. We compare crucial sampling diagnostics among different models: we give the autocorrelation between consecutive samples (lag 1) (as well as the autocorrelation plots for all lags in appendix A.7) to estimate the efficient sample size and the chain intercorrelation via the Gelman test (1 is the optimum) (Brooks and Gelman, 1998). The results are summarized in table 2.

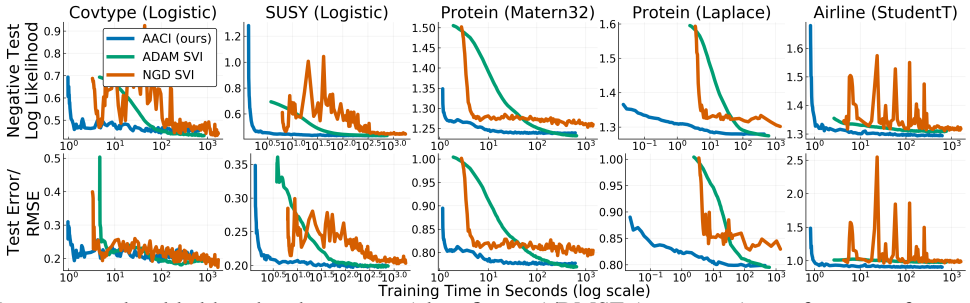
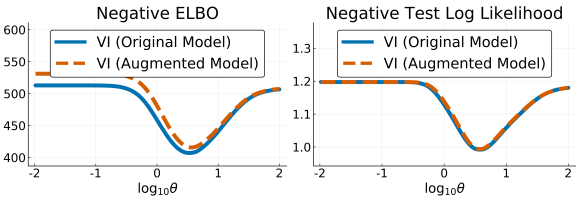
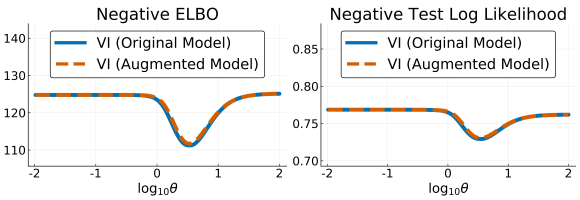


Figure 3. Test negative log-likelihood and test error (classification)/RMSE (regression) as a function of time for different likelihoods.



a) Matern 3/2 Likelihood on the Boston Housing dataset



b) Logistic Likelihood on the Heart dataset

Figure 2. Converged negative ELBO and averaged negative log-likelihood on a held-out dataset in function of the kernel lengthscale, training VI with and without augmentation.

We find that our method has a very low intrachain correlation leading to a high sample efficiency, as well as a low interchain correlation while still being faster than the HMC algorithm. It is even more evident for heavy-tailed likelihood like Student-T or Laplace where HMC can be of more trouble (Betancourt, 2017). Our approach is limited by the $\mathcal{O}(N^3)$ complexity for each sample.

5.2 Augmentation gap

To investigate the effect of augmenting the model when using variational inference, we train the original model using gradient descent and the augmented model until convergence. While we fix the kernel variance at 0.1, we vary the lengthscale θ from 10^{-2} to 10^2. We compare the converged ELBOs as well as the predictive performance on held-out test set. The results for the matern 3/2 and logistic are shown on figure 2, the other likelihoods are show in the appendix A.7. For both shown likelihoods, there is a visible ELBO gap between the augmented model and the original model. However the predictive performance is marginally the same for both models.We can conclude that a potential difference in ELBO values does not affect the prediction performance.

5.3 Convergence speed

To scale our model to large datasets, we use the inducing points technique of Titsias (2009) and we use the stochastic gradient descent approach of Hoffman et al. (2013). We compare our variational approach (Algorithm 1) to using natural gradient descent, (Salimbeni et al., 2018) and ADAM (Hensman et al., 2015) both implemented in GPFlow (Matthews et al., 2017). For all methods we use 200 inducing points determined by k-means++ (Arthur and Vassilvitskii, 2007), minibatches of size 100 and we train the kernel hyperparameters using ADAM (Kingma and Ba, 2014), (the inducing points locations are fixed). We show the predictive performance in function of the training time for multiple likelihoods on figure 3.

Our method is up to two orders of magnitude faster than the state of the art. Moreover, we find that the optimization in our method is more stable (smooth decrease of the loss.

6 CONCLUSION

We proposed a new efficient inference method for GP models that have a super-Gaussian likelihood. Our method builds on an auxiliary variable augmentation that renders the model conditionally conjugate. We showed that in the augmented model, variational inference is up to two orders of magnitude faster and more stable than the state of the art. For small dataset, we proposed a Gibbs sampler that outperforms Hamiltonian Monte Carlo sampling. Previous methods that build on auxiliary variable augmentations (e.g. Wenzel et al., 2019) manually derived the augmentation and inference methods, whereas in our approach the whole procedure is fully automated and works for much more general class of models. Future work may aim on extending our approach to more general models by automatically constructing *hierarchical augmentations* inspired by Galy-Fajou et al. (2019) or Donner and Opper (2018).

References

Abate, J., Choudhury, G. L., and Whitt, W. (2000). An introduction to numerical transform inversion and its application to probability models. In *Computational probability*, pages 257–323. Springer.

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

Arthur, D. and Vassilvitskii, S. (2007). k-means++: The advantages of careful seeding. In *Proceedings of the eighteenth annual ACM-SIAM symposium on Discrete algorithms*, pages 1027–1035. Society for Industrial and Applied Mathematics.

Beckers, T., Kulić, D., and Hirche, S. (2019). Stable gaussian process based tracking control of euler-lagrange systems. *Automatica*, (103):390–397.

Bernstein, S. et al. (1929). Sur les fonctions absolument monotones. *Acta Mathematica*, 52:1–66.

Betancourt, M. (2017). A conceptual introduction to hamiltonian monte carlo. *arXiv preprint arXiv:1701.02434*.

Bishop, C. M. (2006). *Pattern recognition and machine learning*. springer.

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

Brooks, S. P. and Gelman, A. (1998). General methods for monitoring convergence of iterative simulations. *Journal of computational and graphical statistics*, 7(4):434–455.

Cohen, A. M. (2007). *Numerical methods for Laplace transform inversion*, volume 5. Springer Science & Business Media.

Debnath, L. and Bhatta, D. (2014). *Integral transforms and their applications*. Chapman and Hall/CRC.

Devroye, L. (1986). *Nonuniform random variate generation*. Springer-Verlag.

Donner, C. and Opper, M. (2018). Efficient bayesian inference of sigmoidal gaussian cox processes. *The Journal of Machine Learning Research*, 19(1):2710–2743.

Dua, D. and Graff, C. (2017). UCI machine learning repository.

Eleftheriadis, S., Rudovic, O., Deisenroth, M. P., and Pantic, M. (2017). Gaussian process domain experts for modeling of facial affect. *IEEE Transactions on Image Processing*, 26(10):4697–4711.

Galy-Fajou, T., Wenzel, F., Donner, C., and Opper, M. (2019). Multi-class gaussian process classification made conjugate: Efficient inference via data augmentation. *Uncertainty in Artificial Intelligence (UAI)*.

Ge, H., Xu, K., and Ghahramani, Z. (2018). Turing: a language for flexible probabilistic inference. In *International Conference on Artificial Intelligence and Statistics, AISTATS*, pages 1682–1690.

Gneiting, T. (1997). Normal scale mixtures and dual probability densities. *Journal of Statistical Computation and Simulation*, 59(4):375–384.

Gneiting, T. (1999). Radial positive definite functions generated by euclid's hat. *Journal of Multivariate Analysis*, 69(1):88–119.

Hensman, J., Matthews, A., and Ghahramani, Z. (2015). Scalable variational gaussian process classification. *The Journal of Machine Learning Research*.

Hoffman, M. D., Blei, D. M., Wang, C., and Paisley, J. (2013). Stochastic variational inference. *The Journal of Machine Learning Research*, 14(1):1303–1347.

Jaakkola, T. S. and Jordan, M. I. (2000). Bayesian parameter estimation via variational methods. *Statistics and Computing*, 10(1):25–37.

Jylänki, P., Vanhatalo, J., and Vehtari, A. (2011). Robust gaussian process regression with a Student-t likelihood. *Journal of Machine Learning Research*, 12(Nov):3227–3257.

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

Kingma, D. P. and Ba, J. (2014). Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*.

Lawrence, N. D., Rattray, M., and Titsias, M. K. (2009). Efficient sampling for gaussian process inference using control variables. In *Advances in Neural Information Processing Systems*, pages 1681–1688.

Matthews, D. G., Alexander, G., Van Der Wilk, M., Nickson, T., Fujii, K., Boukouvalas, A., León-Villagrá, P., Ghahramani, Z., and Hensman, J. (2017). Gpflow: A gaussian process library using tensorflow. *The Journal of Machine Learning Research*, 18(1):1299–1304.

Merkle, M. (2014). Completely monotone functions: a digest. In *Analytic Number Theory, Approximation Theory, and Special Functions*, pages 347–364. Springer.

Mohamed, S., Rosca, M., Figurnov, M., and Mnih, A. (2019). Monte carlo gradient estimation in machine learning. *arXiv preprint arXiv:1906.10652*.

Palmer, J., Kreutz-Delgado, K., Rao, B. D., and Wipf,

D. P. (2006). Variational em algorithms for non-gaussian latent variable models. In *Advances in neural information processing systems*, pages 1059–1066.

Palmer, J. A. (2006). *Variational and scale mixture representations of non-Gaussian densities for estimation in the Bayesian linear model: Sparse coding, independent component analysis, and minimum entropy segmentation*. PhD thesis, UC San Diego.

Pandit, R. K. and Infield, D. (2018). Comparative analysis of binning and gaussian process based blade pitch angle curve of a wind turbine for the purpose of condition monitoring. *Journal of Physics: Conference Series*, 1102.

Polson, N. G., Scott, J. G., and Windle, J. (2013). Bayesian inference for logistic models using pólya–gamma latent variables. *Journal of the American statistical Association*, 108(504):1339–1349.

Polson, N. G., Scott, S. L., et al. (2011). Data augmentation for support vector machines. *Bayesian Analysis*, 6(1):1–23.

Rasmussen, C. E. (2003). *Gaussian processes in machine learning*. Springer.

Ridout, M. S. (2009). Generating random numbers from a distribution specified by its laplace transform. *Statistics and Computing*, 19(4):439.

Salimbeni, H., Eleftheriadis, S., and Hensman, J. (2018). Natural gradients in practice: Non-conjugate variational inference in gaussian process models. *roceedings of the International Conference on Artificial Intelligence and Statistics (AISTATS)*.

Schoenberg, I. J. (1938). Metric spaces and completely monotone functions. *Annals of Mathematics*, pages 811–841.

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

Titsias, M. K., Lawrence, N., and Rattray, M. (2008). Markov chain monte carlo algorithms for gaussian processes. *Inference and Estimation in Probabilistic Time-Series Models*, 9.

Wang, C. and Blei, D. M. (2013). Variational inference in nonconjugate models. *Journal of Machine Learning Research*, 14(Apr):1005–1031.

Wenzel, F., Galy-Fajou, T., Deutsch, M., and Kloft, M. (2017). Bayesian nonlinear support vector machines for big data. In *Joint European Conference on Machine Learning and Knowledge Discovery in Databases*, pages 307–322. Springer.

Wenzel, F., Galy-Fajou, T., Donner, C., Kloft, M., and Opper, M. (2019). Efficient gaussian process classification using Pòlya-gamma data augmentation. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 33, pages 5417–5424.

Widder, D. V. (1946). *The Laplace transform*. Princeton university press.

Yeh, J. (2006). *Real analysis: theory of measure and integration second edition*. World Scientific Publishing Company.

A APPENDIX

A.1 Proof of theorem 2

Theorem 2 states:

Theorem. *The **complete conditional distributions** of the augmented model presented in Section 3.1 are given by*

$$\begin{aligned} p(\omega_i|f_i, y_i) =& \pi_\varphi\left(\omega_i|\|h(f_i, y_i)\|_2\right), \\ p(\boldsymbol{f}|\boldsymbol{y}, \boldsymbol{\omega}) =& \mathcal{N}\left(\boldsymbol{f}|\boldsymbol{\mu}, \boldsymbol{\Sigma}\right), \end{aligned}$$

where $\boldsymbol{\Sigma} = \left(\mathrm{diag}\left(2\boldsymbol{\omega} \circ \gamma(\boldsymbol{y})\right) + K^{-1}\right)^{-1}$ *and* $\boldsymbol{\mu} = \boldsymbol{\Sigma}\left(g(\boldsymbol{y}) + \boldsymbol{\omega} \circ \beta(\boldsymbol{y}) + K^{-1}\boldsymbol{\mu}_0\right)$*,* $\circ$ *denotes the Hadamard product and the function* $h(\cdot)$ *is given by the form of likelihood (see Eq.5).*

Proof: For the full conditional on $\boldsymbol{f}$:

$$\begin{aligned} p(\boldsymbol{f}|\boldsymbol{y}, \boldsymbol{\omega}) \propto& p(\boldsymbol{y}|\boldsymbol{f}, \boldsymbol{\omega})p(\boldsymbol{f}) \\ \propto& \exp\left[g(\boldsymbol{y})^\top \boldsymbol{f} + (\beta(\boldsymbol{y}) \circ \boldsymbol{\omega})^\top \boldsymbol{f} - \boldsymbol{f}^\top \mathrm{diag}(\gamma(\boldsymbol{y}) \circ \boldsymbol{\omega})\boldsymbol{f} - \frac{1}{2}\boldsymbol{f}^\top K^{-1}\boldsymbol{f}\right] \\ \propto& \exp\left[\left(g(\boldsymbol{y}) + \beta(\boldsymbol{y}) \circ \boldsymbol{\omega}\right)^\top \boldsymbol{f} - \boldsymbol{f}^\top\left[\mathrm{diag}(\gamma(\boldsymbol{y}) \circ \boldsymbol{\omega}) + \frac{1}{2}K^{-1}\right]\boldsymbol{f}\right]. \end{aligned}$$

We get immediately a multivariate normal distribution with $-\frac{1}{2}\Sigma^{-1} = -\mathrm{diag}(\gamma(\boldsymbol{y}) \circ \boldsymbol{\omega}) + \frac{1}{2}K^{-1}$ and $\Sigma^{-1}\mu = g(\boldsymbol{y}) + (\beta(\boldsymbol{y}) \circ \boldsymbol{\omega})$. Which corresponds to the result shown in equation (11).

For the augmented variable ω_i:

$$\begin{aligned} p(\omega_i|y_i, f_i) \propto& p(y_i|f_i, \omega_i)p(\omega_i) \\ \propto& \exp\left(-\|h(y_i, f_i)\|_2^2\omega_i\right)\pi_\varphi(\omega_i|0) \\ =& \pi_\varphi(\omega_i|\|h(y_i, f_i\|_2). \end{aligned}$$

Note that the equation 9 gives the normalization constant directly $\varphi(\|h(y_i, f_i)\|_2^2)$ directly. QED.

A.2 Computation of the moments and cumulants for the augmentation variable

Given the general class of distribution $\pi_\varphi(\omega|c)$ described in Section 3.1, moments and cumulants can be easily computed: The k-th moment of a distribution can be computed by taking the k-th derivative of the moment generating function (equivalent to a negative Laplace transform) at $t = 0$. For example for the first moment:

$$\begin{aligned} \mathbb{E}_{\pi_\varphi(\omega|c)}\left[\omega\right] =& \left.\frac{d\mathcal{L}\left\{\pi_\varphi(\omega|c)\right\}(-t)}{dt}\right|_{t=0} \\ =& \left.\frac{d}{dt}\left[\mathcal{L}\left[\frac{e^{-c^2\omega}\pi_\varphi(\omega|0)}{\varphi(c^2)}\right](-t)\right]\right|_{t=0} \\ =& \left. -\frac{1}{\varphi(c^2)}\frac{d}{dt}\left[\mathcal{L}\left[\pi_\varphi(\omega|b, 0)\right](t + c^2)\right]\right|_{t=0} \\ =& \left. -\frac{1}{\varphi(c^2)}\frac{d\varphi\left(t + c^2\right)}{dt}\right|_{t=0} \\ =& \left. -\frac{d\log\varphi(t)}{dt}\right|_{t=c^2} \\ =& -\frac{\varphi'(c^2)}{\varphi(c^2)} = \overline{\omega} \end{aligned}$$

More generally the k-th moment m_k is defined as :

$$m_k = (-1)^k \frac{1}{\varphi(c^2)} \left. \frac{d^k \varphi(t)}{dt^k} \right|_{c^2}$$

And the cumulants κ_k are computed using the cumulant generating function (log of the moment generating function)

$$\kappa_k = (-1)^k \left. \frac{d^k \log \varphi(t)}{dt^k} \right|_{t=c^2}$$

A.3 Algorithm for the sparse case

Algorithm 3 Augmented Stochastic Variational Inference

Input: Data $(\mathbf{X}, \boldsymbol{y})$, GP model $p(\boldsymbol{y}|\boldsymbol{f}, \boldsymbol{u})$, kernel k
Output: Approximate posterior $q(\boldsymbol{u}) = \mathcal{N}(\boldsymbol{u}|\boldsymbol{m}, \boldsymbol{S})$
Find inducing points inputs Z via k-means
Compute kernel matrices : K_Z, $\kappa = K_{XZ}K_Z^{-1}$
for iteration $t = 1, 2, \ldots,$ **do**
`# Local updates:`
Sample minibatch $\mathcal{B} \subseteq \{1, \ldots, n\}$
for $i \in \mathcal{B}$ **do**
$c_i = \sqrt{\mathbb{E}_{q(f)}\left[h(f_i, y_i)^2\right]}$
$\overline{\omega}_i = \mathbb{E}_{q(\omega_i)}\left[\omega_i\right] = -\varphi'(c_i^2)/\varphi(c_i^2)$
end for
`# Natural gradient updates (CAVI):`
$\widetilde{\boldsymbol{S}} = \left(\kappa^\top \mathrm{diag}\left(2\overline{\boldsymbol{\omega}} \circ \gamma(\boldsymbol{y})\right)\kappa + K_Z^{-1}\right)^{-1}$
$\widetilde{\boldsymbol{m}} = \widetilde{\boldsymbol{S}}\left(K_Z^{-1}\mu_0 + \kappa^\top\left(g(\boldsymbol{y}) + \overline{\boldsymbol{\omega}} \circ \beta(\boldsymbol{y})\right)\right)$
$\{\boldsymbol{m}, \boldsymbol{S}\} \leftarrow (1 - \rho^{(t)})\{\boldsymbol{m}, \boldsymbol{S}\} + \rho^{(t)}\{\widetilde{\boldsymbol{m}}, \widetilde{\boldsymbol{S}}\}$
end for

$\rho^{(t)}$ is an arbitrary learning rate respecting the Robbins-Monroe condition.

A.4 ELBO Analysis

A.4.1 Full ELBO

$$\begin{aligned}\mathrm{ELBO} = &\sum_{i=1}^{N} \mathbb{E}_{q(f_i,\omega_i)}\left[\log p(y_i|f_i,\omega_i)\right] \\ &- \mathrm{KL}[q(f)||p(f)] - \sum_{i=1}^{N} \mathrm{KL}[q(\omega_i)||p(\omega_i)]\end{aligned}$$

$$\begin{aligned}\mathbb{E}_q\left[\log p(y_i|f_i,\omega_i,\theta)\right] &= \log C(\theta) + g(y_i,\theta)\mathbb{E}_{q(f)}\left[f\right] - \mathbb{E}_{q(f)}\left[h(f_i,y_i)^2\right]\mathbb{E}_{q(\omega_i)}\left[\omega_i\right] \\ &= \log C(\theta) + g(y_i,\theta)m_i - \left(\alpha(y_i) - \beta(y_i)m_i + \gamma(y_i)\left(m_i^2 + S_{ii}\right)\right)\overline{\omega}_i \\ \mathrm{KL}[q(f)||p(f)] &= \frac{1}{2}\left[\log\frac{|K|}{|\boldsymbol{S}|} - N + \mathrm{tr}(K^{-1}\boldsymbol{S}) + (\boldsymbol{\mu}_0 - \boldsymbol{m})^\top K^{-1}(\boldsymbol{\mu}_0 - \boldsymbol{m})\right] \\ \mathrm{KL}[q(\omega_i)||p(\omega_i)] &= -\mathbb{E}_{q(\omega_i)}\left[c_i^2\omega_i\right] - \log\varphi(c_i^2) = -c_i^2\overline{\omega}_i - \log\varphi(c_i^2)\end{aligned}$$

Note that we can take the derivatives of the ELBO and set them to 0 to recover exactly the updates in algorithm 1.

A.4.2 Analysis of the optima

By setting c_i^2 as a function of $\boldsymbol{m}$ and $\boldsymbol{S}$ (and setting $\boldsymbol{\mu}_0$ to 0 for simplicity) we can get an ELBO only depending of the variational parameters of $\boldsymbol{f}$.

$$\text{ELBO}(\boldsymbol{m}, \boldsymbol{S}) = C + g^\top \boldsymbol{m} + \frac{1}{2}\left(\underbrace{\log|\boldsymbol{S}| - \text{tr}(K^{-1}\boldsymbol{S}) - \boldsymbol{m}^\top K^{-1}\boldsymbol{m}}_{\text{ELBO}_1}\right) + \sum_i \underbrace{\log\varphi(m_i^2 + S_{ii})}_{\text{ELBO}_2}$$

It is easy to show that ELBO_1 is jointly concave in $\boldsymbol{m}$ and $\boldsymbol{S}$ with a short matrix analysis. However ELBO_2 is more complex : $m_i^2 + S_{ii}$ is jointly convex in $\boldsymbol{m}$ and $\boldsymbol{S}$, $\phi(r)$ is by definition convex as well, however $\phi(m_i^2 + S_{ii})$ is neither jointly convex or concave in $\boldsymbol{m}$ and $\boldsymbol{S}$. It is therefore impossible to guarantee that there is a global optima, however the CAVI updates guarantee us a local optima.

A.4.3 ELBO Gap

For a fixed $q(f)$ we can compare the ELBO of the original model $\mathcal{L}_{std}(q(f))$ and the augmented model $\mathcal{L}_{aug}(q(f)q(\omega))$. It is then straightforward to compute the difference between the two :

$$\begin{aligned}
\Delta\mathcal{L} =& \mathcal{L}_{std}(q(f)) - \mathcal{L}_{aug}(q(f)q(\omega)) \\
=& \mathbb{E}_{q(f)}\left[\log p(y,f) - \log q(f) - \mathbb{E}_{q(\omega)}\left[p(y,f,\omega) - \log q(f)q(\omega)\right]\right] \\
=& \mathbb{E}_{q(f)q(\omega)}\left[-\log\frac{p(y,f,\omega)}{p(y,f)} + \log q(\omega)\right] \\
=& \mathbb{E}_{q(f)q(\omega)}\left[-\log p(\omega|y,f) + \log q(\omega)\right] \\
=& \mathbb{E}_{q(\omega)}\left[\log q(\omega) - \mathbb{E}_{q(f)}\left[\log p(\omega|y,f)\right]\right] \\
=& -c^2\mathbb{E}_{q(\omega)}\left[\omega\right] + \mathbb{E}_{q(\omega)}\left[\log \text{PG}(\omega|1,0)\right] - \log\varphi(c^2) \\
& + \mathbb{E}_{q(f)}\left[f^2\right]\mathbb{E}_{q(\omega)}\left[\omega\right] - \mathbb{E}_{q(\omega)}\left[\log \text{PG}(\omega|1,0)\right] + \mathbb{E}_{q(f)}\left[\log\varphi(f^2)\right] \\
=& -c^2 m - \log\varphi(c^2) + \mathbb{E}_{q(f)}\left[f^2\right] m + \mathbb{E}_{q(f)}\left[\log\varphi(f^2)\right]
\end{aligned}$$

Replacing with the optimal $q^*(\omega) = \frac{e^{-c^2\omega}p(\omega)}{\varphi(c^2)}$ with $c^2 = \mathbb{E}_{q(f)}\left[f^2\right]$

$$\Delta\mathcal{L}^* = -\log\varphi(c^2) + \mathbb{E}_{q(f)}\left[\log\varphi(f^2)\right]$$

A.4.4 Sparse ELBO

When using the inducing points approach the ELBO becomes:

$$\begin{aligned}
\text{ELBO} = & \sum_{i=1}^N \mathbb{E}_{q(f_i,u_i,\omega_i)}\left[\log p(y_i|f_i,u_i,\omega_i)\right] \\
& - \text{KL}[q(u)||p(u)] - \sum_{i=1}^N \text{KL}[q(\omega_i)||p(\omega_i)]
\end{aligned}$$

$$\begin{aligned}
\mathbb{E}_q\left[\log p(y_i|f_i,\omega_i,\theta)\right] &= \log C(\theta) + g(y_i,\theta)\mathbb{E}_{q(f,u)}\left[f\right] - \mathbb{E}_{q(f,u)}\left[h(f_i,y_i)^2\right]\mathbb{E}_{q(\omega_i)}\left[\omega_i\right] \\
&= \log C(\theta) + g(y_i,\theta)(\kappa^\top \boldsymbol{m})_i - \left(\alpha(y_i) - \beta(y_i)(\kappa^\top \boldsymbol{m})_i + \gamma(y_i)\left((\kappa^\top \boldsymbol{m})_i^2 + (\kappa^\top \boldsymbol{S}\kappa)_{ii}\right)\right)\overline{\omega}_i \\
\mathrm{KL}[q(f)||p(f)] &= \frac{1}{2}\left[\log\frac{|K|}{|\boldsymbol{S}|} - N + \mathrm{tr}(K^{-1}\boldsymbol{S}) + (\boldsymbol{\mu}_0 - \boldsymbol{m})^\top K^{-1}(\boldsymbol{\mu}_0 - \boldsymbol{m})\right] \\
\mathrm{KL}[q(\omega_i)||p(\omega_i)] &= -\mathbb{E}_{q(\omega_i)}\left[c_i^2\omega_i\right] - \log\varphi(c_i^2) = -c_i^2\overline{\omega}_i - \log\varphi(c_i^2)
\end{aligned}$$

A.5 Proof of equivalence between Jaakkola bound and data augmentation

Jaakkola and Jordan (2000) proposed an approach purely based on optimization. They are assuming $\log p(y|f)$ contains a part convex in f^2: $\log p(y|f) = \log p_{convex}(f) + \log p_{non-convex}(f,y)$. Using convexity properties they are creating a bound with a Taylor expansion to the first order around an additional variable c^2:

$$\log p_c(f) \geq \log p_c(c) + \frac{d\log p_c(c)}{dc^2}(f^2 - c^2)$$

Putting it back in the full ELBO, they are now getting a quadratic part in f, analytically differentiable, and they just need to optimize the additional variables $\{c_i\}$. Merkle (2014) shows that any completely monotone function is log-convex, i.e. $\log\varphi(r)$ is convex. Therefore we can replace $\log p_c(c)$ by $\log\varphi(r)$ to recover our model in the context of variational inference. Note that the converse does not hold, therefore the complete monotonicity is a stronger assumption.

A.6 Likelihoods used for the experiments

We detail all likelihoods used for the experiments and their formulation as in equation (4).

Laplace Likelihood : $\mathrm{Laplace}(y|f,\beta) = \frac{1}{2\beta}\exp\left(-\frac{|f-y|}{\beta}\right)$

Logistic Likelihood : $p(y|f) = \sigma(yf) = \frac{e^{yf/2}}{2\cosh(|f|/2)}$

Student-T Likelihood : $p(y|f) = \frac{\Gamma((\nu+1)/2)}{\Gamma(\nu/2)\sqrt{\pi\nu}}\left(1 + \frac{(y-f)^2}{\nu}\right)^{-(\nu+1)/2}$

Matern 3/2 Likelihood : $p(y|f) = \frac{4\rho}{\sqrt{3}}\left(1 + \frac{\sqrt{3(y-f)^2}}{\rho}\right)\exp\left(-\frac{\sqrt{3(y-f)^2}}{\rho}\right)$

Likelihood	$C(\theta)$	$g(y,\theta)$	$\|\|h(y,f,\theta)^2\|\|_2^2$	$\alpha(y)$	$\beta(y)$	$\gamma(y)$	$\varphi(r)$
Laplace	$(2\beta)^{-1}$	0	$(y-f)^2$	y^2	$2y$	1	$e^{-\sqrt{r}/\beta}$
Logistic	2^{-1}	$y/2$	f^2	0	0	1	$\cosh^{-1}(\sqrt{r}/2)$
Student-T	$\Gamma((\nu+1)/2)/(\Gamma(\nu)\sqrt{\pi\nu})$	0	$(y-f)^2$	y^2	$2y$	1	$(1+\frac{r}{\nu})^{-(\nu+1)/2}$
Matern 3/2	$4\rho/\sqrt{3}$	0	$(y-f)^2$	y^2	$2y$	1	$(1+\frac{\sqrt{3r}}{\rho})e^{-\sqrt{3r}/\rho}$

A.7 Extra figures

A.7.1 Autocorrelation plots

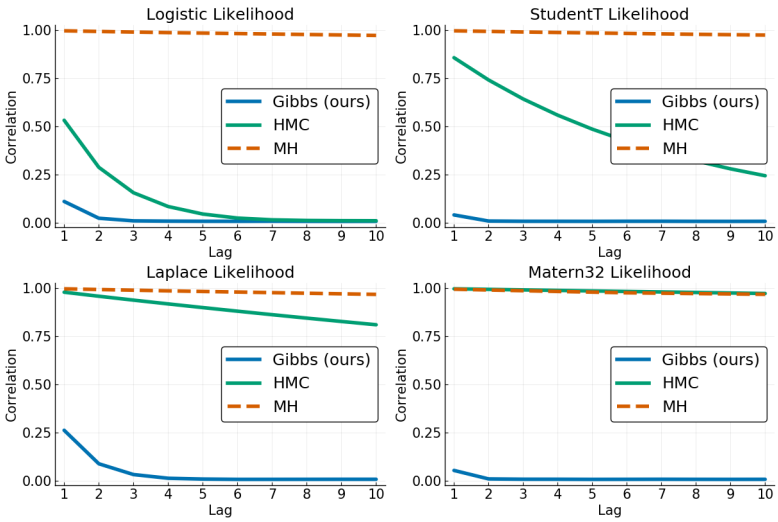


Figure 4. Auto-correlation plots for differents with lags from 1 to 10

A.7.2 HMC Results

ϵ/n_{step}		1	2	5	10
0.01	Time/Sample (s)	0.037	0.045	0.077	0.133
	Lag 1	0.999	0.993	0.978	0.963
	Gelman	3.14	1.02	1.00	2.05
0.05	Time/Sample (s)	0.036	0.046	0.080	0.12
	Lag 1	0.999	0.998	**0.931**	0.948
	Gelman	1.72	1.18	1.01	3.25
0.1	Time/Sample (s)	0.033	0.042	0.073	0.13
	Lag 1	0.997	0.996	0.998	0.994
	Gelman	1.11	1.04	1.27	2.71

Table 3. HMC results for the Laplace likelihood

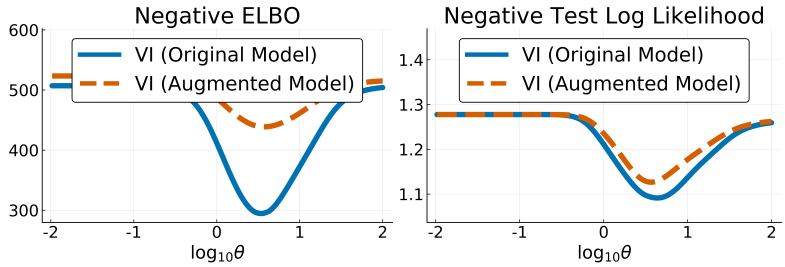
ϵ/n_{step}		1	2	5	10
0.01	Time/Sample (s)	0.675	0.110	0.177	0.251
	Lag 1	0.999	0.999	0.997	0.993
	Gelman	3.14	1.74	1.11	1.02
0.05	Time/Sample (s)	0.148	0.192	0.336	0.573
	Lag 1	0.997	0.993	0.962	**0.857**
	Gelman	1.10	1.02	1.00	1.00
0.1	Time/Sample (s)	0.142	0.193	0.337	NA
	Lag 1	0.993	0.976	0.864	NA
	Gelman	1.03	1.01	1.00	NA

Table 4. HMC results for the Student-T likelihood

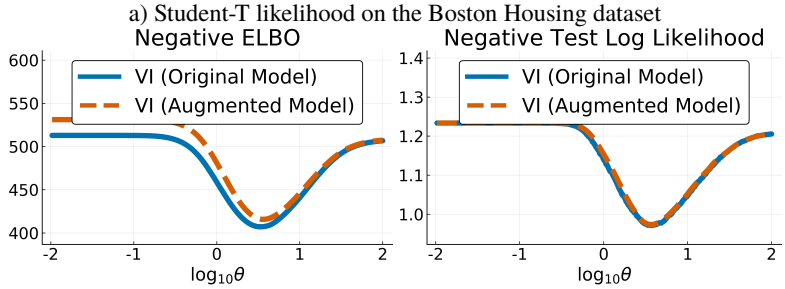
ϵ/n_{step}		1	2	5	10
0.01	Time/Sample (s)	0.009	0.013	0.021	0.041
	Lag 1	0.999	0.999	0.998	0.994
	Gelman	3.19	1.68	1.12	1.02
0.05	Time/Sample (s)	0.011	0.014	0.025	0.41
	Lag 1	0.998	0.994	0.968	0.871
	Gelman	1.11	1.03	1.00	1.00
0.1	Time/Sample (s)	0.011	0.014	0.024	0.048
	Lag 1	0.994	0.979	0.875	**0.532**
	Gelman	1.02	1.01	1.00	1.00

Table 5. HMC Results for the Logistic likelihood

A.7.3 ELBO difference



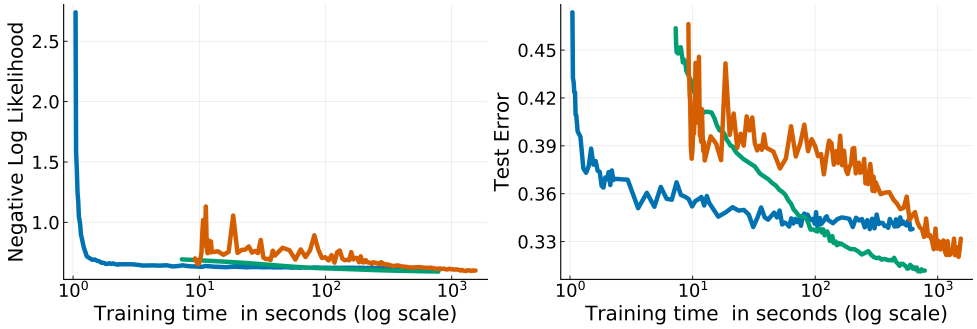
a) Student-T likelihood on the Boston Housing dataset



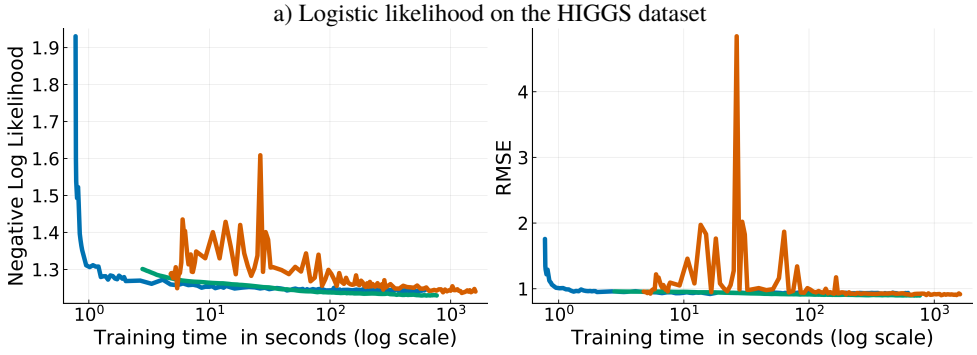
b) Laplace likelihood on the Boston Housing dataset

Figure 5. Converged negative ELBO and averaged negative log-likelihood on a held-out dataset in function of the RBF kernel lengthscale, training VI with and without augmentation.

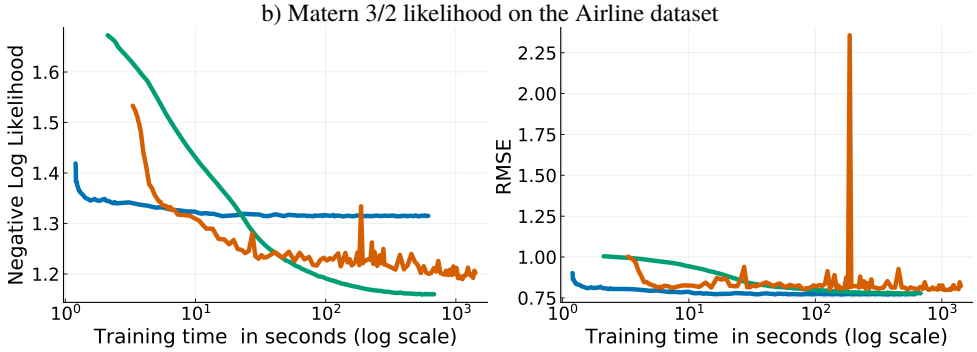
A.7.4 Convergence speed



a) Logistic likelihood on the HIGGS dataset



b) Matern 3/2 likelihood on the Airline dataset



c) Student-T likelihood on the Protein dataset

Figure 6. Supplementary convergence plots

6

Flexible and Efficient Inference with Particles for the Variational Gaussian Approximation

This last published work is different from the previous chapters. Instead of focusing on the representation of the model, we aim at changing the variational distribution representation. The original motivation behind this work was to answer the question: Can we fit a full Gaussian variational distribution to a target distribution without matrix inverses, log-determinant, or second-order derivative computations? The answer resulted in a particle approach: we parametrize the distribution with an arbitrary number of points in the variable domain instead of the mean and covariance. Although the method might not be a state-of-the-art approach for variational inference, it brings insights concerning convergence speed and accuracy of the given posterior.

Authors:
Théo Galy-Fajou,[1], Valerio Perrone,[2], Manfred Opper[1,3]
[1]TU Berlin, Germany, [2]Amazon Web Services, [3]University of Birmingham

Details:
Type: Journal article
Submitted: June 2021
Accepted: July 2021
URL: `https://www.mdpi.com/1099-4300/23/8/990`
Journal: Entropy (Special edition on Approximate Bayesian Inference)

Contributions:

For an explanation of the terms see the Contributor Roles Taxonomy (CReditT)

	T.G-F.	V.P.	M.O.
Conceptualization	✓		✓
Methodology	✓	✓	✓
Formal Analysis	✓		
Software	✓		
Investigation	✓		
Writing - Original Draft	✓	✓	✓
Writing - Review & Editing	✓	✓	✓
Supervision			✓
Funding Acquisition			✓

entropy

MDPI

Article

Flexible and Efficient Inference with Particles for the Variational Gaussian Approximation

Théo Galy-Fajou [1,*], Valerio Perrone [2] and Manfred Opper [1,3]

[1] Artificial Intelligence Group, Technische Universität Berlin, 10623 Berlin, Germany; manfred.opper@tu-berlin.de
[2] Amazon Web Services, 10969 Berlin, Germany; vperrone@amazon.com
[3] Centre for Systems Modelling and Quantitative Biomedicine, University of Birmingham, Birmingham B15 2TT, UK
* Correspondence: galy-fajou@tu-berlin.de

Abstract: Variational inference is a powerful framework, used to approximate intractable posteriors through variational distributions. The de facto standard is to rely on Gaussian variational families, which come with numerous advantages: they are easy to sample from, simple to parametrize, and many expectations are known in closed-form or readily computed by quadrature. In this paper, we view the Gaussian variational approximation problem through the lens of gradient flows. We introduce a flexible and efficient algorithm based on a linear flow leading to a particle-based approximation. We prove that, with a sufficient number of particles, our algorithm converges linearly to the exact solution for Gaussian targets, and a low-rank approximation otherwise. In addition to the theoretical analysis, we show, on a set of synthetic and real-world high-dimensional problems, that our algorithm outperforms existing methods with Gaussian targets while performing on a par with non-Gaussian targets.

Keywords: variational inference; Gaussian; particle flow; variable flow

Citation: Galy-Fajou, T.; Perrone, V.; Opper, M. Flexible and Efficient Inference with Particles for the Variational Gaussian Approximation. *Entropy* **2021**, *23*, 990. https://doi.org/10.3390/e23080990

Academic Editor: Pierre Alquier

Received: 22 June 2021
Accepted: 21 July 2021
Published: 30 July 2021

1. Introduction

Representing uncertainty is a ubiquitous problem in machine learning. Reliable uncertainties are key for decision making, especially in contexts where the trade-off between exploitation and exploration plays a central role, such as Bayesian optimization [1], active learning [2], and reinforcement learning [3]. While Bayesian inference is a principled tool to provide uncertainty estimation, computing posterior distributions is intractable for many problems of interest. Most sampling methods struggle to scale up to large datasets [4], while the diagnosis of convergence is not always straightforward [5]. On the other hand, *Variational Inference* **(VI)** methods can rely on well-understood optimization techniques and scale well to large datasets, at the cost of an approximation quality depending heavily on the assumptions made. The Gaussian family is by far the most popular variational approximation used in VI [6,7]. This is for several reasons. First, Gaussian variational families are easy to sample from, reparametrize, and marginalize. Second, they are easily amenable to diagonal covariance approximations, making them scalable to high dimensions. Third, most expectations are either easily computable by quadrature or Monte Carlo integration, or known in closed-form.

A large body of work covers different approaches to optimize the *Variational Gaussian Approximation* **(VGA)**, with the speed of convergence and the scalability in dimensions as the main concerns. From the perspective of convergence speed, the major bottleneck when computing gradients with stochastic estimators is the estimator variance [8]. *Particle-based methods* with deterministic paths do not have this issue, and have been proven to be highly successful in many applications [9–11]. However, can we use a particle-based

algorithm to compute a VGA? If so, what are its properties and is it competitive with other VGA methods?

In this paper, we attempt to answer these questions by introducing the *Gaussian Particle Flow* **(GPF)**, a framework to approximate a Gaussian variational distribution with particles. GPF is derived from a continuous-time flow, where the necessary expectations over the evolving densities are approximated by particles. The complexity of the method grows quadratically with the number of particles but linearly with the dimension, remaining compatible with other approximations such as structured mean-field approximations. Using the same dynamics, we also derive a stochastic version of the algorithm, *Gaussian Flow* **(GF)**. To show convergence, we prove the decrease in an empirical version of the free energy that is valid for a finite number of particles. For the special case of D–dimensional Gaussian target densities, we show that $D+1$ particles are enough to obtain convergence to the true distribution. We also find, for this case, that convergence is exponentially fast. Finally, we compare our approach with other VGA algorithms, both in fully controlled synthetic settings and on a set of real-world problems.

2. Related Work

The goal of Bayesian inference is to carry out computations with the posterior distribution of a latent variable $x \in R^D$ given some observations y. By Bayes theorem, the posterior distribution is $p(x|y) = \frac{p(y|x)p(x)}{p(y)}$, where $p(y|x)$ and $p(x)$ are, respectively, the likelihood and the prior distribution. Even if the likelihood and the prior are known analytically, marginalizing out high-dimensional variables in the product $p(y|x)p(x)$ in order to compute quantities such as $p(y)$ is typically intractable. *Variational Inference* **(VI)** aims to simplify this problem by turning it into an optimization one. The intractable posterior is approximated by the closest distribution within a tractable family, with closeness being measured by the *Kullback-Leibler* **(KL)** divergence, defined by

$$\mathrm{KL}\left[q(x)||p(x)\right] = \mathbb{E}_q[\log q(x) - \log p(x)],$$

where $\mathbb{E}_q[f(x)] = \int f(x)q(x)dx$ denotes the expectation of f over q. Denoting by $\mathcal{Q}$ a family of distributions, we look for

$$\arg\min_{q\in\mathcal{Q}} \mathrm{KL}\left[q(x)||p(x|y)\right].$$

Since $p(y)$ is not computable in an efficient way, we equivalently minimize the upper bound $\mathcal{F}$:

$$\mathrm{KL}[q(x)||p(x|y)] \leq \mathcal{F}[q] = -\mathbb{E}_q[\log p(y|x)p(x)] - \mathbb{H}_q, \tag{1}$$

where $\mathbb{H}_q$ is the entropy of q $(-\mathbb{E}_q[\log q(x)])$. Here, $\mathcal{F}$ is known as the variational free energy and $-\mathcal{F}$ is known as the Evidence Lower BOund (ELBO). A diverse set of approaches to perform VI with Gaussian families $\mathcal{Q}$ have been developed in the literature, which we review in the following.

2.1. The Variational Gaussian Approximation

The VGA is the restriction of $\mathcal{Q}$ to be the family of multivariate Gaussian distributions $q(x) = \mathcal{N}(m, C)$, where $m \in \mathbb{R}^D$ is the mean and $C \in \{A \in \mathbb{R}^{D\times D} | x^\top Ax \geq 0, \forall x \in \mathbb{R}^D\}$ is the covariance matrix, for which the free energy is found to be

$$\mathcal{F}[q] = -\frac{1}{2}\log|C| + \mathbb{E}_q[\varphi(x)]. \tag{2}$$

where $\varphi(x) = -\log(p(y|x)p(x))$. A standard descent algorithm based on gradients of Equation (2) with respect to variational parameters m, C give rise to some issues. First, naively computing the gradient of the expectation with respect to the covariance matrix

C involves unwanted second derivatives of $\varphi(x)$ [12], which may not be available or may be computationally too expensive in a *black-box* setting. Second, the gradient of the entropy term $\mathbb{H}_q$ entails inverting a non-sparse matrix, which we would like to avoid for higher-dimensional cases. Finally, the positive-definiteness of the covariance matrix leads to non-trivial constraints on parameter updates, which can lead to a slowdown of convergence or, if ignored, to instabilities in the algorithm.

To solve these issues, a variety of approaches have been proposed in the literature. If we focus on factorizable models, we can make a simplification: for problems with likelihoods that can be rewritten as $p(y|x) = \prod_{d=1}^{D} p(y|x_d)$, the number of independent variational parameters is reduced to $2D$ [12,13]. In this special case, the Gaussian expectations in the free energy (2) split into a sum of 1-dimensional integrals, which can be efficiently computed by using numerical quadrature methods. To extend to the general case, gradients of the free energy are estimated by a stochastic sampling approach, which also forms the starting point of our method. This relies on the so-called *reparametrization trick*, where the expectation over the parameter-dependent variational density q_θ is replaced by an expectation over a fixed density q^0 instead. This facilitates the gradient computation because unwanted derivatives of the type $\nabla_\theta q_\theta(x)$ are avoided. For the Gaussian case, the reparametrization trick is a linear transformation of an arbitrary D dimensional Gaussian random variable $x \sim q_\theta(x)$ in terms of a D-dimensional Gaussian random variable $x^0 \sim q^0 = \mathcal{N}(m^0, C^0)$:

$$x = \Gamma(x^0 - m^0) + m, \tag{3}$$

where $\Gamma \in \mathbb{R}^{D\times D}$ and $m \in \mathbb{R}^D$ are the variational parameters. We assume that the covariance C^0 is not degenerate and, for simplicity, we set it as the identity. For instance, the gradient of the expectation given q over a function f given the mean m becomes $\nabla_m \mathbb{E}_q[f(x)] = \mathbb{E}_{q^0}\left[\nabla_m f(\Gamma(x^0 - m^0) + m)\right]$. This can be simply proved by using the reparametrization (3) inside the integral and passing the gradient inside; for more details, see [14].

Given this representation, the free energy is easily obtained as a function of the variational parameters:

$$\mathcal{F}(q) = -\log|\Gamma| + \mathbb{E}_{q^0}\Big[\varphi(\Gamma(x^0 - m^0) + m)\Big]. \tag{4}$$

Other representations are possible. Challis and Barber [13] and Ong et al. [15] use a different reparametrization with a factorized structure of the covariance $C = \Gamma^\top\Gamma + \mathrm{diag}(d)$, where $\Gamma \in \mathbb{R}^{D\times P}$ and $d \in \mathbb{R}^D$, with $P \leq D$ is the rank of $\Gamma^\top\Gamma$. Other representations assume special structures of the precision matrix $\Lambda = C^{-1}$, which allow you to enforce special properties, such as sparsity in [16,17].

In general, these methods tend to scale poorly with the number of dimensions, as one needs to optimize $D(D+3)/2$ parameters. The (structured) *Mean-Field* **(MF)** [18,19] approach imposes independence between variables in the variational distribution. The number of variational parameters is then $2D$, but covariance information between dimensions is lost.

2.2. Natural Gradients

Besides the issue of expectations, more efficient optimizations directions, beyond ordinary gradient descent, have been considered. These can help to deal with constraints such as those given for the covariance matrix. Natural gradients [20] are a special case of Riemannian gradients and utilize the specific Riemannian manifold structure of variational parameters. They can often deal with constraints of parameters (such as the positive definiteness of the covariance), accelerate inference, and improve the convergence of algorithms. The application of such advanced gradient methods typically requires an estimate of the inverse Fisher information matrix as a preconditioner of ordinary gradients. Khan and Nielsen [21] and Lin et al. [22] propose a solution that requires extra second derivatives of the log–posteriors. Salimbeni et al. [23] developed an automatic process to

compute these without the second derivatives but with instability issues. Lin et al. [17] solved these issues by using geodesics on the manifold of parameters, at the price of having to compute inverse matrices as well as Hessians.

2.3. Particle-Based VI

Stochastic gradient descent methods compute expectations (and gradients) at each time step with new independent Monte Carlo samples drawn from the current approximation of the variational density. Particle-based methods for variational inference *draw samples only once* at the beginning of the algorithm instead. They iteratively construct transformations of an initial random variable (having a simple tractable density) where the transformed density leads to the decrease and finally to the minimum of the variational free energy. The iterative approach induces a deterministic temporal flow of random variables which depends on the current density of the variable itself. Using an approximation by the empirical density (which is represented by the positions of a set of 'particles') one obtains a flow of interacting particles which converges asymptotically to an empirical approximation of the desired optimal variational density.

The most popular approach is *Stein Variational Gradient Descent* **(SVGD)** [24], which computes a nonparametric transformation based on the kernelized Stein discrepancy [9]. SVGD has the advantage of not being restricted to a parametric form of the variational distribution. However, using standard distance-based kernels like the squared exponential kernel ($k(x, y) = \exp(-\|x - y\|_2^2/2)$) can lead to underestimated covariances and poor performance in high dimensions [11,25]. Hence, it is interesting to develop particle approaches that approximate the VGA. We provide a more thorough comparison between our method and SVGD in Section 3.6.

2.4. GVA in Bayesian Neural Networks

There has been increased interest in making *Bayesian Neural Networks* **(BNN)** by adding priors to Neural Networks parameters. The true form of the posterior is unknown but VGA has been used due to its ease of use and scalability with the number of dimensions (typically $D \gg 10^5$). Most of the aforementioned methods apply to BNN, but techniques have been specifically tailored with BNN in mind. [26] use the low-rank structure of [13] but exploit the *Local Reparametrization Trick*, where each datapoint y_i gets a different sample from q in order to reduce the stochastic gradient estimator variance. *Stochastic Weight Averaging-Gaussian* **(SWAG)** [27], in which a set of particles obtained via stochastic gradient descent represent a low-rank Gaussian distribution, approximating the true posterior with a prior posterior produced by the network's regularization. While easy to implement, SWAG does not allow you to incorporate an explicit prior, and the resulting distribution does not derive from a principled Bayesian approach.

2.5. Related Approaches

The closest approach to our proposed method is the *Ensemble Kalman Filter* **(EKF)** [28]. It assumes that the posterior is computed in a sequential way, where, at each time step, only single (or smaller batches) of data observations, represented by their likelihoods, become available. An ensemble of particles, representing a Gaussian distribution is iteratively updated with every new batch of observations. EKF allows us to work on high-dimensional problems with a limited amount of particles but is restricted to factorizable likelihoods for which a sequential representation is possible. While EKF maintains a representation of a Gaussian posterior, it is not clear how this relates to the goal of minimizing the free energy or the KL divergence.

3. Gaussian (Particle) Flow

We introduce *Gaussian Particle Flow* **(GPF)** and *Gaussian Flow* **(GF)**, two computationally tractable approaches, to obtain a *Variational Gaussian Approximation* **(VGA)**. In the following, we derive deterministic linear dynamics, which decreases the variational free

energy. We additionally give some variants with a *Mean-Field* **(MF)** approach and prove theoretical convergence guarantees.

In the following, $\frac{d(\cdot)}{dt}$ indicates the total derivative given time, $\frac{\partial(\cdot)}{\partial t}$ partial derivatives given time, $\nabla_x(\cdot)$ gradients given a vector x.

3.1. Gaussian Variable Flows

We next discuss an alternative approach to generate the desired transformation of random variables, leading from a simple (prior) Gaussian density to a more complex Gaussian, which minimizes the variational free energy. It is based on the idea of *variable flows*, i.e., recursive deterministic transformations of the random variables defined by a mapping $x^{n+1} = x^n + \epsilon f^n(x^n)$ where $f^n : \mathbb{R}^D \to \mathbb{R}^D$. Well-known examples of flows are *Normalizing Flows* [29], where f^n are bijections, or *Neural ODEs* [30] where $f^n = f$ is defined by a neural network and x^0 is the input. For simplicity, we will consider small changes $\epsilon \to 0$ and work with flows in the continuous-time limit ($t = n\epsilon$), which follow a system of *Ordinary Differential Equation* **(ODE)**. For the Gaussian case, in the spirit of the reparametrization trick (3), we choose a linear corresponding map f and write

$$\frac{dx^t}{dt} = f^t(x^t) = A^t(x^t - m^t) + b^t, \tag{5}$$

where A^t is a matrix and $m^t \doteq \mathbb{E}_{q^t}[x]$ (which is no longer interpreted as an independent variational parameter). When the initial random variable x^0 is Gaussian distributed, the vectors x^t are also Gaussian for any t. To construct a flow that decreases the free energy over time, we can either compute the time derivative of the specific free energy (2) induced by the ODE (5), or simply derive the general result valid for smooth maps f (see, e.g., [24]). To be self contained, we briefly repeat the main steps: We first compute the change of the free energy in terms of the time derivative of q^t:

$$\begin{aligned}
\frac{d\mathcal{F}[q^t]}{dt} &= \frac{d}{dt}\int q^t(x)\big(\log q^t(x) + \varphi(x)\big)dx \\
&= \int \frac{\partial q^t(x)}{\partial t}\big(\log q^t(x) + \varphi(x)\big)dx + \int q^t(x)\left(\frac{\partial q^t(x)}{\partial t}\frac{1}{q^t(x)} + \frac{\partial \varphi(x)}{\partial t}\right)dx \\
&= \int \frac{\partial q^t(x)}{\partial t}\big(\log q^t(x) + \varphi(x)\big)dx
\end{aligned}$$

where we have used the fact that $\int \frac{\partial q^t(x)}{\partial t}dx = \frac{d}{dt}\int q^t(x)dx = 0$ and $\frac{\partial \varphi(x)}{\partial t} = 0$. We next use the *continuity equation* for the density

$$\frac{\partial q^t(x)}{\partial t} = -\nabla_x \cdot \big(q^t(x) f^t(x)\big),$$

related to the deterministic flow to obtain

$$\begin{aligned}
\frac{d\mathcal{F}[q^t]}{dt} &= \int \nabla_x \cdot \big(q^t(x) f^t(x)\big)\big(\log q^t(x) + \varphi(x)\big)dx \\
&= -\int \big(q^t(x) f^t(x)\big) \cdot \nabla_x \big(\log q^t(x) + \varphi(x)\big)dx \\
&= \int \big(\nabla_x \cdot (q^t(x) f^t(x)) + q^t(x) f^t(x) \cdot \nabla_x \varphi(x)\big)dx \\
&= \int \nabla_x q^t(x) \cdot f^t(x) + q^t(x) f^t(x) \cdot \nabla_x \varphi(x) dx \\
&= -\mathbb{E}_{q^t}\big[\nabla_x \cdot f^t(x) - f^t(x) \cdot \nabla_x \varphi(x)\big]
\end{aligned}$$

where we have applied Green's identity twice and used the fact that $\lim_{x\to\infty} q_t(x) = 0$. Specializing to the linear flow (5), we obtain

$$\frac{d\mathcal{F}[q^t]}{dt} = -\mathrm{tr}[A^t(A^t_\star)^\top] - (b^t)^\top b^t_\star, \tag{6}$$

where

$$\begin{aligned} A^t_\star &\doteq I - \mathbb{E}_{q^t}\left[\nabla_x \varphi(x)(x-m^t)^\top\right] \\ b^t_\star &\doteq -\mathbb{E}_{q^t}[\nabla_x \varphi(x)] \end{aligned} \tag{7}$$

Equation (6) represents the change in the free energy $\mathcal{F}$ for an infinitesimal change in the variables x given by the flow (5). Obviously, the simplest choices

$$A^t \equiv A^t_\star \qquad b^t \equiv b^t_\star \tag{8}$$

lead to a decrease in the free energy $\frac{d\mathcal{F}[q^t]}{dt} \leq 0$. More detailed derivations are given in Appendix A. Additionally, *equality* only happens, when

$$\begin{aligned} I - \mathbb{E}_q\left[\nabla_x \varphi(x)(x-m)^\top\right] &= 0 \\ \mathbb{E}_q[\nabla_x \varphi(x)] &= 0 \end{aligned} \tag{9}$$

Using Stein's lemma [31], we can show that these fixed-point solutions are equal to the conditions for the optimal variational Gaussian distribution solution given in [12]. In Appendix C, we show that our parameter updates can be interpreted as a Riemannian gradient descent method for the free energy (4). This is based on the metric introduced by ([20], Theorem 7.6) as an efficient technique for learning the mixing matrix in models of blind source separation. This gradient should not be confused with the so-called *natural gradient* obtained by pre-multiplying with the inverse Fischer-information matrix.

Of course, there are other choices for A^t and b^t, which lead to a decrease in the free energy and the same fixed-point equations. In Section 3.6, we discuss how SVGD, with a linear kernel, can lead to the same fixed points but with different dynamics.

3.2. From Variable Flows to Parameter Flows

Before we introduce the particle algorithm, we show that the results for the variable flow can also be converted into a temporal change of the parameters Γ^t, m^t, as defined for Equation (3). From this, a corresponding *Gaussian Flow* **(GF)** algorithm can be easily derived. By differentiating the parametrisation $x^t = \Gamma^t(x^0 - m^0) + m^t$ (with m^t now considered as free variational parameter) with respect to time t and using (5), we obtain

$$\frac{dx^t}{dt} = \frac{d\Gamma^t}{dt}(x^0 - m^0) + \frac{dm^t}{dt} = A^t(x^t - m^t) + b^t \tag{10}$$

By inserting $x^t = \Gamma^t(x^0 - m^0) + m^t$ into the right hand side of (10), and using the optimal parameters from (7), we obtain

$$\begin{aligned} \frac{d\Gamma^t}{dt} &= \Gamma^t - \mathbb{E}_{q^0}\left[\nabla_x \varphi(x^t)(x^0 - m^0)^\top\right]\Gamma^t(\Gamma^t)^\top \\ \frac{dm^t}{dt} &= -\mathbb{E}_{q^0}\left[\nabla_x \varphi(x^t)\right] \end{aligned} \tag{11}$$

Note that the expectations are over the probability distribution of the initial random variable x^0. Discretizing Equations (11) in time, and estimating the expectations by drawing independent samples from the fixed Gaussian q^0 at each time step, we obtain our GF algorithm to minimize the variational free energy in the space of Gaussian densities. We summarize the steps of GF in Algorithm 1. Remarkably, this scheme differs from previous VGA algorithms with Riemannian gradients based on the Fisher information

metric (see, e.g., [17,32]) because no *matrix inversions* or *second order derivatives* of the function φ are required.

GF also allows for the computation of a low-rank VGA by enforcing $\Gamma \in \mathbb{R}^{D\times K}$ and $x^0 \in \mathbb{R}^K$. This algorithm scales linearly in the number of dimensions and quadratically in the rank K of the covariance.

It is interesting to note that the reverse construction of a variable flow from a parameter flow is, in general, not possible. This would require the ability to eliminate all variational parameters and the initial variables x^0 in the resulting differential equation for x^t, and replace them with functions of x^t alone. For instance, if we eliminate the initial variables x^0 in terms of $(\Gamma^t)^{-1}$ and x^t the algorithm of [14], the resulting expression still depends on Γ^t.

3.3. Particle Dynamics

The main idea of the particle approach is to approximate the Gaussian density q^t in (7) by the empirical distribution

$$\hat{q}^t \doteq \frac{1}{N}\sum_{i=1}^{N}\delta(x - x_i^t) \tag{12}$$

computed from N samples x_i^t, $i = 1,\dots,N$. These are initially sampled from the density q^0 at time $t = 0$ and are then propagated using the discretized dynamics of the ODE (5):

$$\frac{dx_i^t}{dt} = -\eta_1^t \mathbb{E}_{\hat{q}^t}[\nabla_x \varphi(x)] - \eta_2^t \hat{A}^t (x_i^t - \hat{m}^t) \tag{13}$$

where

$$\hat{A}^t = I - \frac{1}{N}\sum_{i=1}^{N}\nabla_x\varphi(x)(x_i^t - \hat{m}^t)^\top$$

$$\hat{b}^t = \frac{1}{N}\sum_{i=1}^{N}\nabla_x\varphi(x_i^t), \qquad \hat{m}^t = \frac{1}{N}\sum_{i=1}^{N}x_i^t$$

where η_1^t and η_2^t are learning rates (We further comment on the use of different optimization schemes in Section 4.4). Note that although $\mathbb{E}_{\hat{q}^t}\left[\nabla_x\varphi(x)(x-\hat{m}^t)^\top\right]$ is a $D \times D$ matrix, changing the matrix multiplication order leads to a computational complexity of $\mathcal{O}(N^2D)$ with a storage complexity of $\mathcal{O}(N(N+D))$, since neither the empirical covariance matrix or A^t need to be explicitly computed.

Relaxation of Empirical Free Energy and Convergence

We have shown that the continuous-time dynamics (10) of the random variables leads to a decay of the free energy $\mathcal{F}(q^t)$ with time t. Assuming that the free energy is bounded from below, one might conjecture that this property would imply the convergence of the particle algorithm to a fixed point when learning rates are sufficiently small such that the discrete-time dynamics are approximated well by the continuous limit. Unfortunately, the finite number N of particles poses an extra problem. The definition of the free energy $\mathcal{F}(q)$ by the KL–divergence (1) for continuous random variables such as assumes that both $q(\cdot)$ and $p(\cdot|y)$ are densities with respect to the Lebesgue measure. Hence, $\mathcal{F}(\hat{q})$ is not defined if we take $q \equiv \hat{q}$, (12) as the empirical distribution of the finite particle approximation. Nevertheless, we define a finite N *approximation* to the Gaussian free energy, which is also then found to decay under the finite N dynamics. Let us first assume that $N > D$ and define

$$\tilde{\mathcal{F}}(\hat{q}^t) \doteq -\frac{1}{2}\log|\hat{C}^t| + \mathbb{E}_{\hat{q}^t}[\varphi(x)] \tag{14}$$

with the empirical covariance matrix

$$\hat{C}^t = \frac{1}{N}\sum_{i=1}^{N}\left(x_i^t - m^t\right)\left(x_i^t - m^t\right)^\top \tag{15}$$

The definition (14) is chosen in such way that in the large N limit, when the empirical distribution $\hat{q}^t$ converges to a Gaussian distribution q^t, we will also obtain the convergence of the approximation (14) to $\mathcal{F}(q^t)$. It can be shown (see Appendix B) that $\frac{d\tilde{\mathcal{F}}(\hat{q}^t)}{dt} \leq 0$, with equality only at the fixed points of the dynamics.

In applications of our particle method to high-dimensional problems, the limitations of computational power may force us to restrict particle numbers to be smaller than the dimensionality D. For $N < D+1$, the empirical covariance C^t will be singular, and typically contain only $N-1$ non-zero eigenvalues, which leads to the $-\log\left|\hat{C}\right| = \infty$ and makes Equation (14) meaningless. We resolve this issue through a regularisation of the log–determinant term in (14), replacing all zero eigenvalues of $\hat{C}$ by the values 1, i.e., $\lambda_i = 0 \to \tilde{\lambda}_i = 1$. We show in Appendix B that the free energy still decays, provided that the dynamics of the particles stay the same. This regularisation step can be formally stated as a replacement of the empirical covariance (15) in (14) by

$$\hat{C}^t \to \hat{C}^t + \sum_{i:\lambda_i^t=0} e_i^t (e_i^t)^\top$$

where $e_i^t = i$th eigenvector of $\hat{C}^t$.

3.4. Algorithm and Properties

The algorithm we propose is to sample N particles $\{x_1^0, \ldots, x_N^0\}$ where $x_i^0 \in \mathbb{R}^D$ from q^0 (which can be centered around the MAP for example), and iteratively optimize their positions using Equation (13). Once convergence is reached, i.e., $\frac{d\mathcal{F}}{dt} = 0$, we can easily make predictions using the converged empirical distribution $\hat{q}(x) = \frac{1}{N}\sum_{i=1}^{N}\delta(x - x_i)$, where δ is the Dirac delta function, or, alternatively, the Gaussian density it represents, i.e., $q(x) = \mathcal{N}(m, C)$, where $m = \frac{1}{N}\sum_{i=1}^{N} x_i$ and $C = \frac{1}{N}\sum_{i=1}^{N}(x_i - m)(x_i - m)^\top$. To draw samples from $\hat{q}$, no inversions of the empirical covariance C are needed, as we can obtain new samples by computing:

$$x = \frac{1}{\sqrt{N}}\sum_{i=1}^{N}(x_i - m)\circ\xi_i + m, \tag{16}$$

where ξ_i are i.i.d. normal variables: $\xi_i \sim \mathcal{N}(0, \mathbb{I}_D)$. This can be shown by defining D, the deviation matrix, a matrix which columns equal to $D_i = \frac{x_i - m}{\sqrt{N}}$. We naturally have $DD^\top = C$ which makes D the Cholesky decomposition of C.

All the inference steps are summarized in Algorithm 2 and an illustration in two dimensions is provided in Figure 1.

We summarize the principal points of our approach:

- Gradients of expectations have zero variance, at the cost of a bias decreasing with the number of particles and equal to zero for Gaussian target (see Theorem 1);
- It works with noisy gradients (when using subsampling data, for example);
- The rank of the approximated covariance C is $\min(N-1, D)$. When $N \leq D$, the algorithm can be used to obtain a low-rank approximation.
- The complexity of our algorithm is $\mathcal{O}(N^2 D)$ and storing complexity is $\mathcal{O}(N(N+D))$. By adjusting the number of particles used, we can control the performance trade-off;
- GPF (and GF) are also compatible with any kind of structured MF (see Section 3.5);
- Despite working with an empirical distribution ,we can compute a surrogate of the free energy $\mathcal{F}(q)$ to optimize hyper-parameters, compute the lower bound of the log-evidence, or simply monitor convergence.

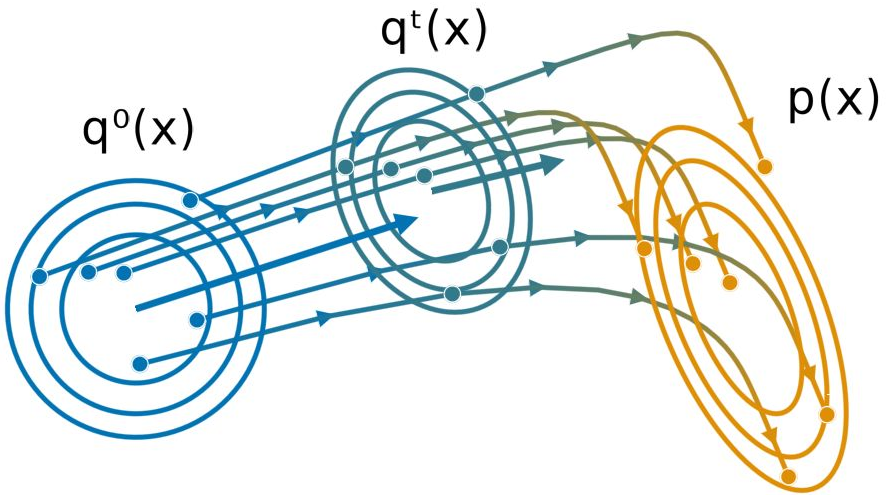


Figure 1. Illustration of the Gaussian Particle Flow algorithm, with $q^0(x)$ and $p(x)$ representing the initial and target distribution respectively. Particles are iteratively moved according to the gradient flow starting from $q^0(x)$, approximating a new Gaussian distribution $q^t(x)$ at each iteration t.

Algorithm 1: Gaussian Flow (GF)

Input: Number of samples N, initial distribution $q^0 = \mathcal{N}(\mu^0, \Gamma^0(\Gamma^0)^\top)$, target $p(x) \propto e^{-\varphi(x)}$, learning rates η_1^t, η_2^t
Output: Variational dist. $q(x) = \mathcal{N}(\mu, \Gamma\Gamma^\top)$
for t *in* $0 : T$ **do**

$\{x_i^0\}_{i=1}^N \sim q^0$ `# Sample N initial particles from q^0`
$x_i = \Gamma^t(x_i^0 - \mu^0) + \mu^t, \ \forall i$ `# Reparametrize`
$g_i = \nabla_x \varphi(x_i), \ \forall i$ `# Compute gradients`
$\mu^{t+1} = \mu^t - \eta_1^t \frac{1}{N} \sum_{i=1}^N \varphi(x_i)$ `# Update μ`
$A = \frac{1}{N} \sum_i g_i (x_i^0 - \mu^0)^\top (\Gamma^t)^\top$ `# Compute matrix`
$\Gamma^{t+1} = \Gamma^t - \eta_2^t A \Gamma^t$ `# Update Γ`

Algorithm 2: Gaussian Particle Flow (GPF)

Input: Number of particles N, initial distribution q^0, target $p(x) \propto e^{-\varphi(x)}$, learning rates η_1^t, η_2^t
Output: Empirical dist. $q(x) = \frac{1}{N} \sum_{i=1}^N \delta_{x,x_i}$
Init: *Sample N particles from* $q^0 : \{x_i^0\}_{i=1}^N$
for t *in* $0 : T$ **do**

$g_i = \nabla_x \varphi(x_i^t), \ \forall i$ `# Compute gradients`
$m = \frac{1}{N} \sum_i x_i, \quad \overline{g} = \frac{1}{N} \sum_i g_i$ `# Compute means`
$A = \frac{1}{N} \sum_i g_i (x_i^t - m)^\top - I$ `# Compute matrix`
$x_i^{t+1} = x_i^t - \eta_1^t \overline{g} - \eta_2^t A (x_i^t - m), \ \forall i$ `# Update particles`

3.4.1. Relaxation of Empirical Free Energy

The definition of the free energy $\mathcal{F}(q)$ from the KL–divergence (1) for a continuous random variables assumes that both $q(\cdot)$ and $p(\cdot|y)$ are densities with respect to the Lebesgue measure. Hence, it is not *a priori* clear that a specific *approximation* $\mathcal{F}(\hat{q}^t)$, based on an empirical distribution $\hat{q}^t(x) \doteq \frac{1}{N} \sum_{i=1}^N \delta(x - x_i^t)$ with a *finite number of particles* N, will decrease under the particle flow. Thus we may not be able to guarantee convergence to a fixed point for finite N. Luckily, as we show in Appendix D, we find that:

$$\frac{d\mathcal{F}(\hat{q}_t)}{dt} = \frac{d(\mathbb{E}_{\hat{q}_t}[\varphi(x)] - \frac{1}{2}\log|C^t|)}{dt} \leq 0. \tag{17}$$

For $N < D+1$, the empirical covariance C^t will typically contain $N-1$ non-zero eigenvalues and lead to $-\log|C| = \infty$, making Equation (17) meaningless. We resolve this issue by introducing a *regularized free energy* $\widetilde{\mathcal{F}}$ where $\log|C^t|$ is replaced by $\sum_{i:\lambda_i>0}\log\lambda_i$ where $\{\lambda_i\}_{i=1}^D$ are the eigenvalues of C^t. We show in Appendix D that, given the dynamics from Equation (5), $\widetilde{\mathcal{F}}$ is also guaranteed to not increase over time. It can, therefore, be used as a regularized proxy for the true $\mathcal{F}$ and used to optimize over hyper-parameters or to monitor convergence. Note that similar proofs exist for SVGD [33] and were proven to be highly non-trivial.

3.4.2. Dynamics and Fixed Points for Gaussian Targets

We illustrate our method by some exact theoretical results for the dynamics and the fixed points of our algorithm when *the target is a multivariate Gaussian density*. While such targets may seem like a trivial application, our analysis could still provide some insight into the performance for more complicated densities.

Theorem 1. *If the target density $p(x)$ is a D-dimensional multivariate Gaussian, only $D+1$ particles are needed for Algorithm 2 to converge to the exact target parameters.*

Proof. The proof is given in Appendix E. □

Theorem 2. *For a target $p(x) = \mathcal{N}(x \mid \mu, \Lambda^{-1})$, i.e., with precision matrixΛ, where $x \in \mathbb{R}^D$, and $N \geq D+1$ particles, the continuous time limit of Algorithm 2 will converge exponentially fast for both the mean and the trace of the precision matrix:*

$$\begin{aligned} m^t - \mu &= e^{-\Lambda t}(m^0 - \mu), \\ \mathrm{tr}((C^t)^{-1} - \Lambda) &= e^{-2t}\mathrm{tr}(\left(C^0\right)^{-1} - \Lambda), \end{aligned}$$

where m^t and C^t are the empirical mean and covariance matrix at time t and $\exp(-\Lambda t)$ is the matrix exponential.

Proof. The proof is given in Appendix F. □

Our result shows that convergence of the mean m^t directly depends on Λ. However, we can also precondition the gradient on m by C^t, i.e., using the natural gradient approximation in the Fisher sense, and eventually get rid of the dependency on Λ when $(C^t)^{-1} \approx \Lambda$.

The exponential relaxation of fluctuations also manifests itself in the decay of the free energy towards its minimum. For the Gaussian target, the free energy exactly separates into two terms corresponding to the mean and fluctuations. We can write $\mathcal{F}(m^t, C^t) = \frac{1}{2}(m^t-\mu)^\top\Lambda(m^t-\mu) + \frac{D}{2} + \mathcal{F}_{fl}(C^t)$, where the nontrivial fluctuation part (subtracted by its minimum) is given by

$$\mathcal{F}_{fl}(C^t) = -\frac{1}{2}\log|C^t| + \frac{1}{2}\mathrm{tr}(\Lambda C^t - I).$$

We can show that

$$-\lim_{t\to\infty}\frac{d\ln\mathcal{F}_{fl}(C^t)}{dt} \geq 4,$$

indicating an asymptotic decrease in $\mathcal{F}_{fl}(C^t)$ faster than e^{-4t}, independent of the target. We can also prove the finite time bound

$$\mathcal{F}_{fl}(C^t) \leq \mathcal{F}_{fl}(C^0) e^{-\left[\frac{2t}{\mathrm{tr}(\Lambda^{-1})(\mathrm{tr}(\Lambda)+|\mathrm{tr}((C^0)^{-1}-\Lambda)|)}\right]}.$$

The degenerate case $\mathbf{N < D + 1}$

Additionally, we can show the following result for the fixed points:

Theorem 3. *Given a D-dimensional multivariate Gaussian target density $p(x) = \mathcal{N}(x|\mu, \Sigma)$, using Algorithm 2 with $N < D + 1$ particles, the empirical mean converges to the exact mean μ. The $N - 1$ non-zero eigenvalues of C^t converge to a subset of the target covariance Σ spectrum. Furthermore, the **global minimum** of the regularised version $\widetilde{\mathcal{F}}$ of the free energy (17) corresponds to the **largest** eigenvalues of Σ.*

Proof. The proof is given in Appendix G. □

This result suggests that C^t might typically converge to an optimal low-rank approximation of Σ. We show an empirical confirmation in Section 4.2 for this conjecture. This suggests that it makes sense to apply our algorithm to high-dimensional problems even when the number of particles is not large. If the target density has significant support close to a low-dimensional submanifold, we might still obtain a reasonable approximation.

3.5. Structured Mean-Field

For high-dimensional problems, it may be useful to restrict the variational Gaussian approximation to the posterior to a specific structure via a structured mean-field approximation. In this way, spurious dependencies between variables that are caused by finite-sample effects could be explicitly removed from the algorithms. This is most easily incorporated in our approach by splitting a given collection of latent variables x into M disjoint subsets $x^{(i)}$. We reorder the vector indices in such a way that the first components correspond to $x^{(1)}, x^{(2)}$, and so on. Hence, we obtain $x = \{x^{(1)}, x^{(2)}, \ldots, x^{(M)}\}$. A structured mean-field approach is enforced by imposing a block matrix structure for the update matrix $A_{MF} = A_{(1)} \oplus \cdots \oplus A_{(M)}$, where $\oplus$ is the direct sum operator. It is easy to see that this construction corresponds to a related block structure of the Γ matrix in Equation (3). This means that the subsets of the random vectors are modeled as independent. Hence, when the number of particles grows to infinity, one recovers the fixed-point equations for the optimal MF structured Gaussian variational approximation from our approach. As previously, as the number of particles grows to infinity, we recover the optimal MF Gaussian variational approximation. Note that using a structured MF does not change the complexity of the algorithm but requires fewer particles to obtain a full-rank solution.

3.6. Comparison with SVGD

Given the similarities with the SVGD methods [24],one could question the differences of our approach. The model proposed by [10] using a *linear kernel* $k(x, x') = x^\top x' + 1$ has similar properties to our approach. The variable update becomes:

$$\begin{aligned}\frac{dx}{dt} &= \frac{1}{N}\sum_{i=1}^{N}(-k(x_i, x)\nabla\varphi(x_i) + \nabla_{x_i}K(x_l, x_i)) \\ &= \mathbb{E}_{\hat{q}}\left[I - \nabla\varphi(x)x^\top\right]x - \mathbb{E}_{\hat{q}}[\nabla\varphi(x)]\end{aligned}$$

The fixed points are

$$\begin{aligned}0 =& \mathbb{E}_{\hat{q}}[\nabla\varphi(x)] \\ I =& \mathbb{E}_{\hat{q}}\left[\nabla\varphi(x)x^\top\right] = \mathbb{E}_{\hat{q}}\left[\nabla\varphi(x)(x-m)^\top\right]\end{aligned}$$

where the last equality holds since $\mathbb{E}_{\hat{q}}[\nabla\varphi(x)] = 0$. This is the same as our algorithm fixed points (9). Similarly to Theorem 1, $D+1$ particles will converge to the exact D-dimensional multivariate Gaussian target. However, the generated flows are different. The main difference is that *we normalize our flow via the L_2 norm*, whereas [10] rely on the *reproducing kernel Hilbert space (RKHS) norm*, i.e., $\|\varphi\|_k^2 = \varphi^\top K^{-1}\varphi$ where $\varphi_i = \varphi(x_i)$ and $K_{ij} = k(x_i, x_j)$. For a full introduction on RKHS, we recommend [34]. Remarkably, centering the particles on the mean, namely, using the modified linear kernel $k(x, x') = (x-m)^\top(x'-m)+1$, leads to the same dynamics. Additionally, when using SVGD, there is no direct possibility of computing the current KL divergence between the variational distribution and the target, unless some values are accumulated [35]. There is also no clear theory explaining what happens when the number of particles is smaller than the number of dimensions, for both distance-based kernels and the linear kernel.

4. Experiments

We now evaluate the efficiency of GPF and GF. First, given a Gaussian target, we compare the convergence of our approach with popular VGA methods, which are all described in Section 2. Second, we evaluate the effect of varying the number of particles for both Gaussian targets and non-Gaussian targets, especially with a low-rank covariance. Then, we evaluate the efficiency of our algorithm on a range of real-world binary classification problems through a Bayesian logistic regression model and a series of BNN on the MNIST dataset.

All the Julia [36] code and data used to reproduce the experiments are available at the Github repository: https://github.com/theogf/ParticleFlow_Exp (accessed on 27 July 2021).

4.1. Multivariate Gaussian Targets

We consider a 20-dimensional multivariate Gaussian target distribution. The mean is sampled from a normal Gaussian $\mu \sim \mathcal{N}(0, I_D)$ and the covariance is a dense matrix defined as $\Sigma = U\Lambda U^\top$, where U is a unitary matrix and Λ is a diagonal matrix. Λ is constructed as $\log_{10}(\Lambda_{ii}) = \frac{\log_{10}(\kappa)(i-1)}{D-1} - 1$ where κ is the condition number, i.e., $\kappa = \Lambda_{\max}/\Lambda_{\min}$. This means that, for $\kappa = 1$, we obtain a $\Sigma = 0.1\mathbb{I}$, and for $\kappa = 100$, we obtain eigenvalues ranging uniformly from 0.1 to 10 in log-space.

We compare GPF and GF to the state-of-the art methods for VGA described in Section 2, namely *Doubly Stochastic VI* **(DSVI)** [14], *Factor Covariance Structure* **(FCS)** [15] with rank $p = D$, *iBayes Learning Rule* **(IBLR)** [17] with a full-rank covariance and their Hessian approach, and Stein Variational Gradient Descent with both a linear kernel (**Linear SVGD**) [10] and a squared-exponential kernel (**Sq. Exp. SVGD**) [24]. For all methods, we set the number of particles or, alternatively, the number of samples used by the estimator, as $D+1$, and use standard gradient descent ($x^{t+1} = x^t + \eta\varphi^t(x^t)$) with a learning rate of $\eta = 0.01$ for all particle methods. We use RMSProp [37] with a learning rate of 0.01 for all stochastic methods. We run each experiment 10 times with 30,000 iterations, and plot the average error on the mean and the covariance with one standard deviation. For GPF, we additionally evaluate the method with and without using natural gradients for the mean (i.e., pre-multiplying the averaged gradient with C^t), indicated, respectively, with a dashed and solid line. Figure 2 reports the L_2 norm of the difference between the mean and covariance with the true posterior over time for the target condition number $\kappa \in \{1, 10, 100\}$.

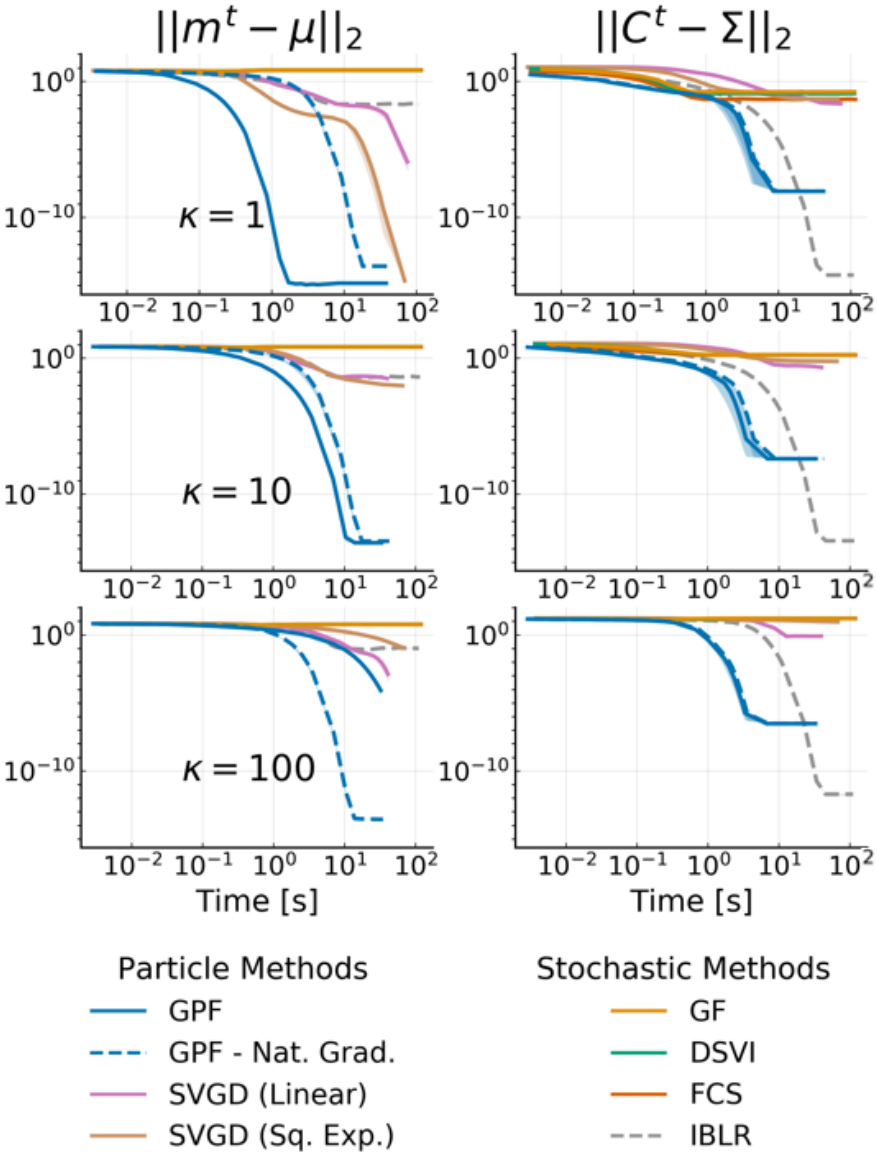


Figure 2. L^2 norm of the difference between the target mean μ (left side) and target covariance Σ (right side) with the inferred variational parameters m^t and C^t against time for 20-dimensional Gaussian targets with condition number κ. We use $D + 1$ particles/samples and show the mean over 10 runs as well as the 68% credible interval. Methods with dashed curves use natural gradients on the mean. Note that DSVI, GF and FCS are overlapping and are, at this scale, indistinguishable from one another.

As Theorem 1 predicts, GPF converges exactly to the true distribution, regardless of the target. GF and other methods based on stochastic estimators cannot obtain the same precision as their accuracy is penalized by the gradient noise. IBLR approximate the covariance perfectly, despite the stochasticity of its estimator; however IBLR needs to compute the true Hessian at each step. When using a Hessian approximation instead, IBLR performed just like DSVI; the true benefit of IBLR appears when second-order functions are computed, which is naturally intractable in high-dimensions. SVGD with a linear kernel, achieves a good performance but is highly unstable: most of the runs (ignored here) diverge. This is due to the dot computation $x^\top x$ which can become extremely high, especially for non-centered data. For this reason, we do not consider this method for the later experiments. SVGD with a sq. exp. kernel obtains a good estimate for the mean but fails to approximate the covariance.

Perhaps surprisingly, GF does not perform much better than DSVI or FCS. This is potentially due to *the benefit of Riemannian gradients being canceled by the gradient noise* [38] providing a strong argument for particle-based methods over stochastic estimators.

Remarkably, we also confirm Theorem 2, that the convergence speed of C^t is independent of the target Σ, while the convergence speed of m^t has this dependency unless the natural gradient is used (see the dashed curves). The case $\kappa = 1$ highlights that *natural gradient do not necessarily improve convergence speed*.

4.2. Low-Rank Approximation for Full Gaussian Targets

We explore the effect of the number of particles for both Gaussian and non-Gaussian targets. We use the same Gaussian target from the previous experiment in 50 dimensions with a full-rank covariance determined by their condition number $\kappa = \frac{\lambda_{\max}}{\lambda_{\min}}$. The covariance eigenvalues λ_i in log-space range uniformly from 0.1 to 0.1κ. For a given target multivariate Gaussian, we vary the number of particles from 2 to $D + 1$ and look at the absolute difference of $|\mathrm{tr}(C - \Sigma)|$. The results in $D = 50$, as well as the corresponding predictions (in dashed-black), from Theorem 3, are shown on Figure 3.

The empirical results perfectly match the theoretical predictions, confirming that, for Gaussian targets, the particles determine a low-rank approximation whose spectrum is equal to the largest eigenvalues from the target.

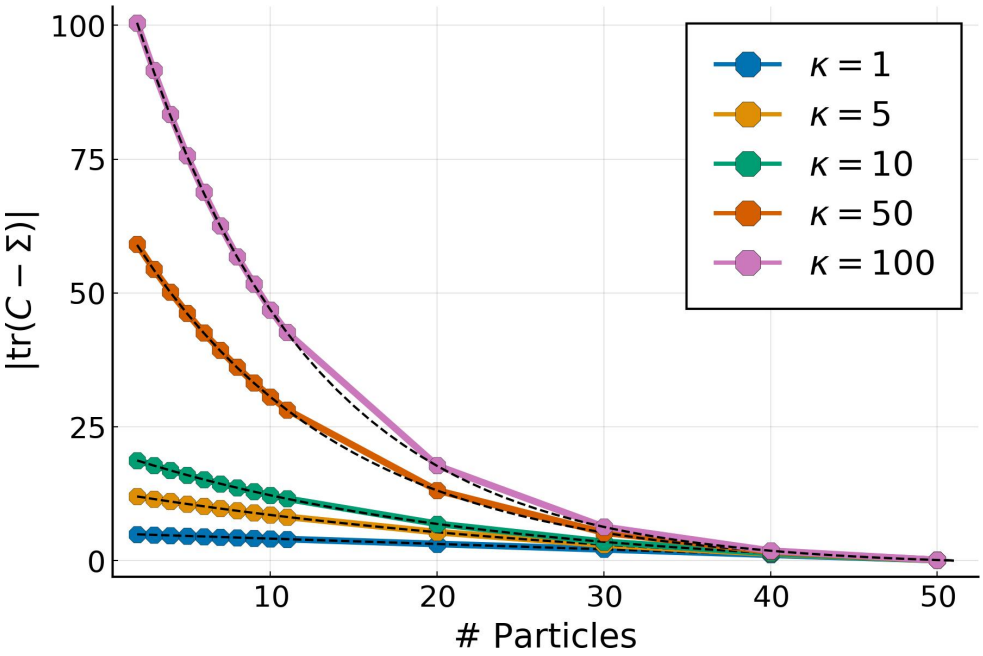


Figure 3. Trace error for a Gaussian target with $D = 50$ and condition numbers κ for a varying number of particles with GPF. Predictions from Theorem 3 are shown in dashed-black.

4.3. High-Dimensional Low-Rank Gaussian Targets

We consider a typical low-rank target case where the dimensionality is high but the effective rank of the covariance is unknown. The target is given by $p(x) = \mathcal{N}(\mu, \Sigma)$ where $\mu \sim \mathcal{N}(0, \mathbb{I}_D)$, the covariance is defined by $\Sigma = U\Lambda U^\top$, where U is a $D \times D$ unitary matrix and Λ is a diagonal matrix defined by

$$\Lambda_{ii} = \begin{cases} \mathcal{N}(2,1), & \text{if } i \leq K \\ 10^{-8}, & \text{otherwise} \end{cases}$$

where K is the effective rank of the target. We pick $D = 500$ and vary $K \in \{10, 20, 30\}$ to simulate a true problem where the correct K is not known. We test all methods allowing

for low-rank structure, namely, GPF, GF, FCS and SVGD (Linear and Sq. Exp.). We fix the rank (or the number of particles) to be 20; therefore, we obtain three cases where the rank is exact, under-estimated, and over-estimated. For all methods, we use RMSProp [37] for the stochastic methods, or a diagonal version of it (see Section 4.4) for the particle ones. The error of the mean and the covariance is shown in Figure 4. Note that the difference in the initial error on the covariance is due to the difficulty of starting with the same covariance between particle and stochastic methods.

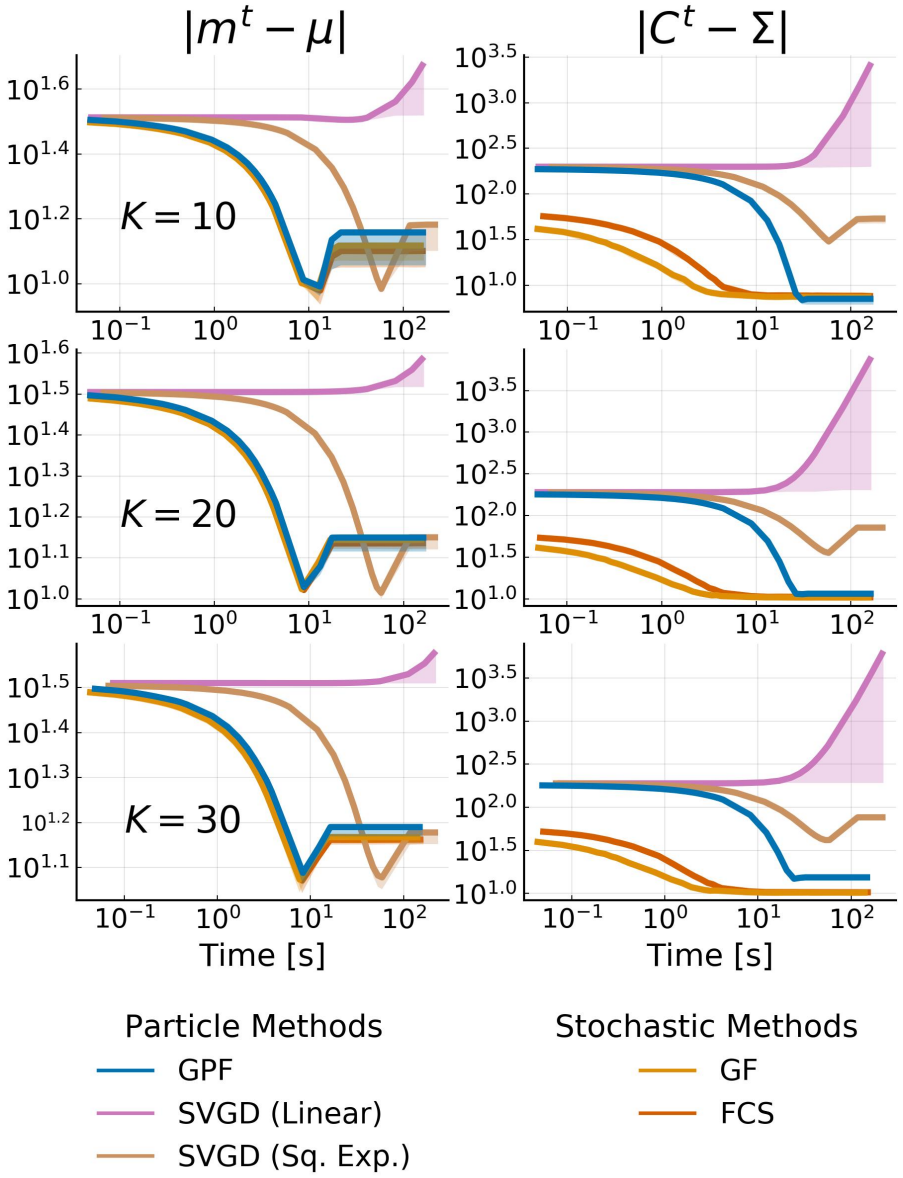


Figure 4. Convergence plot of low-rank methods for a 500-dimensional multivariate Gaussian target with effective rank $K \in \{10, 20, 30\}$. The rank of each method is fixed as 20. The difference in the starting point for the covariance is due to the initialization difference between each method. We show the mean over 10 runs for each method with shadowed areas representing the 68% credible interval.

We observe once again that the SVGD with a linear kernel fails to converge due to the large gradients. All methods perform equally in the estimation of the mean while being non-influenced by the rank of the target. As expected, the approximation quality for the covariance degrades when the rank gets bigger, but all algorithms still converge to good

approximations. SVGD with a sq. exp. kernel performs much worse than the rest of the methods. This is a known phenomenon where, for high dimensions, the covariance SVGD is either over- or underestimated.

4.4. Non-Gaussian Target

We now investigate the behavior of our algorithm with non-Gaussian target distributions. We built a two-dimensional banana distribution: $p(x) \propto \exp(-0.5(0.01x_1^2 + 0.1(x_2 + 0.1x_1^2 - 10)^2))$, varied the number of particles used for GPF in $\{3, 5, 10, 20, 50\}$ and compared it with a standard full-rank VGA approach. We also showed the impact of replacing a fixed η with the Adam [39] optimizer for 50 particles. The results are shown in Figure 5. As expected, increasing the number of particles madesthe distribution obtained via GPF increasingly closer to the optimal standard VGA, even in a non-Gaussian setting. However, using a momentum-based optimizer such as Adam breaks the linearity assumption of the original flow (5) and leads to a twisted representation of the particles. (We observed the same behavior with other momentum-based optimizers). A simple modification of the most known optimizers allows the linearity to be maintained while correctly adapting the learning rate to the shape of the problem. Most optimisers accumulate momentum or gradients element-wise, and end up modifying the updates as $x^{t+1} = x^t + P^t \odot \varphi^t(x^t)$, where $P^t \in \mathbb{R}^{D\times D}$ is the preconditioner obtained via the optimiser and $\odot$ is the Hadamard product. By instead taking the average over each dimensions, we obtained the updates $x^{t+1} = x^t + P^t\varphi^t(x^t)$, where P^t is a $D \times D$ diagonal matrix. The details of the dimension-wise conditioners for ADAM, AdaGrad and AdaDelta are given in Appendix H.

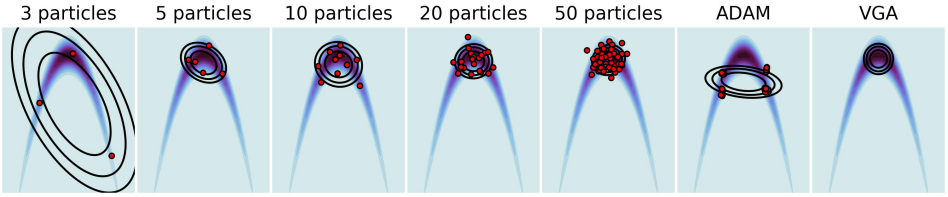
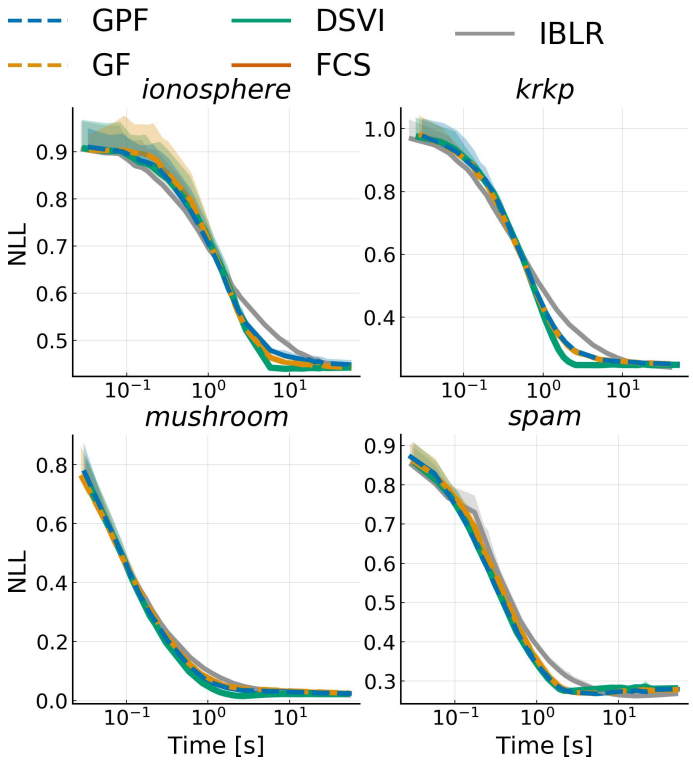


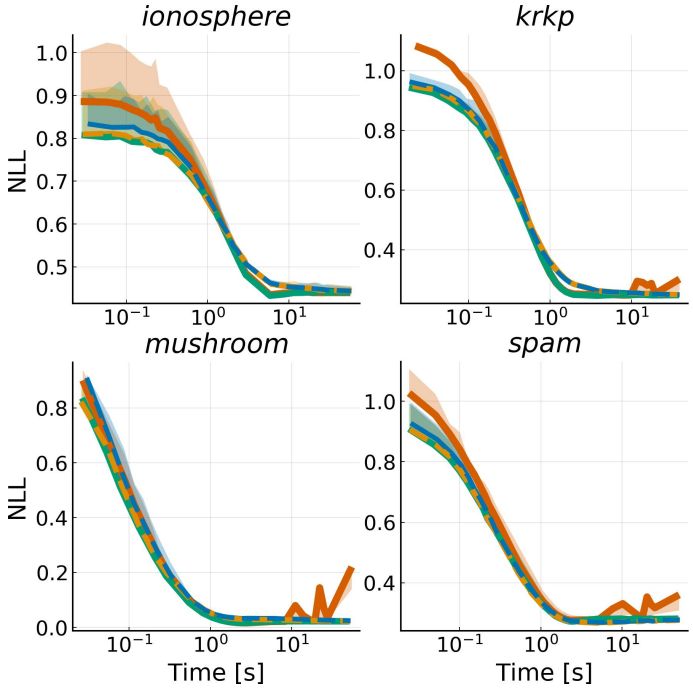
Figure 5. Two-dimensional Banana distribution. Comparison of GPF using an increasing number of particles and a different optimizer (ADAM) with the standard VGA (rightmost plot).

4.5. Bayesian Logistic Regression

Finally, we considered a range of real-world binary classification problems modeled with a Bayesian logistic regression. Given some data $\{(x_i, y_i)\}_{i=1}^N$ where $x_i \in \mathbb{R}^D$ and $y \in \{-1, 1\}$, we defined the model $y_i \sim \text{Bernoulli}(\sigma(w^\top x_i))$ with weight $w \in \mathbb{R}^D$, and with σ being the logistic function. We set a prior on w: $w\ \mathcal{N}(0, 10\mathbb{I}_D)$. We benchmarked the competing approaches over four datasets from the UCI repository [40]: `spam` ($N = 4601, D = 104$), `krkp` ($N = 351, D = 111$), `ionosphere` ($N = 3196, D = 37$) and `mushroom` ($N = 8124, D = 95$). We ran all algorithms discussed in Section 4.1, both with and without a mean-field approximation; SVGD was omitted since it is too unstable. All algorithms were run with a fixed learning rate $\eta = 10^{-4}$, and we used mini-batches of size 100. We show alternative training settings in Appendix I. Note that FCS, for mean-field, simplifies to DSVI Additionally, we did not consider full-rank IBLR, as it is too expensive, and we used their reparametrized gradient version for the Hessian. Figure 6 shows the average negative log-likelihood on 10-fold cross-validation with one standard deviation for each dataset. While, as expected, the advantages shown for Gaussian targets do not transfer to non-Gaussian targets, GPF and GF are consistently on par with competitors. On the other hand, IBLR tends to be outperformed. It is also interesting to note that mean-field does not seem to have a negative impact on these problems, and performance remains the same even with a full-rank matrix.



(**a**) Mean-field approximation



(**b**) No mean-field approximation

Figure 6. Average negative log-likelihood vs. time on a test-set over 10 runs against training time for a Bayesian logistic regression model applied to different datasets. Top plots use a mean-field approximation, while bottom plots use a low-rank structure for the covariance with rank $L = 100$.

4.6. Bayesian Neural Network

We ran our algorithm on a standard network with two hidden layers each, with $L = 200$ neurons and tanh activation functions (we additionally tried ReLU [41], but some baselines failed to converge). We trained on the MNIST dataset [42] ($N = 60{,}000$, $D = 784$) and used an isotropic prior on the weights $p(w) = \mathcal{N}(0, \alpha I_D)$ with $\alpha = 1.0$. We additionally compared these with *Stochastic Weight Averaging-Gaussian* **(SWAG)** [27] with an SGD learning rate of 10^{-6} (selected empirically) and *Efficient Low-Rank Gaussian Variational Inference* **(ELRGVI)** [26]. We varied the assumptions on the covariance matrix to be diagonal (**Mean-Field**), or to have rank $L \in \{5, 10\}$. Additionally, we showed, for GPF, the effect of using a structured mean-field assumption by imposing the independence of the weights between each layer (**GPF (Layers)**).

We trained each algorithm for 5000 iterations with a batchsize of 128($\sim$10 epochs) and reported the final average negative log-likelihood, accuracy and expected calibration error [43] on the test set ($N = 10{,}000$) on Table 1. The predictive distribution is given by

$$p(y = k|x^*, \mathcal{D}) = \int p(y = k|x^*, w)p(w|\mathcal{D})dw \approx \int p(y = k|x^*, w)q(w)dw,$$

where $\mathcal{D}$ is the training data, and x^* is a test sample. We computed the accuracy and the average negative test log-likelihood as:

$$\text{Acc} = \frac{1}{N}\sum_{i=1}^{N} 1_{y_i}(\arg_k \max p(y = k|x_i^*, \mathcal{D}))$$

$$\text{NLL} = -\frac{1}{N}\sum_{i=1}^{N} \log p(y = y_i|x_i^*, \mathcal{D})$$

where $1_y(x)$ is the indicator function (equal to 1 for $y = x$, 0 otherwise). For the definition of expected calibrated error, we refer the reader to [43]. Additional convergence and uncertainty calibration plots can be found in Appendix I.

Table 1. Negative Log-Likelihood (NLL), Accuracy (Acc), and Expected Calibration Error (ECE) for a *Bayesian Neural Networks* **(BNN)** on the MNIST dataset. We varied the rank of the variational covariance from mean-field (all variables are independent) to a low-rank structure with $L \in \{5, 10\}$. Bold numbers indicated the best performance, and italic bold numbers indicate the best performance when restricted to VGA methods. Convergence and calibration plots can be found in Appendix I.

Alg.	Mean-Field			$L = 5$			$L = 10$		
	NLL	Acc	ECE	NLL	Acc	ECE	NLL	Acc	ECE
GPF	0.183	0.95	0.0384	0.166	***0.96***	0.0918	0.172	0.955	0.0869
GPF (Layers)	-	-	-	***0.147***	0.958	**0.0181**	0.178	0.952	0.0395
GF	0.178	0.953	0.0706	0.185	0.956	0.136	0.171	0.952	0.0455
DSVI	0.204	0.945	0.11	-	-	-	-	-	-
SVGD (Sq. Exp)	-	-	-	0.139	0.965	0.0732	**0.133**	**0.967**	0.0879
SWAG	-	-	-	0.257	0.957	0.0662	0.287	0.956	0.0878
ELRGVI	-	-	-	0.453	0.901	0.53	0.537	0.882	0.777

Overall, the SVGD method performed best in terms of both accuracy and negative log-likelihood. However, SVGD is not in the same category as others, since it is not a VGA. For VGAs, we observed that a low-rank approximation improves upon mean-field methods. In particular, assuming independence between layers provides a large advantage to GPF. GPF and GF generally perform equally or better than all the other VGA methods. Note that, although not reported here, all methods needed approximately the same time for the 5000 iterations, except for SWAG, which only needed the MAP and a few thousand iterations of SGD afterward, making it generally faster but also less controlled (a grid search was needed to find the appropriate learning for SGD).

5. Discussion

We introduced GPF, a general-purpose and theoretically grounded, particle-based approach, to perform inference with variational Gaussians as well as GF its parameter version. We were able to show the convergence of the particle algorithm based on an empirical approximation of the free energy. We also showed that we can approximate high-dimensional targets by allowing for low-rank approximations with a small number of particles. The results for Gaussian targets suggest that the convergence of posterior covariance approximation may relax asymptotically fast, with small dependence on the target. This work is the first step in analyzing convergence speed and guarantees in inference with variational Gaussians, and future work could extend guarantees to non-Gaussian problems. One could also take advantage of existing particle-based VI methods to accelerate inference further or reach a better optima [44,45].

Author Contributions: Conceptualization, T.G.-F. and M.O.; methodology, T.G.-F., V.P. and M.O.; software, T.G.-F.; validation, T.G.-F.; formal analysis, T.G.-F.; investigation, T.G.-F.; resources, T.G.-F. and V.P.; data curation, T.G.-F.; writing—original draft preparation, T.G.-F., V.P. and M.O.; writing—review and editing, T.G.-F., V.P. and M.O.; visualization, T.G.-F.; supervision, M.O.; project administration, T.G.-F.; funding acquisition, M.O. All authors have read and agreed to the published version of the manuscript.

Funding: We acknowledge the support of the German Research Foundation and the Open Access Publication Fund of TU Berlin.

Data Availability Statement: Datasets can be found on the UCI dataset website [40] and the MNIST dataset can be found on Yann Lecun website [42].

Acknowledgments: We thank Fela Winkelmolen for his initial help on computations, Jannik Thümmel for his work on the linear SVGD and the reviewers for their insightful comments.

Conflicts of Interest: The authors declare no conflict of interest.

Appendix A. Derivation of the Optimal Parameters

In Section 3, we considered the optimization problem:

$$\min_{A^t, b^t \in \mathcal{B}} \frac{d\mathcal{F}[q^t]}{dt} \text{ where } \mathcal{B} = \{A^t, b^t : \|A^t\|_F^2 = 1,\ \|b^t\|^2 = 1\},$$

where we have introduced $\|A^2\|_F^2 = \mathrm{tr}(AA^\top)$, the Froebius norm and $\|b^t\|$, the L_2 norm and

$$\frac{d\mathcal{F}[q^t]}{dt} = -\mathrm{tr}\Big[A^t (A_\star^t)^\top\Big] - (b^t)^\top b_\star^t \tag{A1}$$

To solve this problem, we used the Lagrange multiplier method. We write the Lagrangian as:

$$\mathcal{L}(A^t, b^t) = \frac{d\mathcal{F}[q^t]}{dt} - \lambda_A g(A^t) - \lambda_b h(b^t),$$

where $g(A) = \mathrm{tr}(AA^\top) - 1$ and $h(b) = \|b\|_2^2 - 1$. For simplicity we can divide the problem as:

$$\begin{aligned}\mathcal{L}(A^t) &= -\mathrm{tr}\Big[A^t (A_\star^t)^\top\Big] - \lambda_A g(A^t)\\ \mathcal{L}(b^t) &= -(b^t)^\top b_\star^t - \lambda_b h(b^t)\end{aligned}$$

For A^t, we have the constraints:

$$\begin{aligned}\nabla_{A^t}\mathrm{tr}\left[A^t(A^t_\star)^\top\right] &= \lambda_A \nabla_{A^t} g(A^t)\\ g(A^t) &= 0\end{aligned}$$

Computing the gradients is straightforward:

$$\begin{aligned}A^t_\star &= 2\lambda_A A^t\\ \Rightarrow A^t &= \frac{A^t_\star}{2\lambda_A}\\ \Rightarrow \frac{1}{4\lambda_A^2}\mathrm{tr}(A^t_\star(A^t_\star)^\top) &= 1\\ \Rightarrow \lambda_A &= \sqrt{\frac{\mathrm{tr}(A^t_\star(A^t_\star)^\top)}{4}}.\end{aligned}$$

which gives us the result $A^t = \frac{A^t_\star}{\|A^t_\star\|_F}$. Similarly for b^t:

$$\begin{aligned}\nabla_{b^t}(b^t)^\top b^t_\star &= \lambda_b \nabla_{b^t} h(b^t)\\ h(b^t) &= 0.\end{aligned}$$

Replacing the gradients gives:

$$\begin{aligned}b^t_\star &= 2\lambda_b b^t\\ \Rightarrow b^t &= \frac{b^t_\star}{2\lambda_b}\\ \Rightarrow \frac{1}{4\lambda_b^2}\|b^t_\star\|_2^2 &= 1\\ \Rightarrow \lambda_b &= \frac{2}{\|b^t_\star\|_2}\end{aligned}$$

which gives us the result $b^t = \frac{b^t_\star}{\|b^t_\star\|_2}$.

Appendix B. Relaxation of the Empirical Free Energy

We prove the decrease in the empirical free energy (17) under the particle flow when the covariance C is nonsingular. We define the empirical distribution $\hat{q}(x) = \frac{1}{N}\sum_{i=1}^N \delta_{x,x_i}$ with a finite number N of particles. The empirical free energy is defined as

$$\mathcal{F}[\hat{q}] = \mathbb{E}_{\hat{q}}[\varphi(x)] - \frac{1}{2}\log|C|.$$

We are interested in the temporal change of the free energy, when particles move under a general linear dynamics

$$\frac{dx_i}{dt} = b + A(x_i - m).$$

The induced dynamics for $\mathcal{F}$ are:

$$\frac{d\mathcal{F}}{dt} = \mathbb{E}_{q^t}\left[\nabla_x\varphi(x)^\top\frac{dx}{dt}\right] - \frac{1}{2}\mathrm{tr}(C^{-1}\frac{dC}{dt})$$

For notational simplicity, we introduce $g(x) = \nabla_x\varphi(x)$ and $\dot{x} = \frac{dx}{dt}$ (similarly $\dot{m} = \frac{dm}{dt}$).

$$\begin{aligned}\frac{dC}{dt} =&\frac{d}{dt}\mathbb{E}_q\Big[(x-m)(x-m)^\top\Big]\\ =&\mathbb{E}_q\Big[(\dot{x}-\dot{m})(x-m)^\top\Big]+\mathbb{E}_q\Big[(x-m)(\dot{x}-\dot{m})^\top\Big]\\ =&\mathbb{E}_q\Big[\dot{x}x^\top+x\dot{x}^\top-\dot{m}m^\top-m\dot{m}^\top\Big]\\ =&\mathbb{E}_q\Big[\dot{x}(x-m)^\top\Big]+\mathbb{E}_q\Big[(x-m)\dot{x}^\top\Big]\end{aligned}$$

$$\begin{aligned}\frac{d\mathcal{F}}{dt} =&\mathbb{E}_q\Big[g(x)^\top\dot{x}\Big]-\\ &\frac{1}{2}\mathbb{E}_q\Big[\mathrm{tr}(C^{-1}\dot{x}(x-m)^\top)+\mathrm{tr}(C^{-1}(x-m)^\top\dot{x}^\top)\Big]\\ =&\mathbb{E}_q\Big[\dot{x}^\top\Big(g(x)-C^{-1}(x-m)\Big)\Big]\end{aligned}\tag{A2}$$

where we used the permutation properties of the trace.

Plugging the dynamics into Equation (A2), we obtain:

$$\begin{aligned}\frac{d\mathcal{F}}{dt} =&b^\top\mathbb{E}_q[g(x)]+\mathbb{E}_q\Big[(x-m)^\top A^\top g(x)\Big]\\ &-\mathbb{E}_q\Big[(x-m)^\top A^\top C^{-1}(x-m)\Big]\end{aligned}\tag{A3}$$

where we used the fact that $b^\top C^{-1}\mathbb{E}_q[x-m]=0$.

We next look for conditions on b and A, under which $\frac{d\mathcal{F}}{dt}<0$, i.e., the dynamics will lead to a decrease in the free energy. We pick $b=-\beta_1\mathbb{E}_q[g(x)]$, where $\beta_1>0$, and we obtain, for the first term in (A3):

$$-\beta_1\|\mathbb{E}_q[g(x)]\|^2\leq 0.$$

For A, let us first define $\psi=\mathbb{E}_q\big[g(x)(x-m)^\top\big]$ and rewrite the second and last term of the Equation (A3) as:

$$\begin{aligned}\mathbb{E}_q\Big[(x-m)^\top A^\top g(x)\Big] =&\mathrm{tr}\Big(\mathbb{E}_q\Big[A^\top g(x)(x-m)^\top\Big]\Big)\\ =&\mathrm{tr}\Big(A^\top\psi\Big)\\ \mathbb{E}_q\Big[(x-m)^\top A^\top C^{-1}(x-m)\Big] =&\mathrm{tr}\Big(A^\top C^{-1}C\Big)\\ =&\mathrm{tr}(A)\end{aligned}$$

Combining both, we get $\mathrm{tr}\big(A^\top(\psi-I)\big)$. Similarly to the previous step, we pick $A=-\beta_2(\psi-I)$, where $\beta_2\geq 0$, which leads to another negative term:

$$-\beta_2\mathrm{tr}((\psi-I)^\top(\psi-I))\leq 0,$$

where we use the fact that $X^\top X$ is a positive semi-definite matrix for any real valued X.

Note that different forms of A (e.g., β_2 are replaced by a positive definite matrix) could be used, as long as the trace of the product stays positive. Inserting b and A, the free energy dynamics become

$$\frac{d\mathcal{F}}{dt}=-\beta_1\|\mathbb{E}_q[g(x)]\|^2-\beta_2\mathrm{tr}((\psi-I)^\top(\psi-I))$$

The variable dynamics are given by

$$\begin{aligned}\frac{dx}{dt} &= -\beta_1 \mathbb{E}_q[g(x)] - \beta_2(\psi - I)(x - m) \\ &= -\beta_1 \mathbb{E}_q[g(x)] \\ &\quad - \beta_2\Big(\mathbb{E}_q\Big[g(x)(x-m)^\top\Big] - I\Big)(x-m),\end{aligned}$$

which is equivalent to Equation (5), for $\beta_1 = \beta_2 = 1$. Our result shows that the empirical approximation of the free energy decreases under the particle flow.

Appendix C. Riemannian Gradient for Matrix Parameter Γ

The parameter flow for the matrix Γ in (11) is given by

$$\frac{d\Gamma^t}{dt} = \Gamma^t - \mathbb{E}_{q^0}\Big[\nabla_x \varphi(x^t)(x^0 - m^0)^\top\Big]\Gamma^t(\Gamma^t)^\top.$$

This is easily rewritten in terms of the parameter gradient as $\frac{d\Gamma^t}{dt} = \frac{\partial \mathcal{F}}{\partial \Gamma}\Gamma\Gamma^\top$

Similar to natural gradients, which are defined by the metric, which is induced by the Fisher–matrix, we can rewrite the parameter change in terms of a different *Riemannian* gradient. This gradient is the direction of change $d\Gamma = \Gamma(t + dt) - \Gamma(t)$, which yields the steepest descent of the free energy over a small time interval dt. As an extra condition, one keeps the length of $d\Gamma$ (measured by a 'natural' metric, which has specific invariance properties) fixed. This is defined by an inner product (the squared length) $\langle d\Gamma, d\Gamma\rangle_\Gamma$ in the tangent space of small deviations $d\Gamma$ from the matrix Γ. Hence, $d\Gamma$ is found by minimising $\mathcal{F}(\Gamma(t) + d\Gamma, m)$ (for small $d\Gamma$) under the condition that $\langle d\Gamma, d\Gamma\rangle_{\Gamma(t)}$ is fixed. Following [20] (Theorem 6), a natural metric in the space of symmetric nonsingular matrices can be defined as

$$\langle d\Gamma, d\Gamma\rangle_\Gamma \doteq \mathrm{tr}\Big((d\Gamma\ \Gamma^{-1})^\top d\Gamma\ \Gamma^{-1}\Big).$$

This metric is invariant against multiplications of Γ and $d\Gamma$ by matrices Y, i.e., $\langle d\Gamma, d\Gamma\rangle_\Gamma = \langle d\Gamma\ Y, d\Gamma\ Y\rangle_{\Gamma Y}$ and reduces to the Euclidian metric at the unit matrix $\Gamma = I$.

The direction of the natural gradient is obtained by expanding the free energy for small $d\Gamma$ and introducing a Lagrange–multiplier λ for the constraint. One ends up with the quadratic form

$$\frac{\partial \mathcal{F}}{\partial \Gamma} d\Gamma + \lambda \mathrm{tr}\Big((d\Gamma\ \Gamma^{-1})^\top d\Gamma\ \Gamma^{-1}\Big)$$

to be minimised by $d\Gamma$. By taking the derivative with respect to $d\Gamma$, one finds that the direction of $d\Gamma$ agrees with the right equation of the flow (11).

Appendix D. Regularised Free Energy for $N \leq D$

The problem of defining an empirical approximation for $N \leq D$ particles is that the empirical covariance becomes singular and typically has $N - 1$ nonzero eigenvalues, and thus $|C| = 0$. Note that the extra 0 eigenvalue is derived from the fact that the empirical sum of fluctuations must be zero, which provides an additional linear constraint.

We can regularise the log determinant term by replacing the zero eigenvalues of C: $\lambda_i = 0 \to \tilde{\lambda}_i = 1$. The new covariance $\tilde{C}$ becomes

$$\log|\widetilde{C}| = \sum_{i:\lambda_i > 0} \log \lambda_i,$$

since $\log 1 = 0$. The dynamics of the particles stays the same. To rewrite this formally in terms of matrices, we define

$$\widetilde{C} = C + C_\perp$$

where

$$C_\perp = \sum_{i:\lambda_i=0} e_i e_i^\top$$

and $e_i = i$th eigenvector of C. This replaces all 0 eigenvalues by 1. $C_\perp$ is a projector: $C_\perp^2 = C_\perp$ and $C_\perp(I - C_\perp) = 0$. We also have $\mathrm{tr}(C_\perp) = D - (N-1)$. In the following, it is useful to introduce the $D \times N$ matrix of fluctuations Z, such that $C = ZZ^\top/N$. The column vectors of Z span the subspace of eigenvectors e_i with $\lambda_i > 0$. Hence, it follows that $C_\perp Z = 0$.

We want to show that the regularised free energy $\widetilde{F}$ decreases under the particle dynamics for $N \leq D$. Since the part of the time derivative of $\widetilde{F}$ that depends on $\frac{dm}{dt}$ is not changed, we will only discuss the fluctuation part in the following.

It is useful to introduce the matrix:

$$\widetilde{A} \doteq I - C_\perp - gZ^\top/N = A - C_\perp,$$

with $g = \nabla_x \varphi(x)$ is the $D \times N$ matrix of the gradient.

$$\begin{aligned}\mathbb{E}_q\left[g(x)^\top \frac{dx}{dt}\right] =& \mathrm{tr}(A) - \mathrm{tr}(A^\top A)\\ =& \mathrm{tr}(\widetilde{A} + C_\perp) - \mathrm{tr}((\widetilde{A} + C_\perp)^\top(\widetilde{A} + C_\perp))\\ =& \mathrm{tr}(\widetilde{A}) - \mathrm{tr}(\widetilde{A}^\top \widetilde{A}).\end{aligned}$$

To obtain this result, we need

$$\begin{aligned}\mathrm{tr}(C_\perp \widetilde{A}) =& \mathrm{tr}(C_\perp \widetilde{A}^\top)\\ =& \mathrm{tr}(C_\perp(I - C_\perp) - C_\perp Z g^\top/N) = 0.\end{aligned}$$

We need to work out

$$\begin{aligned}-\frac{1}{2}\frac{d\ln|\widetilde{C}|}{dt} =& -\frac{1}{2}\mathrm{tr}\left(\frac{d\widetilde{C}}{dt}\widetilde{C}^{-1}\right)\\ =& -\frac{1}{2}\mathrm{tr}\left(\frac{dC}{dt}\widetilde{C}^{-1}\right)\end{aligned}$$

where we have used the fact that the eigenvalues $\lambda_i = 1$ of $\widetilde{C}$ have a zero time derivative and can be omitted. We use the linear dynamics $\frac{dZ}{dt} = AZ$ to obtain:

$$\begin{aligned}\frac{dC}{dt} =& = CA^\top + AC\\ =& (\widetilde{C} - C_\perp)(\widetilde{A}^\top + C_\perp) + (\widetilde{A} + C_\perp)(\widetilde{C} - C_\perp)\\ =& \widetilde{C}\widetilde{A}^\top + \widetilde{A}\widetilde{C} + C_\perp\widetilde{C} + \widetilde{C}C_\perp - \widetilde{A}C_\perp - C_\perp\widetilde{A}^\top - 2C_\perp\\ =& \widetilde{C}\widetilde{A}^\top + \widetilde{A}\widetilde{C},\end{aligned}$$

where we have used $C_\perp^2 = C_\perp$ and $C_\perp\widetilde{A}^\top = 0$. Hence

$$-\frac{1}{2}\mathrm{tr}\left(\frac{d\widetilde{C}}{dt}\widetilde{C}^{-1}\right) = -\mathrm{tr}(\widetilde{A}).$$

Finally, the temporal change in the free energy due to the fluctuations is given by

$$\frac{d\widetilde{\mathcal{F}}}{dt} = -\mathrm{tr}(\widetilde{A}^\top \widetilde{A}) \leq 0.$$

Note that this proof is not only valid for $N \leq D$, but also for $N > D$, as the overall computations are simplified with $C_\perp = 0$. A more detailed proof for $N > D$ is, furthermore, given in Appendix B.

Efficient Computation of $\log\left|\widetilde{C}\right|$

A practical way to compute $\log|\widetilde{C}|$ without performing an eigenvector expansion is to define the $N \times N$ matrix

$$R \doteq Z^\top Z/N + J_{N,N}/N,$$

where $J_{N,N}$ is the $N \times N$ *all-ones* matrix. $Z^\top Z/N$ shares the $N-1$ nonzero eigenvalues with C and has an additional eigenvalue 0 corresponding to the constant eigenvector $(e_N)_i = 1/\sqrt{N}$. Adding an all-ones matrix preserves all existing eigenvalues while replacing the 0 one with a constant. This leads to the following result:

$$-\frac{1}{2}\log|R| = -\frac{1}{2}\sum_{i=1}^{N-1}\log\lambda_i.$$

Appendix E. Proof of Theorem 1: Fixed Points for a Gaussian Model ($N > d$)

Theorem A1 (1)**.** *If the target density $p(x)$ is a D-dimensional multivariate Gaussian, only $D+1$ particles are needed for Algorithm 2 to converge to the exact target parameters.*

The general fixed-point condition for the dynamics (13) of the position x_i for particle i is given by:

$$(I - \mathbb{E}_{\hat{q}}\left[g(x)(x-m)^\top\right])(x_i - m) - \mathbb{E}_{\hat{q}}[g(x)] = 0.$$

for $i = 1, \ldots, N$. By taking the expectation over all particles, we obtain:

$$\mathbb{E}_{\hat{q}}[g(x)] = 0, \tag{A4}$$

where $\hat{q}$ is the empirical distributions of particles at the the fixed point. Note that this result is independent of N, i.e., it is also valid for $N = 1$.

For a D-dimensional Gaussian target $p(x) = \mathcal{N}(\mu, \Sigma)$, we will show that empirical mean and covariance given by the particle algorithm converge to the true mean and covariance matrix of the Gaussian when we use $N \geq D+1$ particles. In this setting, we have $\varphi(x) = \frac{1}{2}x^\top\Sigma^{-1}x - x^\top\Sigma^{-1}\mu$. For simplification, we use the precision matrix $\Lambda = \Sigma^{-1}$ and get

$$\varphi(x) = \frac{1}{2}x^\top\Lambda x - x^\top\Lambda\mu.$$

The gradient $g(x)$ becomes:

$$g(x) = \Lambda(x - \mu)$$

At the fixed points, we have that $\frac{dm}{dt}$ and $\frac{d\Gamma}{dt}$ are equal to 0. For the mean m:

$$\begin{aligned}\frac{dm}{dt} = \mathbb{E}_{\hat{q}}[g(x)] &= 0\\ \Lambda\mathbb{E}_{\hat{q}}[x-\mu] &= 0\\ \Lambda m &= \Lambda\mu\\ m &= \mu\end{aligned}$$

For the matrix Γ, we have

$$\begin{aligned}\frac{d\Gamma}{dt} = -A\Gamma &= 0\\ \Gamma - \mathbb{E}_{q_0}\left[g(x)(x-m)^\top\right]\Gamma &= 0\\ \mathbb{E}_{q_0}\left[\Lambda(x-\mu)(x-m)^\top\right]\Gamma &= \Gamma\\ -2\eta_2\mathbb{E}_{q_0}\left[(x-m)(x-m)^\top\right]\Gamma &= \Gamma\\ \Lambda C\Gamma &= \Gamma\\ \Lambda C^2 &= C\end{aligned}$$

where we use the result for the mean $m = \mu$ and right multiplied by $\Gamma^\top$ as $C = \Gamma\Gamma^\top$. Now, we can only simplify, as $C = \Lambda^{-1} = \Sigma$ if C is not singular. This is true only if its rank is equal to D, needing $D+1$ particles.

Appendix F. Proof of Theorem 2: Rates of Convergence for Gaussian Targets

Theorem A2 (2). *For a target $p(x) = \mathcal{N}(x \mid \mu, \Lambda^{-1})$, where $x \in \mathbb{R}^D$, and $N \geq D+1$ particles, the continuous time limit of Algorithm 2 will converge exponentially fast for both the mean and the trace of the precision matrix:*

$$\begin{aligned}m^t - \mu &= e^{-\Lambda t}(m^0 - \mu),\\ \mathrm{tr}(\left(C^t\right)^{-1} - \Lambda) &= e^{-2t}\mathrm{tr}(\left(C^0\right)^{-1} - \Lambda),\end{aligned}$$

where m^t and C^t are the empirical mean and covariance matrix at time t and $\exp(-\Lambda t)$ is the matrix exponential.

In the following, we assume the target $p(x) = \mathcal{N}(\mu, \Sigma)$ We use the notation $\Lambda \doteq \Sigma^{-1}$ and $\delta C^t = C^t - \Sigma$.

Appendix F.1. Convergence of the Mean

Given our target $p(x)$, similarly to Appendix E we have $g(x) = \Lambda(x-\mu)$, where $\eta_1 = \Sigma^{-1}\mu$ and $\eta_2 = -\frac{1}{2}\Sigma^{-1}$. This transform the first of Equations (11) into

$$\begin{aligned}\frac{dm}{dt} &= -\Lambda(\mathbb{E}_{\hat{q}}[x] - \mu)\\ &= -\Lambda(m-\mu)\end{aligned}$$

If now consider the error on m : $\delta m = m - \mu$ we obtain:

$$\begin{aligned}\frac{d\delta m}{dt} = \frac{dm}{dt} &= -\Lambda(m-\mu)\\ &= -\Lambda\delta m.\end{aligned}$$

Therefore, the mean converges exponentially fast to the true mean. The asymptotic rate is governed by the largest eigenvalue of Λ, i.e., the inverse of the smallest eigenvalue of Σ, $\lambda_{\min}$.

Appendix F.2. Convergence of the Covariance Matrix

Let $z = x - m$, we have from Equation (5), that

$$\frac{dz}{dt} = -Az$$

where $A = \mathbb{E}_{q_0}\left[g(x)z^\top\right] - I$. This expectation can further be simplified as

$$\mathbb{E}_{\hat{q}}\left[\Lambda(x-\mu)z^\top\right] = \Lambda C, \tag{A5}$$

where $q \sim \mathcal{N}(m, C)$. Hence, we have the exact result

$$\frac{dC}{dt} = (I - \Lambda C)C + C(I - C\Lambda). \tag{A6}$$

We know that the optimal target is $C = \Sigma$. Therefore, we define the error $\delta C = C - \Sigma$. Linearizing Equation (A6) gives us

$$\begin{aligned}\frac{d\delta C}{dt} = \frac{dC}{dt} =& (I - \Lambda(\delta C + \Sigma))(\delta C + \Sigma) \\ &+ (\delta C + \Sigma)(I - (\delta C + \Sigma)\Lambda) \\ =& -\Lambda\delta C(\delta C + \Sigma) - (\delta C + \Sigma)\delta C\Lambda \\ \approx& -\Lambda\delta C\Sigma - \Sigma\delta C\Lambda\end{aligned}$$

We were not yet able to find a general solution of this equation, but we can obtain a simple result for the trace $y^t \doteq \mathrm{tr}(\delta C)$ at time t:

$$\frac{dy^t}{dt} \simeq -2y^t.$$

We, therefore, have a asymptotic linear convergence: $y^t \propto e^{-2t}y^0$ which is independent of the parameters of the Gaussian model.

We can also equivalently obtain a non-asymptotic estimate of a specific error measure for the precision matrix. Using equation (A6), we have the following dynamics for the precision C^{-1}:

$$\begin{aligned}\frac{dC^{-1}}{dt} =& -C^{-1}\frac{dC}{dt}C^{-1} \\ =& -C^{-1}(I - \Lambda C) - (I - \Lambda C)C^{-1}\end{aligned}$$

Taking the trace

$$\begin{aligned}\frac{d\mathrm{tr}(C^{-1})}{dt} =& -2\mathrm{tr}(C^{-1}) - 2\mathrm{tr}(\Lambda) \\ \frac{d\mathrm{tr}(C^{-1} - \Lambda)}{dt} =& -2\mathrm{tr}(C^{-1} - \Lambda)\end{aligned}$$

Hence we get the following exact result:

$$\mathrm{tr}((C^t)^{-1} - \Lambda) = e^{-2t}\mathrm{tr}((C^0)^{-1} - \Lambda)$$

which is again independent of the parameters of the Gaussian model.

Additionally, this tells us that if the covariance C is non-singular at time $t = 0$, it will remain non-singular for all t ($\mathrm{tr}(C^{-1})$ would be infinite). Hence, if we start with $N > d$ particles with a proper empirical covariance, they cannot collapse to make C singular.

Appendix F.3. Convergence of the Trace of the Covariance

The asymptotic result on traces obtained previously can be turned into an exact inequality. We have

$$\frac{d\delta C}{dt} = -\Lambda\delta C\Sigma - \Sigma\Lambda\delta C - \Lambda(\delta C)^2 - (\delta C)^2\Lambda$$

Taking the trace, we get

$$\frac{d\mathrm{tr}(\delta C)}{dt} = -2\mathrm{tr}(\delta C) - 2\mathrm{tr}(\delta C\Lambda\delta C)$$

Since $\delta C\Lambda\delta C$ is positive definite, we have $-2\mathrm{tr}(\delta C\Lambda\delta C) \leq 0$ and thus

$$\frac{d\mathrm{tr}(\delta C)}{dt} \leq -2\mathrm{tr}(\delta C)$$

leading to:

$$\mathrm{tr}(\delta C^t) \leq \mathrm{tr}(\delta C^0)e^{-2t}$$

by using by Grönwall's lemma [46]:

Lemma A1 (Grönwall)**.** *For an interval $I_0 = [0, \infty)$ and a given function f differentiable everywhere in I_0 and satisfying:*

$$f'(t) \leq \beta(t)f(t), \quad t \in I_0$$

then f is bounded by the corresponding differential equation $g'(t) = \beta(t)g(t)$:

$$f(t) \leq f(0)\int_0^t \beta(s)ds, \quad t \in I_0$$

The bound is nontrivial only if $\mathrm{tr}(\delta C) \geq 0$. This would be natural assumption for a Bayesian model, if C^0 is the prior covariance and the eigenvalues of C^t at $t = \infty$ (corresponding to the posterior) are reduced by the data.

Appendix F.4. Decay of Fluctuation Part of the Free Energy

Still focusing on the Gaussian model, we can further derive a bound on the free energy. It is easy to see that for the Gaussian case, the free energy in Equation (4) separates into a sum of two terms. The first one depends on the mean m^t only and the second one on only the fluctuations (i.e., C^t).

We will consider the second, nontrivial part only. We assume that the covariance matrix is nonsingular (corresponding to $N > D$). The fluctuation part of the free energy (minus its minimum) is given by

$$\mathcal{F}_{fl} = -\frac{1}{2}\ln|I - B| - \frac{1}{2}\mathrm{tr}(B)$$

where we have introduced the matrix $B \doteq I - \Lambda C$. One can show that its eigenvalues are real and are upper bounded by 1. First, we can show from the equations of motion that

$$\frac{d\mathcal{F}_{fl}}{dt} = -\mathrm{tr}(BB^\top) \tag{A7}$$

Second, using the elementary bound $-\ln(1-u) \leq \frac{u}{1-u}$ valid for $u \leq 1$ and applied to the eigenvalues of B yields

$$\begin{aligned}
\mathcal{F}_{fl} \leq & \frac{1}{2}\mathrm{tr}(B(I-B)^{-1} - B) \\
= & \frac{1}{2}\mathrm{tr}(B(I-B)^{-1} - B(I-B)(I-B)^{-1}) \\
= & \frac{1}{2}\mathrm{tr}(B^2(I-B)^{-1}) \\
= & \frac{1}{2}\mathrm{tr}(B^2C^{-1}\Lambda^{-1}) \leq \frac{1}{2}\mathrm{tr}(B^\top\Lambda^{-1}BC^{-1})
\end{aligned}$$

The last two equalities used the definition $B = I - \Lambda C$. Since $B^\top\Lambda^{-1}B$ and C^{-1} are both positive definite, we can bound the last term by (see ([47], Theorem 6.5))

$$\begin{aligned}
\mathcal{F}_{fl} \leq & \frac{1}{2}\mathrm{tr}(B^\top\Lambda^{-1}B)\mathrm{tr}(C^{-1}) \leq \\
& \frac{1}{2}\mathrm{tr}(BB^\top)\mathrm{tr}(\Lambda^{-1})\mathrm{tr}(C^{-1})),
\end{aligned}$$

where, in the last line, we have bounded the trace of a product of p.d. matrices a second time.

Combining with Equation (A7) we show that

$$\frac{d\mathcal{F}_{fl}}{dt} \leq -\frac{2\mathcal{F}_{fl}}{\mathrm{tr}(\Lambda^{-1})\mathrm{tr}(C^{-1})}$$

We can plug in our result from Theorem 2:

$$\begin{aligned}
\mathrm{tr}(C^{-1}) = & \mathrm{tr}(\Lambda) + \mathrm{tr}(C^{-1} - \Lambda) \\
= & \mathrm{tr}(\Lambda) + e^{-2t}\mathrm{tr}((C^0)^{-1} - \Lambda) \\
\leq & \mathrm{tr}(\Lambda) + e^{-2t}|\mathrm{tr}((C^0)^{-1} - \Lambda)| \\
\leq & \mathrm{tr}(\Lambda) + |\mathrm{tr}((C^0)^{-1} - \Lambda)|
\end{aligned}$$

We can plug this in and use Grönwall's Lemma A1 to get an exponential bound

$$\mathcal{F}_{fl}(C^t) \leq \mathcal{F}_{fl}(C^0)e^{-\left[\frac{2t}{\mathrm{tr}(\Lambda^{-1})(\mathrm{tr}(\Lambda)+|\mathrm{tr}((C^0)^{-1}-\Lambda)|)}\right]}.$$

Appendix F.5. Asymptotic Decay of the Free Energy:

For large times t, we can do better. Let us analyse the asymptotic decay constant $\mathcal{F}_{fl} \simeq e^{-\lambda_{free} t}$ defined by

$$\begin{aligned}\lambda_{free} \doteq -\lim_{t\to\infty} \frac{d\ln(\mathcal{F}_{fl})}{dt} &= -\lim \frac{\frac{d\mathcal{F}_{fl}}{dt}}{\mathcal{F}_{fl}} \\ &= \lim \frac{\mathrm{tr}(BB^\top)}{-\frac{1}{2}\ln|I-B| - \frac{1}{2}\mathrm{tr}(B)} \geq \\ &\lim \frac{\mathrm{tr}(B^2)}{-\frac{1}{2}\ln|I-B| - \frac{1}{2}\mathrm{tr}(B)}\end{aligned}$$

In the last inequality, we used $\mathrm{tr}(BB^\top) \geq \mathrm{tr}(B^2)$. Everything is expressed by traces of functions of B, and thus by its eigenvalues. Since $B \to 0$ as $t \to \infty$ (this applies also to its eigenvalues u), we can use Taylor's expansion $\ln(1-u) + u = -u^2/2 + O(u^3)$ to show that

$$\lambda_{free} \geq 4$$

which is independent of Λ.

Appendix G. Proof of Theorem 3: Fixed-Points for Gaussian Model ($N \leq D$)

Theorem A3 (3)**.** *Given a D-dimensional multivariate Gaussian target density $p(x) = \mathcal{N}(x|\mu, \Sigma)$, using Algorithm 2 with $N < D+1$ particles, the empirical mean converges to the exact mean μ. The $N-1$ non-zero eigenvalues of C^t converge to a subset of the target covariance Σ spectrum. Furthermore, the **global minimum** of the regularised version $\widetilde{\mathcal{F}}$ of the free energy (17) corresponds to the **largest** eigenvalues of Σ.*

Applying Equation (A4) to our fixed point equation, we obtain

$$(I - \mathbb{E}_{\hat{q}}\left[g(x)(x-m)^\top\right])(x_i - m) = 0,\ \forall i = 1, \ldots, N$$

Hence, the set of centered positions of the particles $S = \{x_i - m\}_{i=1}^N$, are all eigenvectors of the matrix $\mathbb{E}_{\hat{q}}\left[g(x)(x-m)^\top\right]$ with eigenvalue 1. S spans a $N-1$ dimensional space (we have $\sum_{i=1}^N (x_i - m) = 0$).

If we specialise to a Gaussian target $p(x) = \mathcal{N}(x \mid \mu, \Sigma)$, (and $\Lambda = \Sigma^{-1}$ we have $g(x) = \Lambda(x-\mu)$ and can reuse the result from Equation (A5):

$$\begin{aligned}\mathbb{E}_{\hat{q}}\left[g(x)(x-m)^\top\right] &= \Lambda \mathbb{E}_{\hat{q}}\left[(x-m)(x-m)^\top\right] \\ &= \Lambda C.\end{aligned}$$

Using the equality above, we get:

$$\begin{aligned}\Lambda C(x_i - m) &= (x_i - m) \\ C(x_i - m) &= \Sigma(x_i - m),\ \forall i = 1, \ldots, N\end{aligned}$$

which shows that the obtained low-rank covariance C and the target covariance Σ have $N-1$ eigenvectors and eigenvalues in common.

However, are these the largest ones? We look at the modified free energy (17) (ignoring the contribution of the mean):

$$\min \widetilde{\mathcal{F}} = \min \left\{ -\frac{1}{2} \sum_{i:\lambda_i>0} \ln \lambda_i + \mathrm{tr}(\Lambda C) \right\}$$

where λ_i are the eigenvalues of the empirical covariance C. We first note that $\mathrm{tr}(\Lambda C) = N-1$, independent of which eigenvalues are obtained at the fixed point. This is easily seen by the following argument: If we use the index–set $\mathcal{I}$ for the common eigenvectors e_i and eigenvalues λ_i, $i \in \mathcal{I}$, we can write

$$C = \sum_{i \in \mathcal{I}} e_i \lambda_i e_i^\top$$
$$\Sigma = \sum_{i} e_i \lambda_i e_i^\top$$

From this we obtain

$$\mathrm{tr}(\Lambda C) = \mathrm{tr}(\sum_{i \in \mathcal{I}} e_i \lambda_i^{-1} \lambda_i e^\top) = N-1$$

From this result we obtain

$$\min \widetilde{\mathcal{F}} = \max \frac{1}{2} \sum_{i:\lambda_i>0} \ln \lambda_i - (N-1),$$

The term $N-1$ is a constant, but the first term makes a difference: The **absolute minimum** of $\widetilde{\mathcal{F}}$ is achieved, when the λ_i are $N-1$ **largest** eigenvalues of Σ. Our simulations empirically show that the algorithm usually converges to the absolute minimum.

Appendix H. Dimension-Wise Optimizers

Here, we list some of the most populars optimizers used and their dimension-wise versions. In all algorithms, we consider φ the matrix created by the concatenation of the flow of each particle : $\varphi = [\varphi_1, \ldots, \varphi_N]$, where $\varphi_n = \varphi(x_n)$ We additionally use the notation $\varphi_{n,i}$ for the i-th dimension of the flow of the n-th particle. The main differences between the original algorithms and their modified version were put in red.

Appendix H.1. ADAM

The ADAM algorithm is given by:

Algorithm A1: ADAM

Input: $\varphi^t, m^{t-1}, v^{t-1}, \beta_1, \beta_2, \eta$
Output: Δ
$m^t_{n,d} = \beta_1 m^{t-1}_{n,d} + (1-\beta_1)\varphi^t_{n,d}$
$v^t_{n,d} = \beta_2 v^{t-1}_{n,d} + (1-\beta_2)\left(\varphi^t_{n,d}\right)^2$
$\Delta_{n,d} = \eta \frac{m^t_{n,d}}{(1-\beta_1^t)\left(\sqrt{v^t_{n,d}(1-\beta_2^t)^{-1}}+\epsilon\right)}$

Algorithm A2: Dimension-wise ADAM

Input: $\varphi^t, m^{t-1}, v^{t-1}, \beta_1, \beta_2, \eta$
Output: Δ
$m^t_{n,d} = \beta_1 m^{t-1}_{n,d} + (1-\beta_1)\varphi^t_{n,d}$;
$v^t_d = \beta_2 v^{t-1}_d + (1-\beta_2)\frac{1}{N}\sum_{n=1}^{N}\left(\varphi^t_{n,d}\right)^2$;
$\Delta_{n,d} = \eta \frac{m^t_{n,d}}{(1-\beta_1^t)\left(\sqrt{v^t_d(1-\beta_2^t)^{-1}}+\epsilon\right)}$;

Appendix H.2. AdaGrad

The AdaGrad algorithm is given by:

Algorithm A3: AdaGrad

Input: φ^t, v^{t-1}, η
Output: Δ
$v^t_{n,d} = v^{t-1}_{n,d} + \left(\varphi^t_{n,d}\right)^2$
$\Delta_{n,d} = \eta \frac{\varphi^t_{n,d}}{\sqrt{v^t_{n,d}}+\epsilon}$

Algorithm A4: Dimension-wise AdaGrad

Input: φ^t, v^{t-1}, η
Output: Δ
$v^t_d = v^{t-1}_d + \frac{1}{N}\sum_{n=1}^{N}\left(\varphi^t_{n,d}\right)^2$
$\Delta_{n,d} = \eta \frac{\varphi^t_{n,d}}{\sqrt{v^t_d}+\epsilon}$

Appendix H.3. RMSProp

The RMSProp algorithm is given by:

Algorithm A5: RMSProp

Input: $\varphi^t, v^{t-1}, \rho, \eta$
Output: Δ
$v^t_{n,d} = \rho v^{t-1}_{n,d} + (1-\rho)\left(\varphi^t_{n,d}\right)^2$
$\Delta_{n,d} = \eta \frac{\varphi^t_{n,d}}{\sqrt{v^t_{n,d}}+\epsilon}$

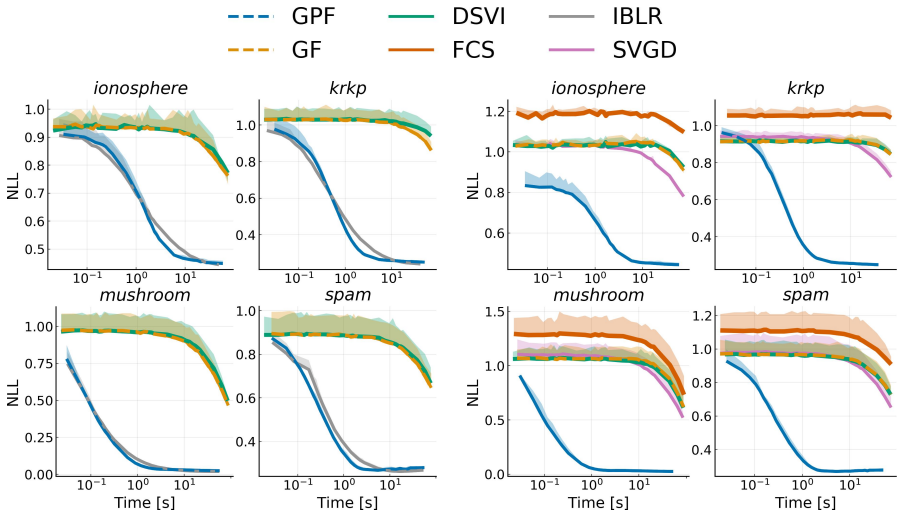
Algorithm A6: Dimension-wise RMSProp

Input: $\varphi^t, v^{t-1}, \rho, \eta$
Output: Δ
$v^t_d = \rho v^{t-1}_d + (1-\rho)\frac{1}{N}\sum_{n=1}^{N}\left(\varphi^t_{n,d}\right)^2$
$\Delta_{n,d} = \eta \frac{\varphi^t_{n,d}}{\sqrt{v^t_d}+\epsilon}$

Appendix I. Additional Figures

Appendix I.1. Bayesian Logistic Regression

Similarly to the previous section, we also show results with the RMSProp optimizer with learning rate 1×10^{-4}.



(**a**) Mean-field approximation (**b**) No mean-field approximation

Figure A1. Similarly to Figure 6, we show the average negative log-likelihood on a test-set over 10 runs against training time on different datasets for a Bayesian logistic regression problem. The dashed curve represents the low-rank approximation with RMSProp for methods based on stochastic estimators.

Appendix I.2. Bayesian Neural Network

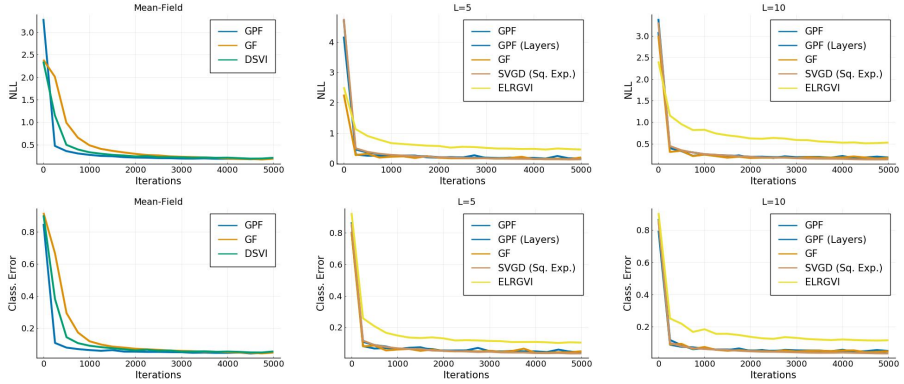


Figure A2. Convergence of the classification error and average negative log-likelihood as a function of time.

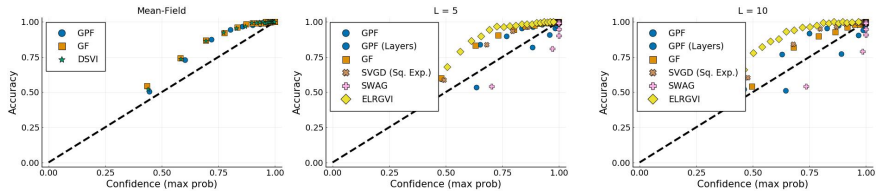


Figure A3. Accuracy vs confidence. Every test sample is clustered in function of its highest predictive probability. The accuracy of this cluster is then computed. A perfectly calibrated estimator would return the identity.

References

1. Shahriari, B.; Swersky, K.; Wang, Z.; Adams, R.P.; de Freitas, N. Taking the human out of the loop: A review of Bayesian optimization. *Proc. IEEE* **2016**, *104*, 148–175. [CrossRef]
2. Settles, B. *Active Learning Literature Survey*; Computer Sciences Technical Report 1648; University of Wisconsin–Madison: Madison, WI, USA, 2009.
3. Sutton, R.S.; Barto, A.G. *Reinforcement Learning: An Introduction*; The MIT Press: Cambridge, MA, USA, 2018.
4. Bardenet, R.; Doucet, A.; Holmes, C. On Markov chain Monte Carlo methods for tall data. *J. Mach. Learn. Res.* **2017**, *18*, 1515–1557.
5. Cowles, M.K.; Carlin, B.P. Markov chain Monte Carlo convergence diagnostics: A comparative review. *J. Am. Stat. Assoc.* **1996**, *91*, 883–904. [CrossRef]
6. Barber, D.; Bishop, C.M. Ensemble learning for multi-layer networks. In *Advances in Neural Information Processing Systems*; MIT Press: Cambridge, MA, USA, 1998; pp. 395–401.
7. Graves, A. Practical Variational Inference for Neural Networks. In Proceedings of the 24th International Conference on Neural Information Processing Systems, Granada, Spain, 12–15 December 2011; Volume 24, pp. 2348–2356.
8. Ranganath, R.; Gerrish, S.; Blei, D. Black box variational inference. In Proceedings of the Seventeenth International Conference on Artificial Intelligence and Statistics, Reykjavik, Iceland, 22–25 April 2014; pp. 814–822.
9. Liu, Q.; Lee, J.; Jordan, M. A kernelized Stein discrepancy for goodness-of-fit tests. In Proceedings of the 33rd International Conference on Machine Learning, New York, NY, USA, 20–22 June 2016; pp. 276–284.
10. Liu, Q.; Wang, D. Stein variational gradient descent as moment matching. In Proceedings of the 32nd International Conference on Neural Information Processing Systems, Montreal, QC, Canada, 3–8 December 2018; Volume 32, pp. 8868–8877
11. Zhuo, J.; Liu, C.; Shi, J.; Zhu, J.; Chen, N.; Zhang, B. Message Passing Stein Variational Gradient Descent. In Proceedings of the 35th International Conference on Machine Learning, Stockholm, Sweden, 10–15 July 2018; Volume 80, pp. 6018–6027.
12. Opper, M.; Archambeau, C. The variational Gaussian approximation revisited. *Neural Comput.* **2009**, *21*, 786–792. [CrossRef] [PubMed]
13. Challis, E.; Barber, D. Gaussian kullback-leibler approximate inference. *J. Mach. Learn. Res.* **2013**, *14*, 2239–2286.
14. Titsias, M.; Lázaro-Gredilla, M. Doubly stochastic variational Bayes for non-conjugate inference. In Proceedings of the 31st International Conference on Machine Learning, Beijing, China, 21–26 June 2014; pp. 1971–1979.
15. Ong, V.M.H.; Nott, D.J.; Smith, M.S. Gaussian variational approximation with a factor covariance structure. *J. Comput. Graph. Stat.* **2018**, *27*, 465–478. [CrossRef]
16. Tan, L.S.; Nott, D.J. Gaussian variational approximation with sparse precision matrices. *Stat. Comput.* **2018**, *28*, 259–275. [CrossRef]
17. Lin, W.; Schmidt, M.; Khan, M.E. Handling the Positive-Definite Constraint in the Bayesian Learning Rule. In Proceedings of the 37th International Conference on Machine Learning, Virtual, 13–18 July 2020; Volume 119, pp. 6116–6126.
18. Hinton, G.E.; van Camp, D. Keeping the Neural Networks Simple by Minimizing the Description Length of the Weights. In Proceedings of the Sixth Annual Conference on Computational Learning Theory, Santa Cruz, CA, USA, 26–28 July 1993; COLT '93; Association for Computing Machinery: New York, NY, USA, 1993; pp. 5–13.
19. Blei, D.M.; Kucukelbir, A.; McAuliffe, J.D. Variational inference: A review for statisticians. *J. Am. Stat. Assoc.* **2017**, *112*, 859–877. [CrossRef]
20. Amari, S.I. Natural Gradient Works Efficiently in Learning. *Neural Comput.* **1998**, *10*, 251–276. [CrossRef]
21. Khan, M.E.; Nielsen, D. Fast yet simple natural-gradient descent for variational inference in complex models. In Proceedings of the International Symposium on Information Theory and Its Applications (ISITA), Singapore, 28–31 October 2018; pp. 31–35.
22. Lin, W.; Khan, M.E.; Schmidt, M. Fast and simple natural-gradient variational inference with mixture of exponential-family approximations. In Proceedings of the 36th International Conference on Machine Learning, Long Beach, CA, USA, 9–15 June 2019; pp. 3992–4002.
23. Salimbeni, H.; Eleftheriadis, S.; Hensman, J. Natural Gradients in Practice: Non-Conjugate Variational Inference in Gaussian Process Models. In Proceedings of the Twenty-First International Conference on Artificial Intelligence and Statistics, Lanzarote, Canary Islands, 9–11 April 2018; pp. 689–697.
24. Liu, Q.; Wang, D. Stein variational gradient descent: A general purpose bayesian inference algorithm. *arXiv* **2016**, arXiv:1608.04471.
25. Ba, J.; Erdogdu, M.A.; Ghassemi, M.; Suzuki, T.; Sun, S.; Wu, D.; Zhang, T. Towards Characterizing the High-dimensional Bias of Kernel-based Particle Inference Algorithms. In Proceedings of the 2nd Symposium on Advances in Approximate Bayesian Inference, Vancouver, BC, Canada, 8 December 2019.
26. Tomczak, M.; Swaroop, S.; Turner, R. Efficient Low Rank Gaussian Variational Inference for Neural Networks. In Proceedings of the Advances in Neural Information Processing Systems, Virtual, 6–12 December 2020; Volume 33.
27. Maddox, W.J.; Izmailov, P.; Garipov, T.; Vetrov, D.P.; Wilson, A.G. A simple baseline for bayesian uncertainty in deep learning. In Proceedings of the Advances in Neural Information Processing Systems, Vancouver, BC, Canada, 8–14 December 2019; pp. 13153–13164.
28. Evensen, G. Sequential data assimilation with a nonlinear quasi-geostrophic model using Monte Carlo methods to forecast error statistics. *J. Geophys. Res. Oceans* **1994**, *99*, 10143–10162. [CrossRef]

29. Rezende, D.; Mohamed, S. Variational inference with normalizing flows. In Proceedings of the 32nd International Conference on Machine Learning, Lille, France, 7–9 July 2015; pp. 1530–1538.
30. Chen, R.T.; Rubanova, Y.; Bettencourt, J.; Duvenaud, D. Neural ordinary differential equations. In Proceedings of the 32nd International Conference on Neural Information Processing, Montréal, QC, Canada, 3–8 December 2018; pp. 6572–6583.
31. Ingersoll, J.E. *Theory of Financial Decision Making*; Rowman & Littlefield: Lanham, MD, USA, 1987; Volume 3.
32. Barfoot, T.D.; Forbes, J.R.; Yoon, D.J. Exactly sparse gaussian variational inference with application to derivative-free batch nonlinear state estimation. *Int. J. Robot. Res.* **2020**, *39*, 1473–1502. [CrossRef]
33. Korba, A.; Salim, A.; Arbel, M.; Luise, G.; Gretton, A. A Non-Asymptotic Analysis for Stein Variational Gradient Descent. In Proceedings of the 32nd International Conference on Neural Information Processing, Virtual, 6–12 December 2020; Volume 33. pp. 4672–4682.
34. Berlinet, A.; Thomas-Agnan, C. *Reproducing Kernel Hilbert Spaces in Probability and Statistics*; Springer Science & Business Media: Berlin/Heidelberg, Germany, 2011.
35. Zaki, N.; Galy-Fajou, T.; Opper, M. Evidence Estimation by Kullback-Leibler Integration for Flow-Based Methods. In Proceedings of the Third Symposium on Advances in Approximate Bayesian Inference, Virtual Event, January–February 2021.
36. Bezanson, J.; Edelman, A.; Karpinski, S.; Shah, V.B. Julia: A fresh approach to numerical computing. *SIAM Rev.* **2017**, *59*, 65–98. [CrossRef]
37. Tieleman, T.; Hinton, G. *Lecture 6.5-rmsprop, Coursera: Neural Networks for Machine Learning*; Technical Report; University of Toronto: Toronto, ON, USA, 2012.
38. Zhang, G.; Li, L.; Nado, Z.; Martens, J.; Sachdeva, S.; Dahl, G.; Shallue, C.; Grosse, R.B. Which Algorithmic Choices Matter at Which Batch Sizes? Insights From a Noisy Quadratic Model. In *Advances in Neural Information Processing Systems*; Wallach, H., Larochelle, H., Beygelzimer, A., d'Alché-Buc, F., Fox, E., Garnett, R., Eds.; Curran Associates, Inc.: Red Hook, NY, USA 2019; Volume 32, pp. 8196–8207.
39. Kingma, D.P.; Ba, J. Adam: A method for stochastic optimization. *arXiv* **2014**, arXiv:1412.6980.
40. Dua, D.; Graff, C. UCI Machine Learning Repository. 2017. Available online: https://archive.ics.uci.edu/ml/datasets.php (accessed on 28 July 2021).
41. Agarap, A.F. Deep learning using rectified linear units (relu). *arXiv* **2018**, arXiv:1803.08375.
42. LeCun, Y. The MNIST Database of Handwritten Digits. Available online: http://yann.lecun.com/exdb/mnist/ (accessed on 20 July 2021).
43. Guo, C.; Pleiss, G.; Sun, Y.; Weinberger, K.Q. On calibration of modern neural networks. In Proceedings of the 34th International Conference on Machine Learning, Sydney, Australia, 6–11 August 2017; pp. 1321–1330.
44. Liu, C.; Zhuo, J.; Cheng, P.; Zhang, R.; Zhu, J. Understanding and accelerating particle-based variational inference. In Proceedings of the 36th International Conference on Machine Learning, Long Beach, CA, USA, 9–15 June 2019; pp. 4082–4092.
45. Zhu, M.H.; Liu, C.; Zhu, J. Variance Reduction and Quasi-Newton for Particle-Based Variational Inference. In Proceedings of the 37th International Conference on Machine Learning, Virtual, 13–18 July 2020.
46. Gronwall, T.H. Note on the derivatives with respect to a parameter of the solutions of a system of differential equations. *Ann. Math.* **1919**, *20*, 292–296. [CrossRef]
47. Zhang, F. *Matrix Theory: Basic Results and Techniques*; Springer Science & Business Media: Berlin/Heidelberg, Germany, 2011.

7

Discussions and extensions

This chapter presents both discussions and extensions on the models and ideas presented in Chapters 3, 4, 5. All figures presented are reproducible by running the examples provided in the GitHub repository `https://github.com/theogf/Phd-Thesis`. Section 7.1 considers how augmentations can be generalized further and what analysis we need to fully understand the improvement brought by augmentations. Section 7.2 presents new augmented models for $\mathcal{GP}$ regression with heteroscedastic noise. Section 7.3 explores how HMC could be used (or not) with augmented models. Section 7.4 shows how the multi-class model of Chapter 4 can be improved in multiple ways. Section 7.5 presents a way to combine inducing points and sampling using augmentations. Finally, Section 7.6 consider more largely the limitations existing with our augmentation approach.

7.1 Further generalizations and understanding

The works presented in this thesis only scratched the surface of how helpful mixtures and representations are.

Moment Generating Functions

We are still exploring ways to identify larger classes of functions identifiable as scale mixtures or hierarchical mixtures. Already mentioned in Chapters 4 and 5, the connection with the Moment Generating Function (MGF) of a distribution is a promising direction. We already identified augmentable functions as being a transformed MGF of the augmented variables in Chapter 5:

$$\varphi(x^2) = \int_0^\infty e^{-x^2\omega} p(\omega) d\omega \forall x \in \mathbb{R} \;\equiv\; \mathrm{MGF}_{p(\omega)}(x) = \varphi(-\sqrt{x}), \qquad \forall x \geq 0.$$

However, this is limited to MGF of continuous variables with a square transformation on the inputs. We can extend the notion of augmentable functions to MGF of discrete and multivariate distributions, where the domain of ω is not always $\mathbb{R}^+$. For example, we used the MGF of a Poisson distribution in Chapter 4:

$$\exp(\lambda(e^x - 1)) = \sum_{n=0}^{\infty} e^{ne^x} \mathrm{Po}(n|\lambda).$$

It is not a scale mixture of Gaussians, but with the right variable transformations, it can still be useful. The MGF of a Poisson is known, but we could also consider arbitrary MGF since we are able to sample from a distribution given its Laplace transform only [47].

The MGF is also an interesting tool for creating hierarchical models. Since the MGF is of the form $\sum_x e^{tx}p(x)$ or $\int e^{tx}p(x)dx$, by setting $t = \log \sigma(f)$, we get scales mixtures of the form $\sum_x \sigma^x(f)$. Thanks to the property that $\sigma^n(f)$ is augmentable for any $n \in \mathbb{R}^+$, we can use Pólya-Gamma variables and obtain a conditionally conjugate model for a $\mathcal{GP}$. Additional examples of such constructions are shown in this chapter in Sections 7.2 and 7.4.

Marginalizing out augmented variables

A potential improvement for augmented models is the identification of marginalizable augmented variables that keep the conditional conjugacy of the model. For example, in the multi-class model from Chapter 4, the augmented variable λ can be marginalized out, as shown in Section 7.4. We can reduce the dimensionality of the model and avoid tricky situations like the inner loop updates in Chapter 4. This marginalization step is avoidable by identifying the right MGF from the start. As shown in Section 7.2.2, switching between marginalized and augmented models gives great inference flexibility.

Convergence speed analysis

An unfinished work (despite trying) is to establish convergence rates (error as a function of the number of iterations) for the CAVI algorithm and derive theoretical bounds on the intra-chain correlation and of the ergodicity for the Gibbs sampler. Experimental results indicate that the error on the variational free energy (and variational parameters) is decreasing as $\|\mathcal{F}^* - \mathcal{F}^t\| \propto C_0 e^{-ct}$, where t is the number of iterations, but we did not manage to write a formal proof. We show the decay for both the variational free energy and the variational parameters for different examples in Figure 7.1.

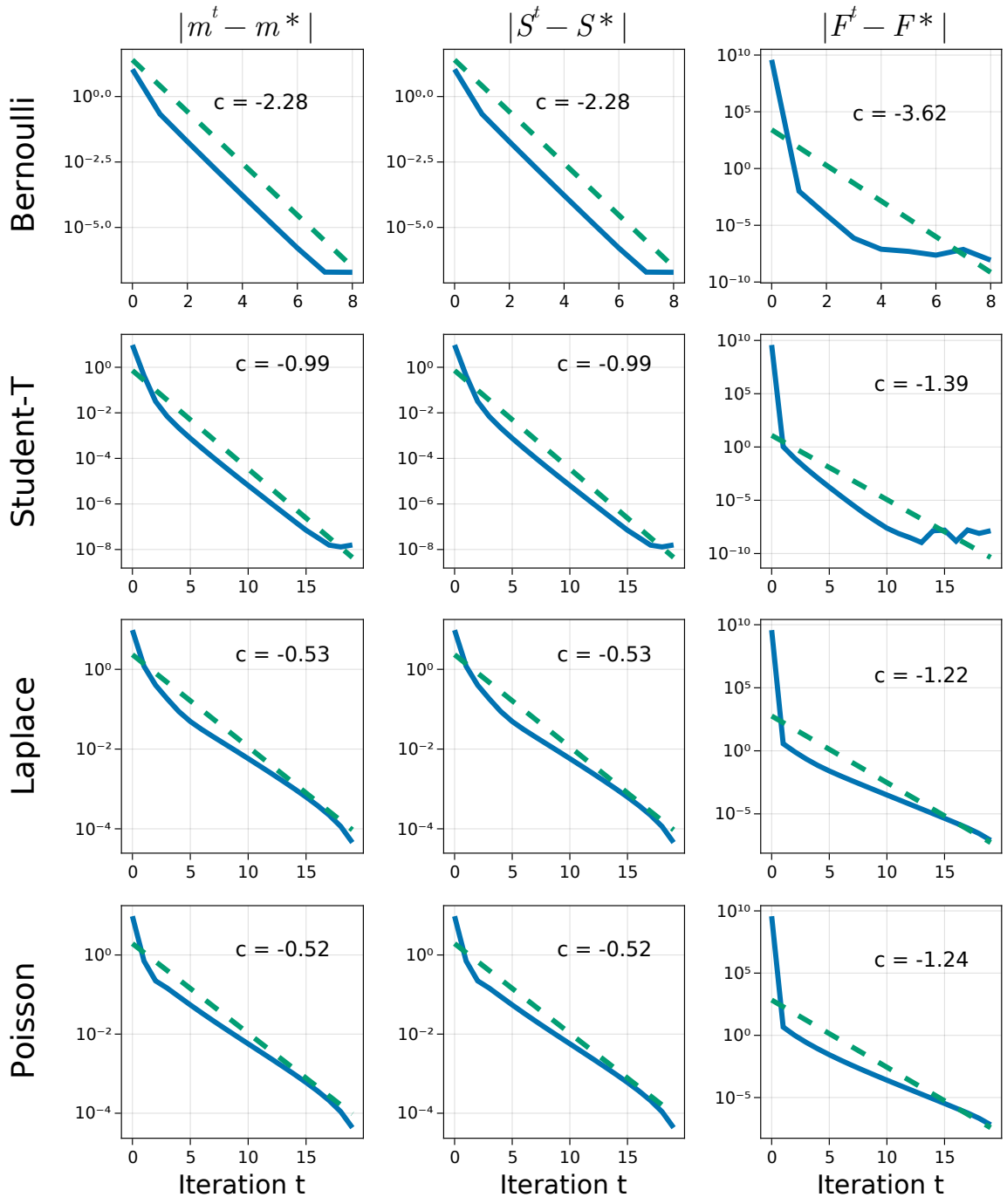


Figure 7.1: Convergence plot of the CAVI updates for a one-dimensional toy example with different likelihoods (y-axis in log scale). The solid blue line shows the empirical error over the number of iterations and the dashed green line shows the fit of the function $C_0 \exp(ct)$. The exponential coefficient is written down explicitly for each likelihood.

7.2 Double bounds for intricate latent $\mathcal{GP}$s

The multi-class model developed in Chapter 4 paves the way to work with multi-latent models and hierarchical augmentations. Based on this idea, we developed another multi-latent model on the heteroscedastic regression likelihood [58, 32]. It models simultaneously the mean and variance of a regression likelihood with two latent $\mathcal{GP}$**s** f and g. We consider both Gaussian and Non-Gaussian

likelihoods since we can stack augmentations. We start with the simplest model: the heteroscedastic Gaussian likelihood.

7.2.1 Heteroscedastic Gaussian Likelihood

A crucial model choice is the function mapping g to the likelihood variance ϵ^2. The exponential link, i.e. $\epsilon^2(x) = \exp(g(x))$, is the most popular, however to be able to apply our augmentations, we use the link $\epsilon^2(x) = (\lambda\sigma(g(x)))^{-1}$. Let's look at the case of the heteroscedastic Gaussian likelihood, defined as:

$$p(y|f, g, \lambda) = \frac{\sqrt{\lambda\sigma(g)}}{\sqrt{2\pi}} \exp\left(-\frac{\lambda\sigma(g)(y-f)^2}{2}\right). \tag{7.1}$$

The augmentations for this likelihood are straightforward and quite similar to the multi-class ones from Chapter 4.

$$\begin{aligned}\exp\left(-\frac{\lambda\sigma(g)(y-f)^2}{2}\right) &= \exp\left(\frac{\lambda(\sigma(-g)-1)(y-f)^2}{2}\right) \\ &= \sum_{n=0}^{\infty} \sigma^n(-g)\mathrm{Po}\left(n \;\middle|\; \frac{\lambda(y-f)^2}{2}\right),\end{aligned} \tag{7.2}$$

where we used the MGF of the Poisson distribution. Using the Pólya-Gamma augmentation and the additivity property of Pólya-Gamma variables, we get the final augmented likelihood:

$$p(y, n, \omega \mid f, g, \lambda) = \frac{\sqrt{\lambda}}{2^n\sqrt{\pi}} \exp\left(\frac{1}{2}\left(g\left(\frac{1}{2}-n\right) - \frac{g^2}{\omega}\right)\right) \mathrm{PG}\left(\omega \mid \frac{1}{2}+n, 0\right) \mathrm{Po}\left(n \mid \lambda\frac{(y-f)^2}{2}\right) \tag{7.3}$$

The interesting part about this augmented likelihood (7.3) is that although it is conditionally conjugate in g, ω, and n, it is unclear how to infer f: it is quadratic in g but not in f. It turns out that the Gibbs sampler for this model is very simple: We take the augmented likelihood $p(y, \omega, n|f, g, \lambda)$, marginalize out n and ω and, as expected, we get the original likelihood (7.1), which is conditionally conjugate with f. The conditional $p(f|y, g, \lambda)$ on this likelihood is the **collapsed conditional**. In a Gibbs sampling scheme, this allows us to perform a **collapsed step**. We give all the Gibbs sampling steps in Algorithm 2. So far, we have excluded the λ parameter from inference. By putting a Gamma prior $\mathrm{Ga}(\lambda|\alpha, \beta)$, where α is the shape and β is the rate, the collapsed conditional is available in closed-form:

$$p(\lambda|\boldsymbol{f}, \boldsymbol{g}, \boldsymbol{y}) = \mathrm{Ga}(\lambda|\alpha + \frac{N}{2}, \beta + \sum_{i=1}^{N} \frac{\sigma(g^i)}{2}(y^i - f^i)^2).$$

As underlined in Section 2.3.2, the CAVI updates need the model's full conditionals and are not compatible with collapsed conditionals. To solve this problem, we need to reverse-engineer how CAVI updates are obtained and start with a first bound on the KL divergence:

$$\begin{aligned}\mathrm{KL}\left(q(\boldsymbol{f})q(\boldsymbol{g})||p(\boldsymbol{f}, \boldsymbol{g}|\boldsymbol{y})\right) &\leq \min_{q(\boldsymbol{g})} -\mathbb{E}_{q(\boldsymbol{g})}\left[\mathbb{E}_{q(\boldsymbol{f})}\left[\log p(\boldsymbol{y}|\boldsymbol{f}, \boldsymbol{g})\right]\right] + \mathrm{KL}\left(q(\boldsymbol{f})q(\boldsymbol{g})||p(\boldsymbol{f})p(\boldsymbol{g})\right) - \log p(\boldsymbol{y}) \\ &= \min_{q(\boldsymbol{g})} -\mathbb{E}_{q(\boldsymbol{g})}\left[\log p(\boldsymbol{y}|\boldsymbol{g}, \boldsymbol{\mu}_f^*, \boldsymbol{\Sigma}_f^*)\right] + \mathrm{KL}\left(q(\boldsymbol{g})||p(\boldsymbol{g})\right) + \mathrm{KL}_f^* - \log p(\boldsymbol{y}) = \mathcal{F}_1.\end{aligned}$$

$p(\boldsymbol{y}|\boldsymbol{g}, \boldsymbol{\mu}_f^*, \boldsymbol{\Sigma}_f^*)$ and KL_f^* are expectations computed with the optimal $q^*(\boldsymbol{f}) = \mathcal{N}\left(\boldsymbol{f}|\boldsymbol{\mu}_f^*, \boldsymbol{\Sigma}_f^*\right)$. We can now use the augmentations from Equation (7.3) on the expected log-likelihood, where we replaced $(y_i - f_i)^2$ by $(y_i - (\boldsymbol{\mu}_f^*)_i)^2 + (\Sigma_f^*)_{ii}$, and build a second bound.

$$\mathcal{F}_1 \leq \min_{q(\boldsymbol{g})q(\boldsymbol{\omega}, \boldsymbol{n})} \mathbb{E}_{q(\boldsymbol{g})q(\boldsymbol{\omega}, \boldsymbol{n})}\left[\log p(\boldsymbol{\omega}, \boldsymbol{n}, \boldsymbol{y}|\boldsymbol{g}, \boldsymbol{\mu}_f^*, \boldsymbol{\Sigma}_f^*)\right] + \mathrm{KL}\left(q(\boldsymbol{g})||p(\boldsymbol{g})\right) + \mathrm{KL}_f^* = \mathcal{F}_2 \tag{7.4}$$

It is straightforward to find the optimal variational distributions $q^*(\boldsymbol{g})$ and $q^*(\boldsymbol{\omega}, \boldsymbol{n})$ minimizing $\mathcal{F}_2$ which allows us to use CAVI updates. Then, injecting the optimal distribution $q^*(\boldsymbol{g})q(\boldsymbol{\omega}, \boldsymbol{n})$ in $\mathcal{F}_2$, we can derive the optimal $\boldsymbol{\mu}_f^*$ and $\boldsymbol{\Sigma}_f^*$, obtainable in closed-form. The resulting CAVI updates are given in Algorithm 3. For λ, we can use the second bound (7.4) and obtain a closed-form maximum-likelihood estimate, given in Algorithm 3.

This double-bound approach is very similar to Lázaro-Gredilla and Titsias [32], although they are using the exponential link and need some extra computations.

Algorithm 2 Gibbs sampling for the Heteroscedastic Gaussian likelihood

input: $\boldsymbol{f}, \boldsymbol{g}, \lambda, \boldsymbol{y}$, $p(\boldsymbol{f}, \boldsymbol{g}) = \mathcal{N}(\boldsymbol{f}|\boldsymbol{\mu}_f^0, K)\mathcal{N}(\boldsymbol{g}|\boldsymbol{\mu}_g^0, K)$, $p(\lambda|\alpha, \beta)$.

for t in 1 : N samples **do**

Draw $\lambda \sim p(\lambda|\boldsymbol{f}, \boldsymbol{g}, \boldsymbol{y}) = \text{Ga}(\lambda|\alpha + \frac{N}{2}, \beta + \sum_{i=1}^N \frac{\sigma(g^i)}{2}(y^i - f^i)^2)$.

Draw $n^i \sim p(n^i|f^i, g^i, \lambda) = \text{Po}(\lambda\sigma(-g^i)\frac{(y^i - f^i)^2}{2})$

Draw $\omega^i \sim p(\omega^i|n^i, g^i) = \text{PG}(0.5 + n^i, |g^i)$

Draw $\boldsymbol{g} \sim p(\boldsymbol{g}|\boldsymbol{n}, \boldsymbol{\omega}) = \mathcal{N}(\boldsymbol{\mu}_g, \boldsymbol{\Sigma}_g)$

where $\boldsymbol{\Sigma}_g = \left(K^{-1} + \text{diag}(\boldsymbol{\omega})\right)^{-1}$ and $\boldsymbol{\mu}_g = \boldsymbol{\Sigma}_g \left(K^{-1}\boldsymbol{\mu}_g^0 + \frac{0.5 - \boldsymbol{n}}{2}\right)$

Draw $\boldsymbol{f} \sim p(\boldsymbol{f}|\boldsymbol{g}, \lambda) = \mathcal{N}(\boldsymbol{\mu}_f, \boldsymbol{\Sigma}_f)$

where $\boldsymbol{\Sigma}_f = \left(K^{-1} + \lambda\text{diag}(\sigma(\boldsymbol{g}))\right)^{-1}$ and $\boldsymbol{\mu}_f = \boldsymbol{\Sigma}_f \left(K^{-1}\boldsymbol{\mu}_f^0 + \lambda\text{diag}(\sigma(\boldsymbol{g}))\frac{\boldsymbol{y}}{2}\right)$

end for

Algorithm 3 CAVI Updates for the Heteroscedastic Gaussian likelihood

input: $q(\boldsymbol{f}, \boldsymbol{g}) = \mathcal{N}(\boldsymbol{f}|\boldsymbol{\mu}_f, \boldsymbol{\Sigma}_f)\mathcal{N}(\boldsymbol{g}|\boldsymbol{\mu}_g, \boldsymbol{\Sigma}_g)$, $p(\boldsymbol{f}, \boldsymbol{g}) = \mathcal{N}(\boldsymbol{f}|\boldsymbol{\mu}_f^0, K)\mathcal{N}(\boldsymbol{g}|\boldsymbol{\mu}_g^0, K)$, $\boldsymbol{y}$ and λ.

while convergence criteria is not met **do**

$\psi^i = \widetilde{\sigma}(q(g^i))$

$\lambda = \frac{N}{\sum_{i=1}^N (1 - \psi^i)\sqrt{(y^i - \mu_f^i)^2 + \Sigma_f^{ii}}}$

$\gamma^i = \frac{\lambda}{2}\psi^i\sqrt{(y^i - \mu_f^i)^2 + \Sigma_f^{ii}}$

$c^i = \sqrt{(\mu_g^i)^2 + \Sigma_g^{ii}}$

$\theta^i = \mathbb{E}_{q(\omega^i|n^i)q(n^i)}\left[\omega^i\right] = \frac{0.5 + \gamma^i}{2c^i}\tanh\left(\frac{c^i}{2}\right)$

$\boldsymbol{\Sigma}_f = \left(K^{-1} + \lambda\text{diag}(1 - \boldsymbol{\psi})\right)^{-1}$

$\boldsymbol{\mu}_f = \boldsymbol{\Sigma}_f \left(K^{-1}\boldsymbol{\mu}_f^0 + \lambda\text{diag}(1 - \boldsymbol{\psi})\boldsymbol{y}\right)$

$\boldsymbol{\Sigma}_g = \left(K^{-1} + \text{diag}(\boldsymbol{\theta})\right)^{-1}$

$\boldsymbol{\mu}_g = \boldsymbol{\Sigma}_g \left(K^{-1}\boldsymbol{\mu}_g^0 + \frac{0.5 + \boldsymbol{\gamma}}{2}\right)$

end while

where $q(\boldsymbol{n}, \boldsymbol{\omega}) = \prod_{i=1}^N \text{PG}(\omega^i|0.5 + n, c^i)\text{Po}(n^i|\gamma^i)$ and $\widetilde{\sigma}(q(g^i)) = \frac{e^{-\mu_g^i/2}}{\sqrt{(\mu_g^i)^2 + \Sigma_k^{ii}/2}}$ can be seen as a close approximation to $\mathbb{E}_{q(g^i)}\left[\sigma(-g^i)\right]$.

A 1-dimensional toy example is shown in Figure 7.2 with the results of the inference algorithms.

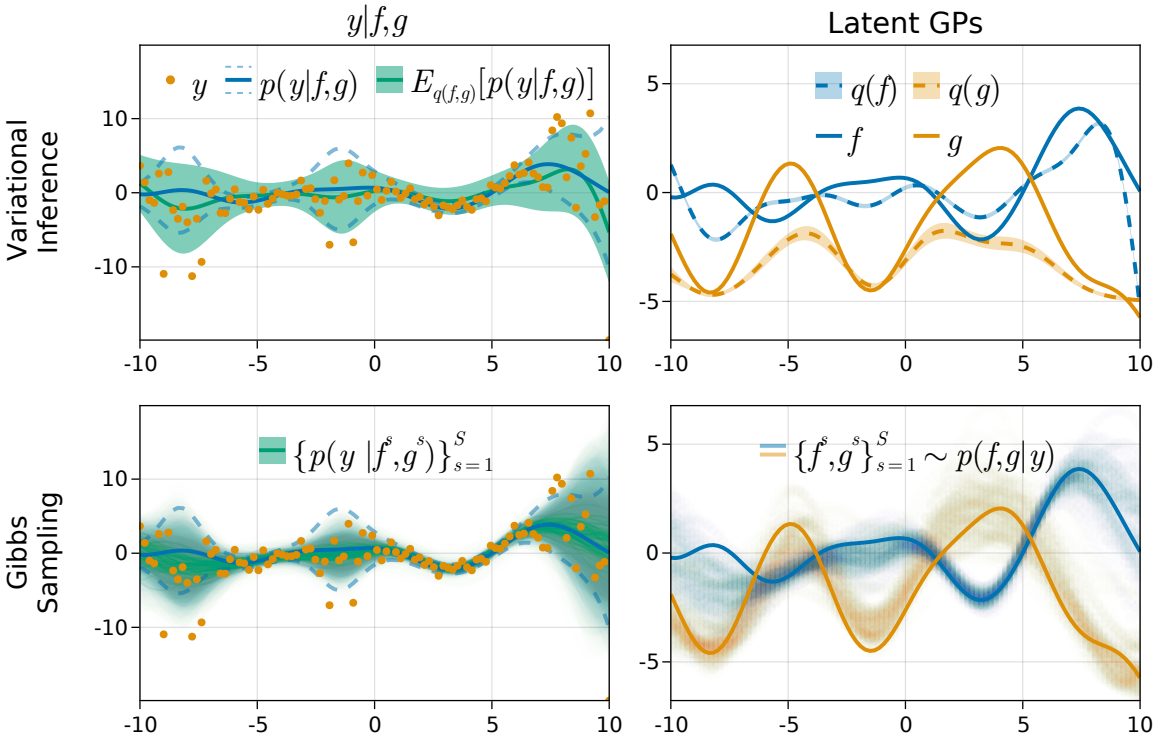


Figure 7.2: Toy example of a heteroscedastic Gaussian regression problem and the resulting inference from Algorithm 2 (Gibbs sampling, bottom plots) and Algorithm 3 (Variational Inference, top plots). The left plots show the output space. The training data $\boldsymbol{y}$ are in orange, the generating likelihood is shown in blue (mean in solid line and one standard deviation in dashed-line). The green bands show the predictive distributions with one standard deviation obtained after posterior inference (one band for variational inference and cumulative bands for the sampling approach). The right plots show the true latent functions f and g used to generate y as well as the inferred posteriors: variational on top (mean with one standard deviation) and samples at the bottom.

We can see that on this one-dimensional example, VI but more particularly Gibbs sampling, manage to recover the original model. For VI, the variance on the latent f is almost negligible since all the data variance is absorbed into the likelihood variance term. The samples obtained with Gibbs sampling, without any warmup, fit nicely the true processes of f and g.

An implementation as well as detailed derivations are in the AugmentedGPLikelihoods.jl package [15].

7.2.2 Heteroscedastic Non-Gaussian Likelihood

This method extends to non-Gaussian likelihoods as well. We take the example of the heteroscedastic Student-t likelihood, where we have a local scale with standard deviation $\epsilon(x) = \lambda\sigma(g)$ with $\lambda \in \mathbb{R}^+$.

Similar to the heteroscedastic Gaussian likelihood (7.1), we get the likelihood:

$$p(y|f,g,\lambda,\nu) = \frac{\Gamma(\frac{\nu+1}{2})\sqrt{\lambda\sigma(g)}}{\Gamma(\frac{\nu}{2})\sqrt{\pi\nu}}\left(1+\lambda\sigma(g)\frac{(y-f)^2}{\nu}\right)^{-\frac{\nu+1}{2}} \tag{7.5}$$

To simplify the notation, we define the scaled residuals $\Delta = \Delta_\nu(f, y, \lambda) = \frac{\lambda(y-f)^2}{\nu}$, $\alpha = \frac{\nu+1}{2}$ and the normalization constant $Z = \frac{\Gamma(\frac{\nu+1}{2})}{\Gamma(\frac{\nu}{2})}$. We can proceed with the first augmentation:

$$\begin{aligned}(1+\sigma(g)\Delta)^{-\alpha} =& (1+\Delta(1-\sigma(-g)))^{-\alpha} \\ =& (1+\Delta-\Delta\sigma(-g))^{-\alpha} \\ =& (\Delta\sigma(-g))^{-\alpha}\left(\frac{\sigma(-g)\Delta}{1+\Delta-\sigma(-g)\Delta}\right) \\ =& \sum_{k=0}^{\infty}\Delta^k \mathrm{NB}(k|\sigma(-g),\alpha), \end{aligned} \tag{7.6}$$

where we used the MGF of the **Negative Binomial** distribution.

We obtain the same result by performing first the augmentation of the Student-t with a Gamma variable:

$$\begin{aligned} p(y|f,g,\lambda) =& \int_0^\infty \mathcal{N}(y|f,(\lambda\sigma(g)\gamma)^{-1})\mathrm{IG}(\gamma|\frac{\nu}{2},\frac{\nu}{2})d\gamma \\ p(y,\gamma|f,g,\lambda) =& \mathcal{N}(y|f,(\lambda\sigma(g)\gamma)^{-1})\mathrm{IG}\left(\gamma|\frac{\nu}{2},\frac{\nu}{2}\right). \end{aligned} \tag{7.7}$$

$\mathcal{N}(y|f,(\lambda\sigma(g)\gamma)^{-1})$ is the same starting point as Equation (7.1) with an additional scaling γ. The next augmentation steps are the same as in Equation (7.2) with an augmentation with a Poisson variable. Marginalizing out the Gamma variable γ results in a Negative Binomial distribution.

Back to Equation (7.6), we rework the likelihood by reorganizing the terms in the augmented likelihood.

$$\begin{aligned} p(y,k|f,g,\lambda,\nu) =& Z\sqrt{\lambda}\sigma^{\frac{1}{2}}\sigma(-g)^{-\alpha}\Delta^{-\alpha}\Delta^k \underbrace{C(k,\alpha)\sigma^\alpha(g)\sigma(-g)^k}_{\mathrm{NB}(k|\sigma(-g),\alpha)} \\ =& ZC(k,\alpha)\sqrt{\lambda}(\sigma(g))^{\frac{1}{2}+\alpha}(\sigma(-g))^{k-\alpha}\Delta^{k-\alpha} \end{aligned}$$

where $C(k,\alpha) = \frac{\Gamma(r+k)}{k!\Gamma(r)}$ is the normalization constant of the negative binomial. We set $Z' = ZC(\alpha,k)\sqrt{\lambda}$ as a constant independent of f or g. The final step is the Pólya-Gamma augmentation:

$$p(y,k,\omega|f,g,\lambda,\nu) = Z'\Delta^{k-\alpha}2^{-(\frac{1}{2}+k)}\exp\left(\frac{1}{2}\left(\frac{1}{2}+2\alpha-k\right)g+g^2\omega\right)\mathrm{PG}(\omega|\frac{1}{2}+k,0). \tag{7.8}$$

Like for the heteroscedastic Gaussian likelihood, the augmented likelihood (7.8) is conjugate in g but not in f. We can find the collapsed conditional for f in closed-form.

The key to performing inference on this augmented model, is to use the right augmented likelihood for each variable. For example, for f and λ, we only want to use the Inverse Gamma augmentation described in Equation (7.7). For ω, g, and k (used as a mixture of inverse Gamma and Poisson) we will use the fully augmented likelihood (7.8). This will give a combination of collapsed conditionals and full conditionals directly usable in a Gibbs sampling scheme. For the CAVI updates, we reuse the double bound idea of Section 7.2.1.

The full derivations, resulting algorithms and implementation will be found in the `AugmentedGPLikelihoods.jl` package [15].

7.3 Using Hamilton Monte Carlo on the augmented model

The Gibbs sampler in the experiments of Chapters 3 and 5 outperforms the state-of-the-art HMC algorithm introduced in Section 2.3.1. A recurrent question I got is: Is the performance gain due only

to the augmentation or the Gibbs sampling scheme? To answer this question, we try using the HMC algorithm on augmented models.

Before doing any experiments, let us consider the consequences that the augmented model has on the HMC sampler. First, the augmentation increases the dimensionality of the model. For N observations, we need KN more dimensions (where K depends on the model); therefore, gradient computations and algorithm tuning should be more expensive. On the other hand, since the likelihood is simplified to a quadratic problem, the computational complexity of each step can decrease! The second issue with using HMC on the augmented model is that the probability distribution function (pdf) of the prior distribution on augmented variables is not always available in closed-form or not usable at all. For example, one approximates the probability of a Pólya-Gamma variable with a truncated alternating series, Truncated series are computationally expensive and can also be biased and unstable! My experience with the Pólya-Gamma variables is that even when using tricks like "logsumexp" to improve numerical stability, the pdf approximation can be negative, breaking the computations. Finally, the critical problem with HMC is that it only works with continuous variables. Some augmentations directly involve discrete variables like the Poisson in the multi-class setting, making it incompatible with a scheme involving only HMC.

We try running HMC and NUTS with a compatible augmentation (augmented variable pdf known in closed-form, no discrete variables). Figure 7.3 shows the auto-correlation plots on $\mathcal{GP}$ regression problem with a Student-t likelihood with $\nu = 3$ degrees of freedom applied on the Boston housing dataset (506 data points, 13 dimensions) [19]. We draw one chain of 2000 samples (plus 500 adaptation samples for HMC and NUTS) for both the original and augmented model.

From the first look, HMC applied on the augmented model has a lower auto-correlation. When using NUTS, the gain becomes less clear. Moreover, the algorithm produces antithetic chains, making it harder to have a proper comparison. The Gibbs sampler has the smallest intra-chain correlation, but one could argue that negative correlations are desirable to compute expectations. However, HMC and NUTS turned out to be much slower than the Gibbs sampler: the Gibbs sampler took around 20 sec to run against an average of 12 minutes for HMC and NUTS. This difference is due to HMC (and NUTS) needing to compute many gradients for every sample. Perhaps surprisingly, there was no significant time difference between the augmented and original models for HMC and NUTS.

Note that HMC is already, in a sense, making an augmentation of its own with the momentum variables, and it could be added to the list of successful types of augmentations improving inference.

We should only consider these results preliminary since we used a simple likelihood, and the dataset is relatively small and easy.

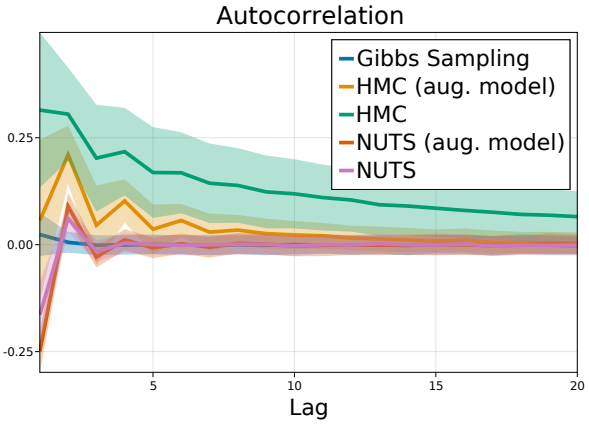


Figure 7.3: Auto-correlation function of the Gibbs sampler, HMC and NUTS on the augmented model, and HMC and NUTS on the original model. The mean is shown with one standard-deviation over all dimensions.

7.4 Improvements on the Multi-Class Classification

We recently figured out additional ways to improve the multi-class classification model and the associated inference. We present them here in 3 different sections.

7.4.1 Marginalizing out variables

In the augmentation derived in Chapter 4, we add $2K+1$ new variables per observation: λ, $\{n_i\}_{i=1}^K$ and $\{\omega_i\}_{i=1}^K$. However, we can reduce this number to $2K$ and avoid unnecessary inner loops by marginalizing out λ. When deriving the augmentations, one ends up with the following augmented likelihood:

$$p(y=k, \{n_j\}_{j=1}^K, \lambda | \{f_j\}_{j=1}^K) = \sigma(f_k) \prod_{j=1}^K \sigma(-f_j)^{n_j} \mathrm{Po}(n_j|\lambda), \tag{7.9}$$

where we omitted the improper prior $1_{[0,\infty)}$ on λ. We can marginalize out λ:

$$\begin{aligned}
\int_0^\infty \prod_{j=1}^K \sigma(-f_j)^{n_j} \mathrm{Po}(n_j|\lambda) d\lambda =& \frac{1}{\prod_{j=1}^K n_j!} \int_0^\infty \lambda^{\sum_{j=1}^K n_j} e^{-K\lambda} d\lambda \\
=& \frac{K^{-\sum_{j=1}^K n_j}}{\prod_{j=1}^K n_j!} \prod_{j=1}^K \sigma(-f_j)^{n_j} \int_0^\infty (K\lambda)^{\sum_{j=1}^K n_j} e^{-K\lambda} d\lambda \\
=& \prod_{j=1}^K \sigma(-f_j)^{n_j} \Gamma(1 + \sum_{j=1}^K n_j) \prod_{j=1}^K \left(\frac{1}{K}\right)^{n_j} \frac{1}{n_j!}.
\end{aligned} \tag{7.10}$$

Which is proportional to a **Negative Multinomial** $\mathrm{NM}(x_0, \boldsymbol{p})$ defined by:

$$\mathrm{NM}(\boldsymbol{x}|x_0, \boldsymbol{p}) = \Gamma\left(\sum_{j=0}^K x_j\right) \frac{p_0^{x_0}}{\Gamma(x_0)} \prod_{j=1}^K \frac{p_j^{x_j}}{x_j!}$$

with parameters $x_0 = 1$, $\boldsymbol{p} = \left\{\frac{\sigma(-f_j)}{K}\right\}_{j=1}^K$, and where $p_0 = 1 - \sum_{j=1}^K p_j$. Note that the normalization term p_0 is missing in Equation (7.10). However, we do not add it, as it would render the likelihood unusable. We keep the prior unnormalized, but this does not influence the inference, as in Chapter 4, since all full conditionals are available in closed-form and normalized.

These derivations could have been avoided by noticing that the MGF of a negative binomial distribution is given by:

$$\mathrm{MGF}_{\mathrm{NM}(x_0, \boldsymbol{p})}(\boldsymbol{t}) = \left(\frac{p_0}{1 - \sum_{j=1}^K p_j e^{t_j}}\right)^{x_0}.$$

Both the Gibbs sampling and CAVI updates based on this marginalization are described in Algorithms 4 and 5.

7.4.2 A new model for the multi-class classification

In Chapter 4, two concerns can be raised. First, the parametrization of a categorical distribution with K categories requires only $K-1$ independent parameters $\boldsymbol{p}$ due to the constraint $\sum_{j=1}^K p_j = 1$. However, in the original model, which we will call **over-parametrized**, we consider K independent parameters. Second, the augmented variable λ has the improper prior $p(\lambda) = 1_{[0,\infty)}$, which is a proper measure but is not normalizable. It is not an important concern since **the posterior is normalizable**

despite the improper prior. Nevertheless, one might argue that improper priors should be avoided, as it does not allow model comparison.

On a side note, the fact that augmentations with improper priors still lead to valid inference is a good indication that scale mixtures for augmentation can be extended to non-normalizable measures.

These two issues seem connected, but we do not have any proof for it.

We propose an alternative parametrization with $K-1$ latent $\mathcal{GP}$**s**. The likelihood stays the same but with one latent being fixed:

$$p(y=k|\{f_j\}_{j=1}^{K-1}) = \begin{cases} \frac{\sigma(f_k)}{D+\sum_{j=1}^{K-1}\sigma(f_j)}, & \text{if } 1 \leq k < K-1 \\ \frac{D}{D+\sum_{j=1}^{K-1}\sigma(f_j)}, & \text{if } k = K-1 \end{cases}, \tag{7.11}$$

where $D = \sigma(f_K) \in [0,1]$. We call this version of the likelihood **bijective** since the dimensionality of the simplex output is the same as the inputs.

This likelihood comes with different properties. Unlike the softmax link, the logistic-softmax link is not translation invariant[1]. We can not freely exchange classes, and the "fixed" class has a different behavior than the rest. For example, since we fix D, the probability for classes other than K will be upper bounded by $\frac{1}{D+1}$. For example, taking $D = 0.5$ ($f_K = 0$) leads to a maximum probability of 1 for the class K and 2/3 for all other classes. On the other hand, if $D = 0$, the probability of the class K will always be 0. The bijective likelihood can still be practical if we do not care about one of the classes. Additionally, the scaled model presented in the next Section 7.4.3 can also help with the imbalance between classes.

Starting from the likelihood in Equation 7.11 the first augmentation that led to an improper prior in the over-parametrized model of Chapter 4:

$$\frac{1}{\sum_{j=1}^{K}\sigma(f_j)} = \int_0^\infty e^{-\lambda\sum_{j=1}^{K}\sigma(f_j)}d\lambda$$

is replaced by the known MGF of a Gamma distribution with the following mixture:

$$\begin{aligned}\frac{1}{D+\sum_{j=1}^{K-1}\sigma(f_j)} =& \frac{1}{D+\sum_{j=1}^{K-1}\sigma(f_j)} = \frac{1}{D}\frac{1}{1+\frac{1}{D}\sum_{j=1}^{K=1}\sigma(f_j)} \\ =& \frac{1}{D}\int_0^\infty e^{-\lambda\sum_{j=1}^{K-1}\sigma(f_j)}\text{Ga}\left(\lambda|1,\frac{1}{D}\right)d\lambda,\end{aligned}$$

which is true for $D > 0$.

The next augmentations steps are the same for the bijective and over-parametrized models: We use the MGF of the Poisson distribution and finally the Pólya-Gamma augmentation. We show the whole derivations on Algorithms 4 and 5 and show an example on Figure 7.4. We show 1-dimensional examples with 3 classes with and without the bijection on Figure 7.4 and 7.5

Algorithm 4 Gibbs sampling updates: $K/K-1$ latent $\mathcal{GP}$**s** for K classes

input: $\boldsymbol{F} = \{\boldsymbol{f}_k\}_{k=1}^K$, $p(\boldsymbol{F}) = \prod_{k=1}^{K/K-1} p(\boldsymbol{f}_k|\boldsymbol{\mu}_0, K_{\boldsymbol{X}})$, $Y = \{\boldsymbol{y}^i\}_{i=1}^N$ (one-hot encoded)
for t in 1: # samples **do**
  Draw $\boldsymbol{n}^i \sim p(\boldsymbol{n}^i|\boldsymbol{F}) = \text{NM}(1, \boldsymbol{p}^i)$ where $p_k^i = \frac{\sigma(-f_k^i)}{K} / \frac{\sigma(-f_k^i)}{D+K-1}$
  Draw $\omega_k^i \sim p(\omega_k^i|f_k^i, n_k^i, y_k^i) = \text{PG}(y_k^i + n_k^i, |f_k^i|)$
  Draw $\boldsymbol{f}_k \sim p(\boldsymbol{f}_k|\boldsymbol{\omega}_k, \boldsymbol{n}_k, \boldsymbol{Y}) = \mathcal{N}(\boldsymbol{m}_k, \boldsymbol{S}_k)$
  where $\boldsymbol{S}_k = \left(K_{\boldsymbol{X}}^{-1} + \text{diag}(\boldsymbol{\omega}_k)\right)^{-1}$ and $\boldsymbol{m}_k = \boldsymbol{S}_k\left(K_{\boldsymbol{X}}^{-1}\boldsymbol{\mu}_0 + \frac{\boldsymbol{y}_k - \boldsymbol{n}_k}{2}\right)$
end for

[1] There is no function $f(\Delta)$ such that $\sigma(x+\Delta) = f(\Delta)\sigma(x)$ for all x.

Algorithm 5 CAVI updates: $K/K-1$ latent $\mathcal{GP}$**s** for K classes

input: $q(\boldsymbol{F}) = \prod_{k=1}^{K/K-1} q(\boldsymbol{f}_k|\boldsymbol{\mu}_k, \boldsymbol{\Sigma}_k)$, $p(\boldsymbol{F} = \prod_{k=1}^{K/K-1} p(\boldsymbol{f}_k|\boldsymbol{\mu}_0, K)$, $Y = \{\boldsymbol{y}^i\}_{i=1}^N$ (one-hot encoded)

while convergence criteria is not met **do**

$$c_k^i = \sqrt{(\mu_k^i)^2 + \Sigma_k^{ii}}$$
$$p_k^i = \frac{\widetilde{\sigma}(q(f_k^i))}{K} \Big/ \frac{\widetilde{\sigma}(q(f_k^i))}{D+K-1}$$
$$\boldsymbol{\gamma}^i = \mathbb{E}_{q(\boldsymbol{n}^i)}\left[\boldsymbol{n}^i\right] = \frac{\boldsymbol{p}^i}{1-\sum_{i=1}^K p_k^i}$$
$$\theta_k^i = \mathbb{E}_{q(\omega_k^i)}\left[\omega_k^i\right] = \frac{y_k^i + \gamma_k^i}{2c_k^i} \tanh\left(\frac{c_k^i}{2}\right)$$
$$\boldsymbol{\Sigma}_k = \left(K_{\boldsymbol{X}}^{-1} + \operatorname{diag}(\boldsymbol{\theta}_k)\right)^{-1}$$
$$\boldsymbol{\mu}_k = \boldsymbol{\Sigma}_k \left(K_{\boldsymbol{X}}^{-1}\boldsymbol{\mu}_0 + \frac{\boldsymbol{y}_k - \boldsymbol{\gamma}_k}{2}\right)$$

end while

where $q(\boldsymbol{N}, \boldsymbol{\Omega}) = \prod_{i=1}^N \mathrm{PG}(\boldsymbol{\omega}^i|\boldsymbol{y}^i + \boldsymbol{n}^i, \boldsymbol{c}^i)\mathrm{NM}(\boldsymbol{n}^i|1, \boldsymbol{p}^i)$ and $\widetilde{\sigma}(q(f_k^i)) = \frac{e^{-\mu_k^i/2}}{\sqrt{(\mu_k^i)^2 + \Sigma_k^{ii}/2}}$ is an approximation to the $\sigma(-f_k^i)$.

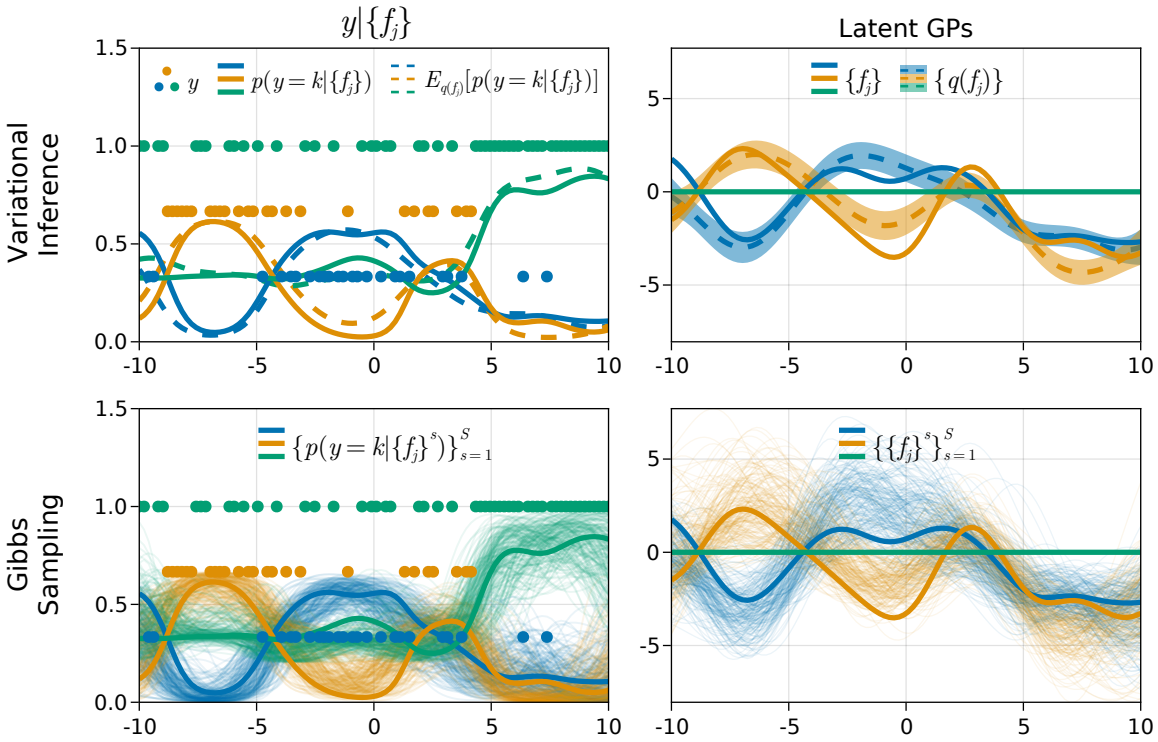


Figure 7.4: Illustration of Algorithms 4 and 5 with the bijective link introduced in Section 7.4.2 and the marginalization of Section 7.4.1. Each color represents a class, and we compare the true process to the inferred one for both Gibbs sampling and variational inference. The solid lines represent the true probabilities and latent $\mathcal{GP}$**s**. The plots on top show the variational inference results, with the expected predictive probability on the left and the variational posterior on the right. The plots at the bottom show the probabilities and latent $\mathcal{GP}$**s** obtained via Gibbs sampling.

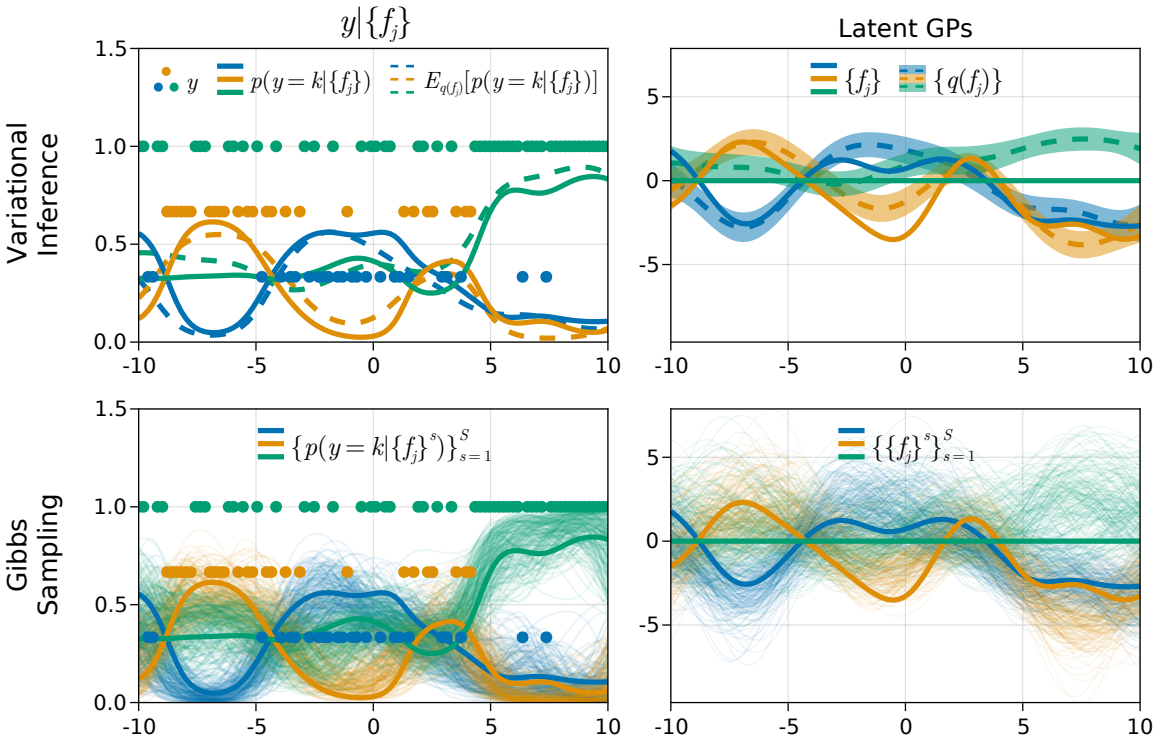


Figure 7.5: Illustration of Algorithms 4 and 5 with the overparametrized link with the marginalization of Section 7.4.1. Each color represents a class, and we compare the true process to the inferred one for both Gibbs sampling and variational inference. The solid lines represent the true probabilities and latent $\mathcal{GP}$**s**. The plots on top show the variational inference results, with the expected predictive probability on the left and the variational posterior on the right. The plots at the bottom show the probabilities and latent $\mathcal{GP}$**s** obtained via Gibbs sampling.

Both the bijective and over-parametrized links fit correctly this one-dimensional example. The over-parametrized link in Figure 7.5 do not approximate correctly the fixed latent $f_K = 0$ but still returns good predictive distributions.

When repeatedly running these examples, we observe that the predictive probabilities for the bijective link are consistently more accurate, but the predictive log-likelihood for the correct class is higher on the over-parametrized link. To confirm this trend, we would need further experiments on real datasets and with a higher number of classes.

7.4.3 Scaling the logistic-softmax link

The logistic-softmax link has issues with the predictive probabilities, in particular with many classes. Because of the boundedness of the logistic function, the logistic-softmax link needs large values of f_i to reach prediction probabilities close to 1. Even when the model should be very confident about a prediction and the latent $\mathcal{GP}$**s** are correctly inferred, the predictive probability for the correct class will be around $(1-\epsilon)/((K-1)\epsilon + 1 - \epsilon)$ where ϵ is the minimum value taken by $\sigma(f)$. With a $\mathcal{GP}$ prior centered at 0 and a reasonable kernel variance, f can not take large values. For example, taking 10 classes, if we assume $f_y = 4$ for the correct class and -4 for the others, $\epsilon \approx 0.018$, which gives a probability of 0.858 with the logistic-softmax link against 0.996 for the softmax link.

This can be solved by using a scaled logistic function. We add K hyperparameters $\boldsymbol{\theta} = \{\theta_i\}_{i=1}^K$ such that the likelihood becomes

$$p(y = k|\{f_j\}_{j=1}^K, \boldsymbol{\theta}) = \frac{\theta_k \sigma(f_k)}{\sum_{j=1}^K \theta_j \sigma(f_j)}.$$

The $\boldsymbol{\theta}$ parameters can be optimized using the ELBO with the other hyperparameters. These can also provide information about each class, a high θ_j meaning that the j-th class has zones of very high confidence. With the likelihood augmented with the variable λ, the collapsed-conditional and the maximum-likelihood optimum of $\boldsymbol{\theta}$ is available in closed-form. The maximum-likelihood optimizer is given by:

$$\theta_k^* = \frac{\sum_{i=1}^N \delta(y^i, k)}{\sum_{i=1}^N \mathbb{E}_{q(\lambda^n)}\left[\lambda^n\right]\left(1 - \widetilde{\sigma}(q(f_k^i))\right)},$$

where $\delta(x, y)$ is the Kronecker delta function, equal to 1 if $x = y$ and 0 otherwise and where $\widetilde{\sigma}(q(f_k^i))$ is defined as in Algorithm 5. We used the model definition where λ is not marginalized out.

By putting a prior $\mathrm{Ga}(\theta_k|\alpha, \beta)$, the collapsed conditional of each θ_k is given by:

$$p(\theta_k|\boldsymbol{f}_k, \boldsymbol{\lambda}) = \mathrm{Ga}(\theta_k|\alpha + \sum_{i=1}^N \delta(y^i, k), \beta + \sum_{i=1}^N \lambda^i \sigma(f_k^i))$$

A Julia implementation as well as detailed derivations can be found in the `AugmentedGPLikelihoods.jl` package [15].

7.5 Sampling from a sparse augmented model

Another work in progress regards the sampling of sparse $\mathcal{GP}$**s** models. Sampling from the augmented model proves to be very effective (see Chapter 5) while still producing samples from the posterior $p(\boldsymbol{f}|\boldsymbol{y})$ of the original model. Unfortunately, this property does not transfer when using sparse $\mathcal{GP}$**s** (for a reminder on sparse $\mathcal{GP}$**s**, see Section 2.2.3) and the scalability is limited. Simply adding inducing points locations Z with realizations $\boldsymbol{u} = f(Z)$ leads to a Gibbs sampling algorithm with a computational complexity of $\mathcal{O}((N + M)^3)$ per step and does not help with scalability. To solve this problem, we propose to mix the Gibbs sampling approach we presented in Chapter 5 with variational inference.

We build on the work of Hensman et al. [22]. They make the Titsias' assumption [53], i.e. setting the variational distribution as $q(\boldsymbol{u}, \boldsymbol{f}) = q(\boldsymbol{u})p(\boldsymbol{f}|\boldsymbol{u})$. Since they also assume a fully factorizable likelihood $p(\boldsymbol{y}|\boldsymbol{f}) = \prod_i p(y_i|f_i)$, only marginals $q(f_i)$ are required and the computational complexity of the bound decreases to $\mathcal{O}(NM^2 + M^3)$. Hensman et al. [22] show the optimal variational distribution of the inducing variables $\boldsymbol{u}$ minimizing $\mathrm{KL}\left(q(\boldsymbol{u}, \boldsymbol{f})||p(\boldsymbol{u})p(\boldsymbol{f}|\boldsymbol{u})p(\boldsymbol{y}|\boldsymbol{f})\right)$ for a factorizable likelihood $p(\boldsymbol{y}|\boldsymbol{f}) = \prod_i p(y_i|f_i)$ is given by:

$$\log q^*(\boldsymbol{u}) = \sum_i \mathbb{E}_{p(f_i|\boldsymbol{u})}\left[\log p(y_i|f_i)\right] + \log p(\boldsymbol{u}) + C, \tag{7.12}$$

where C is an intractable constant. $q^*(\boldsymbol{u})$ does not have a specific form in the general case, but we can sample from it by using HMC and evaluating the integrals $\mathbb{E}_{p(f_i|\boldsymbol{u})}\left[\log p(y_i|f_i)\right]$ numerically[2] as in [22].

We propose instead to derive a variational Gibbs sampling algorithm to draw samples from the variational distribution minimizing the Renyi divergence [57] defined as

$$D_\alpha(p, q) = \frac{1}{\alpha(\alpha - 1)} \log \int \alpha p(x) + (1 - \alpha)q(x) - p^\alpha(x)q^{1-\alpha}(x)dx, \quad \alpha \in \mathbb{R}^+. \tag{7.13}$$

The Renyi divergence converges to the forward KL divergence: $\mathrm{KL}\left(p||q\right)$ for $\alpha = 1$ and the reverse KL divergence: $\mathrm{KL}\left(q||p\right)$ for $\alpha = 0$ [57]. We define our variational distribution as $q(\boldsymbol{u}, \boldsymbol{f}, \boldsymbol{\Omega}) = q(\boldsymbol{u}, \boldsymbol{\Omega}) \prod_i p(f_i|\boldsymbol{u})$, and aim at minimizing $D_\alpha(p(\boldsymbol{u}, \boldsymbol{f}, \boldsymbol{\Omega}|\boldsymbol{y}), q(\boldsymbol{u}, \boldsymbol{f}, \boldsymbol{\Omega}))$. Note that we do not assume any independence between $\boldsymbol{u}$ and $\boldsymbol{\Omega}$, only that every f_i is conditionally independent given $\boldsymbol{u}$. There

[2] With quadrature for low-dimensions

is no parametric closed-form for the optimal distribution $q^*(\boldsymbol{u}, \boldsymbol{\Omega})$ minimizing the divergence in Equation (7.13), hence we take the approach of Hensman et al. [22] and sample from it instead. We draw $\boldsymbol{u}$, $\boldsymbol{f}$ and $\boldsymbol{\Omega}$ with a blocked Gibbs sampler, by sampling from the optimal variational distribution minimizing the conditional Renyi divergences:

$$\boldsymbol{\Omega}^i \sim q^*(\boldsymbol{\Omega}) = \arg_q \min D_\alpha \left(p(\boldsymbol{\Omega}|\boldsymbol{u}^{i-1}, \boldsymbol{f}^{i-1}, \boldsymbol{y}), q(\boldsymbol{\Omega})\right) \tag{7.14}$$

$$\begin{aligned}\boldsymbol{u}^i, \boldsymbol{f}^i \sim q^*(\boldsymbol{u}, \boldsymbol{f}) &= \arg_q \min D_\alpha \left(p(\boldsymbol{u}, \boldsymbol{f}|\boldsymbol{\Omega}^i, \boldsymbol{y}), q(\boldsymbol{u}, \boldsymbol{f})\right) \\ &= \arg_q \min D_\alpha \left(p(\boldsymbol{u})p(\boldsymbol{f}|\boldsymbol{u})p(\boldsymbol{f}|\boldsymbol{\Omega}^i, \boldsymbol{y}), q(\boldsymbol{u}) \prod_i p(f_i|\boldsymbol{u})\right).\end{aligned} \tag{7.15}$$

For all α, the minimizer for $q^*(\boldsymbol{\Omega})$ is $p(\boldsymbol{\Omega}|\boldsymbol{u}^{i-1}, \boldsymbol{f}^{i-1}, \boldsymbol{y})$, setting the conditional divergence to 0. With the approach from Chapter 5, we know $p(\boldsymbol{\Omega}|\boldsymbol{u}, \boldsymbol{f}, \boldsymbol{y})$ (which can be simplified to $p(\boldsymbol{\Omega}|\boldsymbol{f}, \boldsymbol{y})$) in closed-form and can sample from it with linear complexity with respect to the number of data points.

Bui et al. [8] solved the optimization problem of Equation (7.15) for Gaussian likelihoods, with the **Power-EP** algorithm. Since $p(\boldsymbol{f}|\boldsymbol{\Omega}, \boldsymbol{y})$ is conjugate in $\boldsymbol{f}$, the optimal $q^*(\boldsymbol{u})$ is a multivariate normal distribution with the mean and variance known in closed-form for all $\alpha \in \mathbb{R}^+$. Each sampling step for $\boldsymbol{u}$ and $\boldsymbol{f}$ only has complexity $\mathcal{O}(M^3 + M^2 N)$. Like in the Power-EP setting, $\alpha = 0$ corresponds to solving the variational approach of Titsias [53], while $\alpha = 1$ corresponds to solve the Fully Independent Training Conditional (FITC) approach of Snelson and Ghahramani [51], as shown in Bui et al. [8].

The only parameters left are the hyperparameters $\boldsymbol{\theta}$, omitted in the previous equations, that can represent a real challenge. For $\alpha = 0$, we could sample from $q^*(\boldsymbol{\theta})$ with the HMC algorithm in a separate Gibbs sampling step. For other α, we could optimize $q(\boldsymbol{\theta})$ with variational inference methods [33, 24], and hot-start with the previous distribution. The complete variational Gibbs sampler is described in Algorithm 6.

Algorithm 6 Variational Gibbs Sampler for Sparse $\mathcal{GP}$s

input: $\boldsymbol{y}$, $\boldsymbol{u}^0 \sim p(u)$, $\boldsymbol{f}^0 \sim p(\boldsymbol{f}|\boldsymbol{u}^0)$, $\boldsymbol{\theta}^0 \sim p(\boldsymbol{\theta})$
for t in 1: # samples **do**
  Draw $\boldsymbol{\Omega}^i \sim p(\boldsymbol{\Omega}|\boldsymbol{f}^{i-1}, \boldsymbol{\theta}^{i-1}\boldsymbol{y})$ *(in closed form)*
  Draw $\boldsymbol{u}^i, \boldsymbol{f}^i \sim q^*(\boldsymbol{u}, \boldsymbol{f}) = \arg_q \min D_\alpha \left(p(\boldsymbol{u}, \boldsymbol{f}|\boldsymbol{\Omega}^i, \boldsymbol{\theta}^{i-1}, \boldsymbol{y}), q(\boldsymbol{u}, \boldsymbol{f})\right)$ *(in closed form)*
  Draw $\boldsymbol{\theta}^i \sim q^*(\boldsymbol{\theta}^i) = \arg\min D_\alpha \left(p(\boldsymbol{\theta}|\boldsymbol{u}^i, \boldsymbol{f}^i, \boldsymbol{\Omega}^i, \boldsymbol{y}), q(\boldsymbol{\theta})\right)$ *(HMC or optimization)*
end for

Our approach completely gets rid of expectation computations for $\boldsymbol{u}$. It opens up more possibilities over more complex likelihoods like the multi-class or heteroscedastic ones where computing expectations numerically, like in Equation (7.12), is a limitation. For medium-sized datasets, this outperforms the CAVI algorithm as it has the same convergence speed but does not suffer from the mean-field assumption on the variational parameters. We show preliminary results on Figure 7.6 for a binary classification problem on the Magic Telescope dataset (10 dimensions, 19020 data points) [5]. The experiment is run with a 10-fold cross-validation, we use $M = 50$ inducing points selected via the k-means++ algorithm [2], and we keep the hyperparameters fixed. We compare our approach (**VI-Gibbs**) with $\alpha = 0$ against the HMC[3] variational sampling method of Hensman et al. [22] mentioned earlier (**VI-HMC**), a standard VI method optimized with an L-BFGS optimizer (**Std. VI**) and the augmented VI approach from Chapter 3 with CAVI updates (**Aug. VI**). We show the classification error and test negative log-likelihood over time on Figure 7.6.

[3]HMC is run with a fixed step-size of 0.1 and with 10 leapfrog steps.

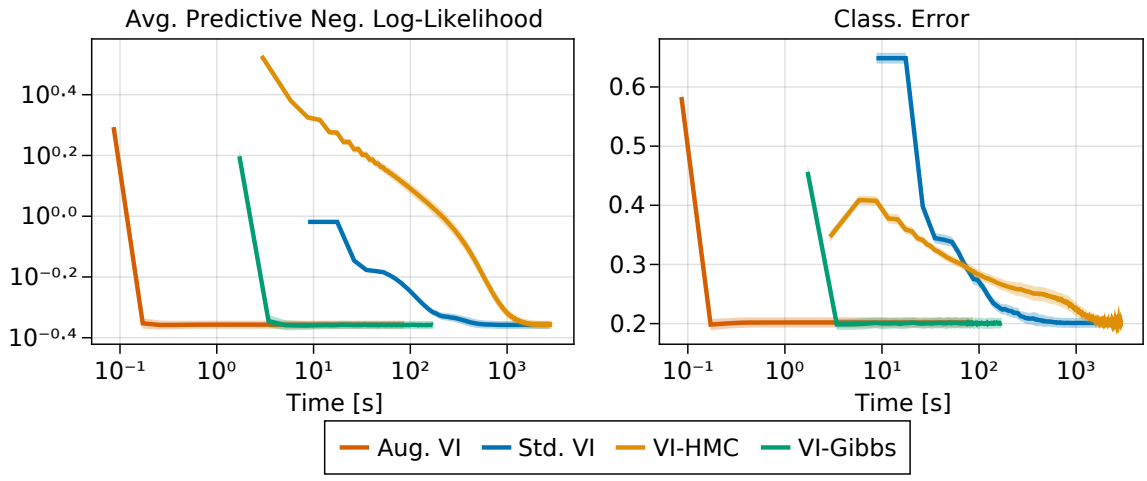


Figure 7.6: Negative test log-likelihood and classification test error over time on the Magic Telescope dataset. The mean with one standard deviation over 10 runs is shown for each algorithm.

These are first results, and there is still work on optimizing the implementation, but some first impressions can already be drawn. In terms of iterations, VI-Gibbs is just as fast as the CAVI updates but seem to have a slightly better optima. It also completely outperforms methods applied on the original model.

These preliminary graphs look very promising, but adding hyperparameter sampling might slow down the process. We also need to compare results with different likelihoods and different αs.

7.6 Limitations

Unfortunately, augmentations are not a silver bullet for approximate Bayesian inference.

Augmentable functions

The largest issue is naturally the limited domain of application. Only a constrained set of functions can be augmented. The idea of generalization using MGF as mentioned in Section 7.1 is promising but limited nonetheless. When they exist, the identification of augmentable functions in a given model can be tedious and may require lengthy derivations. We often need to rearrange terms and use mathematical identities before applying procedures like the ones described in this thesis. It is accessible to someone with expertise, but automatizing this derivation process is complicated. Current progress in symbolic programming could eventually help in this direction. We could automate this process by having a lookup table of augmentable functions and manipulating terms symbolically.

Mean-field approximation in VI Another issue is the variational distribution $q(\boldsymbol{f}, \boldsymbol{\Omega})$ (or $q(\boldsymbol{u}, \boldsymbol{\Omega})$) approximating the posterior $p(\boldsymbol{f}, \boldsymbol{\Omega}|\boldsymbol{y})$ of the augmented model is not as accurate as the variational distribution $q(\boldsymbol{f})$ (or $q(\boldsymbol{u})$) approximating the posterior $p(\boldsymbol{f}|\boldsymbol{y})$ of the original model (see Section 2.3.2). Although the original model can be recovered from the augmented model by marginalizing out the augmented variables $\boldsymbol{\Omega}$, the MF approximation loses information (correlation between $\boldsymbol{\Omega}$ and $\boldsymbol{f}$) and breaks this link. Marginalizing out $\boldsymbol{\Omega}$ in $q^*(\boldsymbol{f}, \boldsymbol{\Omega})$ will not return the optimal $q^*(\boldsymbol{f})$ trained on the original model. Interestingly, the bound difference comes exclusively from the mean-field assumption between $q(\boldsymbol{f})$ and $q(\boldsymbol{\Omega})$. We can even identify these bound differences via the interpretation of Jaakkola and Jordan [26] as missing terms from a Taylor series, as shown in Chapter 3. When analyzing the quality of the predictive distributions, the variational distribution trained on the augmented model proves to be almost as good as the variational distribution trained on the original model. The difference

of bounds mentioned earlier is often not significant at convergence but will create a difference nonetheless. These empirical results give us an indication that $\boldsymbol{f}$ and $\boldsymbol{\Omega}$ are naturally strongly decorrelated, which would explain why the Gibbs sampling and CAVI updates are so efficient.

8

Conclusion

With this thesis, I want to motivate the use of different representations to ease inference in probabilistic models. The work on scale mixtures exploits the best out of the blocked Gibbs sampling and the blocked CAVI algorithms. Deriving these augmentations can be complicated and require a certain expertise. Finding more generalizations and rules will simplify and make this approach more accessible.

We do not have a clear theoretical understanding of the reason for the fast convergence of these algorithms. By exploring the properties of these likelihoods, we work on obtaining bounds on the convergence speed of these algorithms. An intuition on why these augmentations work so well is the notion of decoupling. Many inference bottlenecks come from very highly-correlated variables and heavy tails of distributions [3]. By separating these components into different variables, all parts become easier to model and do not suffer from the typical inference issues mentioned beforehand. These ideas do not represent an actual theory for now, and we need a thorough analysis. A better understanding could give insights into how convergence speed and variable correlations are connected.

Another challenge, as pointed out in Chapter 7, is to widen the class of functions representable as mixtures. The most promising lead are Moment Generating Function (MGF), but there is little theory on their properties. Schwartz [50] is one of the few persons who developed a theory on distributions and their Laplace transforms, but, to our knowledge, the relevant pieces are missing.

Regardless, one of the biggest challenges is to popularize the use of such models. The gradient descent approach for VI of Hensman and Matthews [21] is by far the most popular, partly due to the success of the `GPFlow` library [36]. Implementing these augmentations in popular libraries would be a good step. There has been an effort in the Julia programming language [4] with the `AugmentedGPLikelihoods.jl` [15], but implementations in `GPyTorch` [17] or `GPFlow` would help the adoption of these techniques.

References

[1] Amari, S. I. (1998). Natural Gradient Works Efficiently in Learning. *Neural Computation*, 10(2):251–276. ZSCC: 0002989 ISBN: 0899-7667.

[2] Arthur, D. and Vassilvitskii, S. (2007). k-means++: The advantages of careful seeding. In *Proceedings of the eighteenth annual ACM-SIAM symposium on Discrete algorithms*, pages 1027–1035. Society for Industrial and Applied Mathematics. ZSCC: NoCitationData[s0].

[3] Betancourt, M. (2017). A conceptual introduction to Hamiltonian Monte Carlo. *arXiv preprint arXiv:1701.02434*. ZSCC: 0000306.

[4] Bezanson, J., Edelman, A., Karpinski, S., and Shah, V. B. (2017). Julia: A fresh approach to numerical computing. *SIAM Review*, 59(1):65–98.

[5] Bock, R., Chilingarian, A., Gaug, M., Hakl, F., Hengstebeck, T., Jiřina, M., Klaschka, J., Kotrč, E., Savickỳ, P., Towers, S., et al. (2004). Methods for multidimensional event classification: a case study using images from a cherenkov gamma-ray telescope. *Nuclear Instruments and Methods in Physics Research Section A: Accelerators, Spectrometers, Detectors and Associated Equipment*, 516(2-3):511–528.

[6] Brooks, S., Gelman, A., Jones, G., and Meng, X.-L. (2011). *Handbook of markov chain monte carlo*. CRC press.

[7] Bui, T. D., Yan, J., and Turner, R. E. (2017a). A unifying framework for gaussian process pseudo-point approximations using power expectation propagation. *The Journal of Machine Learning Research*, 18(1):3649–3720.

[8] Bui, T. D., Yan, J., and Turner, R. E. (2017b). A Unifying Framework for Gaussian Process Pseudo-Point Approximations using Power Expectation Propagation. *arXiv:1605.07066 [cs, stat]*. ZSCC: 0000072 arXiv: 1605.07066.

[9] Cressie, N. (1990). The origins of kriging. *Mathematical geology*, 22(3):239–252.

[10] Csató, L. (2002). *Gaussian processes: iterative sparse approximations*. PhD thesis.

[11] Csató, L. and Opper, M. (2002). Sparse on-line Gaussian processes. *Neural computation*, 14(3):641–668. ZSCC: 0000751 Publisher: MIT Press.

[12] Donner, C. and Opper, M. (2018). Efficient bayesian inference for a gaussian process density model. *arXiv preprint arXiv:1805.11494*.

[13] Duane, S., Kennedy, A. D., Pendleton, B. J., and Roweth, D. (1987). Hybrid monte carlo. *Physics letters B*, 195(2):216–222.

[14] Galy-Fajou, T. (2021). theogf/AugmentedGaussianProcesses.jl.

[15] Galy-Fajou, T. (2022). JuliaGaussianProcesses/AugmentedGPLikelihoods.jl: v0.4.9.

[16] Galy-Fajou, T., Widmann, D., Yalburgi, S., willtebbutt, st, Falk, I., Ridderbusch, S., Wright, T., david vicente, Khan, S., Ge, H., Giersdorf, J., TagBot, J., Mones, L., Monticone, P., Viljoen, R., Schölly, S., and Öcal, K. (2022). JuliaGaussianProcesses/KernelFunctions.jl.

[17] Gardner, J., Pleiss, G., Weinberger, K. Q., Bindel, D., and Wilson, A. G. (2018). Gpytorch: Blackbox matrix-matrix gaussian process inference with gpu acceleration. *Advances in neural information processing systems*, 31.

[18] Gorinova, M., Moore, D., and Hoffman, M. (2020). Automatic Reparameterisation of Probabilistic Programs. In *International Conference on Machine Learning*, pages 3648–3657. PMLR. ZSCC: 0000004 ISSN: 2640-3498.

[19] Harrison Jr, D. and Rubinfeld, D. L. (1978). Hedonic housing prices and the demand for clean air. *Journal of environmental economics and management*, 5(1):81–102. ZSCC: 0001726 Publisher: Elsevier.

[20] Henao, R., Yuan, X., and Carin, L. (2014). Bayesian Nonlinear Support Vector Machines and Discriminative Factor Modeling. *Nips*, (Mcmc):1–9. ZSCC: 0000028.

[21] Hensman, J. and Matthews, A. (2015). Scalable Variational Gaussian Process Classification. *Aistats*, 38:1–9. ZSCC: 0000200 arXiv: 1411.2005.

[22] Hensman, J., Matthews, A. G. d. G., Filippone, M., and Ghahramani, Z. (2015). MCMC for Variationally Sparse Gaussian Processes. *arXiv:1506.04000 [stat]*. ZSCC: 0000090 arXiv: 1506.04000.

[23] Hensman, J., Sheffield, U., Fusi, N., and Lawrence, N. (2013). Gaussian Processes for Big Data. *Proceedings of UAI 29*, pages 282–290. ZSCC: NoCitationData[s1] arXiv: 1309.6835 ISBN: 978-1-4503-1285-1.

[24] Hernandez-Lobato, J., Li, Y., Rowland, M., Bui, T., Hernández-Lobato, D., and Turner, R. (2016). Black-box alpha divergence minimization. In *International Conference on Machine Learning*, pages 1511–1520. PMLR.

[25] Hoffman, M. D. and Gelman, A. (2014). The No-U-Turn sampler: adaptively setting path lengths in Hamiltonian Monte Carlo. *Journal of Machine Learning Research*, 15(1):1593–1623. ZSCC: 0001680.

[26] Jaakkola, T. S. and Jordan, M. I. (1997). A Variational Approach to Bayesian Logistic Regression Models and their Extensions. In *Sixth International Workshop on Artificial Intelligence and Statistics*, pages 283–294. PMLR. ZSCC: 0000268 ISSN: 2640-3498.

[27] Jaakkola, T. S. and Jordan, M. I. (2000). Bayesian parameter estimation via variational methods. *Statistics and Computing*, 10(1):25–37. ZSCC: 0000581.

[28] Jensen, C. S., Kjærulff, U., and Kong, A. (1995). Blocking gibbs sampling in very large probabilistic expert systems. *International Journal of Human-Computer Studies*, 42(6):647–666.

[29] Jordan, M. I. and Mitchell, T. M. (2015). Machine learning: Trends, perspectives, and prospects. *Science*, 349(6245):255–260.

[30] Kulesza, A. and Taskar, B. (2012). Determinantal point processes for machine learning. pages 1–120. ZSCC: 0000516 arXiv: 1207.6083 ISBN: 9781601986283.

[31] Lázaro-Gredilla, M. and Figueiras-Vidal, A. (2009). Inter-domain gaussian processes for sparse inference using inducing features. In Bengio, Y., Schuurmans, D., Lafferty, J., Williams, C., and Culotta, A., editors, *Advances in Neural Information Processing Systems*, volume 22. Curran Associates, Inc.

[32] Lázaro-Gredilla, M. and Titsias, M. K. (2011). Variational heteroscedastic gaussian process regression. In *ICML*.

[33] Li, Y. and Turner, R. E. (2016). Rényi divergence variational inference. *Advances in neural information processing systems*, 29.

[34] Lin, W., Schmidt, M., and Khan, M. E. (2020). Handling the Positive-Definite Constraint in the Bayesian Learning Rule. *arXiv:2002.10060 [cs, stat]*. ZSCC: 0000000 arXiv: 2002.10060.

[35] Liu, J. S. (1994). The collapsed gibbs sampler in bayesian computations with applications to a gene regulation problem. *Journal of the American Statistical Association*, 89(427):958–966.

[36] Matthews, A. G. d. G., van der Wilk, M., Nickson, T., Fujii, K., Boukouvalas, A., León-Villagrá, P., Ghahramani, Z., and Hensman, J. (2017). GPflow: A Gaussian process library using TensorFlow. *Journal of Machine Learning Research*, 18(40):1–6.

[37] Murphy, K. P. (2012). *Machine learning: a probabilistic perspective*. Adaptive computation and machine learning series. MIT Press, Cambridge, MA. ZSCC: 0007949.

[38] Murray, I., Adams, R., and MacKay, D. (2010). Elliptical slice sampling. In *Proceedings of the thirteenth international conference on artificial intelligence and statistics*, pages 541–548. JMLR Workshop and Conference Proceedings.

[39] Neal, R. M. (2003). Slice sampling. *Annals of Statistics*, 31(3):705–741. ZSCC: 0001947 arXiv: 1003.3201v1 ISBN: 00905364.

[40] Neal, R. M. et al. (2011). Mcmc using hamiltonian dynamics. *Handbook of markov chain monte carlo*, 2(11):2.

[41] Nguyen, T. M. and Wu, Q. M. (2012). Robust student's-t mixture model with spatial constraints and its application in medical image segmentation. *IEEE Transactions on Medical Imaging*, 31(1):103–116. ZSCC: NoCitationData[s0] ISBN: 0278-0062.

[42] O'Hagan, A. and Forster, J. J. (2004). *Kendall's advanced theory of statistics, volume 2B: Bayesian inference*, volume 2. Arnold.

[43] Palmer, J. A. (2006). *Variational and scale mixture representations of non-Gaussian densities for estimation in the Bayesian linear model: Sparse coding, independent component analysis, and minimum entropy segmentation*. PhD thesis, UC San Diego. ZSCC: 0000014.

[44] Polson, N. G., Scott, J. G., and Windle, J. (2012). Bayesian inference for logistic models using Polya-Gamma latent variables. pages 1–42. ZSCC: NoCitationData[s0] arXiv: 1205.0310.

[45] Quinonero-Candela, J. and Rasmussen, C. E. (2005). A unifying view of sparse approximate gaussian process regression. *The Journal of Machine Learning Research*, 6:1939–1959.

[46] Rasmussen, C. E. and Williams, C. K. I. (2018). *Gaussian Processes for Machine Learning*, volume 1. MIT press Cambridge. ZSCC: NoCitationData[s0] arXiv: 026218253X Publication Title: Gaussian Processes for Machine Learning ISSN: 0129-0657.

[47] Ridout, M. S. (2009). Generating random numbers from a distribution specified by its Laplace transform. *Statistics and Computing*, 19(4):439. ZSCC: 0000049 Publisher: Springer.

[48] Salimbeni, H., Eleftheriadis, S., and Hensman, J. (2018). Natural Gradients in Practice: Non-Conjugate Variational Inference in Gaussian Process Models. *arXiv:1803.09151 [cs, stat]*. ZSCC: 0000028 arXiv: 1803.09151.

[49] Schlaifer, R. and Raiffa, H. (1961). *Applied statistical decision theory*.

[50] Schwartz, L. (1952). Transformation de laplace des distributions. *Comm. Sém. Math. Univ. Lund [Medd. Lunds Univ. Mat. Sem.]*, 1952(Tome Supplémentaire):196–206.

[51] Snelson, E. and Ghahramani, Z. (2009). Sparse Gaussian Processes using Pseudo-inputs. *Advances in Neural Information Processing Systems 18*, pages 1–24. ZSCC: NoCitationData[s0] ISBN: 9780262232531.

[52] Solin, A., Hensman, J., and Turner, R. E. (2018). Infinite-Horizon Gaussian Processes. *arXiv:1811.06588 [cs, stat]*. ZSCC: 0000013 arXiv: 1811.06588.

[53] Titsias, M. (2009). Variational Learning of Inducing Variables in Sparse Gaussian Processes. *Aistats*, 5:567–574. ZSCC: 0000724.

[54] Titsias, M. and Lázaro-Gredilla, M. (2014). Doubly stochastic variational bayes for non-conjugate inference. In *International conference on machine learning*, pages 1971–1979. PMLR.

[55] Turner, R., Deisenroth, M., and Rasmussen, C. (2010). State-space inference and learning with gaussian processes. In Teh, Y. W. and Titterington, M., editors, *Proceedings of the Thirteenth International Conference on Artificial Intelligence and Statistics*, volume 9 of *Proceedings of Machine Learning Research*, pages 868–875, Chia Laguna Resort, Sardinia, Italy. PMLR.

[56] van der Wilk, M., Dutordoir, V., John, S., Artemev, A., Adam, V., and Hensman, J. (2020). A framework for interdomain and multioutput gaussian processes.

[57] Van Erven, T. and Harremos, P. (2014). Rényi divergence and kullback-leibler divergence. *IEEE Transactions on Information Theory*, 60(7):3797–3820.

[58] Wang, C. and Neal, R. M. (2012). Gaussian Process Regression with Heteroscedastic or Non-Gaussian Residuals. *arXiv:1212.6246 [cs, stat]*. ZSCC: 0000044 arXiv: 1212.6246.

[59] Wenzel, F., Galy-Fajou, T., Deutsch, M., and Kloft, M. (2017). Bayesian nonlinear support vector machines for big data. In *Joint European Conference on Machine Learning and Knowledge Discovery in Databases*, pages 307–322. Springer. ZSCC: 0000020.

[60] Wenzel, F., Galy-Fajou, T., Donner, C., Kloft, M., and Opper, M. (2018). Efficient Gaussian Process Classification Using Polya-Gamma Data Augmentation. *arXiv:1802.06383 [cs, stat]*. ZSCC: NoCitationData[s0] arXiv: 1802.06383.

[61] Widmann, D., willtebbutt, Galy-Fajou, T., st, Yalburgi, S., Ge, H., david vicente, Bosch, N., Schmitz, N., Viljoen, R., Wright, T., and andreaskoher (2022). JuliaGaussianProcesses/AbstractGPs.jl.

[62] Williams, C. K., Rasmussen, C. E., Scwaighofer, A., and Tresp, V. (2002). Observations on the nyström method for gaussian process prediction.

[63] Wilson, J. T., Borovitskiy, V., Terenin, A., Mostowsky, P., and Deisenroth, M. P. (2021). Pathwise conditioning of gaussian processes. *Journal of Machine Learning Research*, 22(105):1–47.

A

Additional work

The following work does not fit the storyline of the thesis and is therefore presented here only as a side project.

A.1 Adaptive Inducing Points Selection for Gaussian Processes

Two important questions raised when using the sparse $\mathcal{GP}$**s** presented in Section 2.2.3 are: How should the inducing points be located? How many points does one need to reach a desired level of accuracy? This work tries to answer these questions by proposing an adaptive algorithm, working in $\mathcal{O}(N)$ time and also valid in an online setting.

Although the algorithm proves to be more efficient than standard methods and to have interesting theoretical properties related to Determinantal Point Processes [30], it has serious tuning issues. The parameters regulating the algorithm, how often one adds a point or removes one, are tightly correlated to the kernel hyperparameters. When optimizing hyperparameters during training, an unstable behavior may lead to picking all points as inducing points or selecting none. I presented this work in the Continual Learning Workshop of ICML 2020.

Authors:
Théo Galy-Fajou[1], Manfred Opper[1]
[1]TU Berlin

Details:
Type: Workshop article
Submitted: June 2020
Accepted: July 2020
URL: `https://drive.google.com/file/d/1IPTUBfY_b2WElTWBIVU4lrbHcXnbTWdB/view`
Workshop: Continual Learning (ICML 2020)

Adaptive Inducing Points Selection for Gaussian Processes

Théo Galy-Fajou [1] **Manfred Opper** [1]

[1] Technical University of Berlin

Abstract

Gaussian Processes (**GPs**) are flexible non-parametric models with strong probabilistic interpretation. While being a standard choice for performing inference on time series, GPs have little techniques to work in a streaming setting. (Bui et al., 2017) developed an efficient variational approach to train online GPs by using sparsity techniques: The whole set of observations is approximated by a smaller set of inducing points (**IPs**) and moved around with new data. Both the number and the locations of the IPs will affect greatly the performance of the algorithm. In addition to optimizing their locations we propose to adaptively add new points, based on the properties of the GP and the structure of the data.

1. Introduction

Gaussian Processes (**GPs**) are flexible non-parametric models with strong probabilistic interpretation. They are particularly fitted for time-series (Roberts et al., 2013) but one of their biggest limitations is that they scale cubically with the number of points (Williams & Rasmussen, 2006). Quinonero-Candela & Rasmussen (2005) introduced the notion of sparse GPs, models approximating the posterior by a smaller number M of inducing points (**IPs**) and reducing the inference complexity from $\mathcal{O}(N^3)$ to $\mathcal{O}(M^3)$ where M is the number of IPs. Titsias (2009) introduced them later in a variational setting, allowing to optimize their locations. Based on this idea, (Bui et al., 2017) introduced a variational streaming model relying on inducing points. One of their algorithm's features is that hyper-parameters can be optimized and more specifically the number of inducing can vary between batches of data. However in their work, the number of IPs is fixed and their locations are simply optimized against the variational bound of the marginal likelihood. Having a fixed number of IPs limits the model's scope if the total data size is unknown. A gradient based approach leads to two problems:
- IP's locations need to be optimized until convergence for every batch. Therefore batches need to be sufficiently large to get a meaningful improvement. If the new data comes in very far from the original positions of the IPs, the optimization will be extremely slow.
- The number of IPs being fixed, there is no way to know how many will be required to have a desired accuracy. Finding the optimal number of IPs is also not an option as it is an ill-posed problem: the objective will only decrease with more IPs, i.e. the optimum is obtained when every data point is an IP.

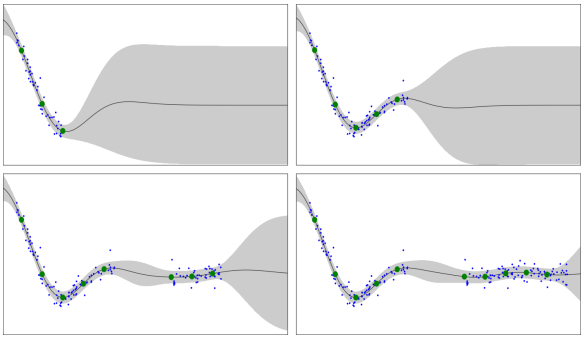


Figure 1: Illustration of the inducing point selection process. Blue points represent inducing points, green points data and the orange line represent the mean of the prediction from the GP model surrounded by one standard error. The dashed represent the space covered by the existing IPs, only points seen outside those areas are selected as new IPs.

We propose a different approach to this problem with a simple algorithm, **Online Inducing Points Selection** (**OIPS**), requiring only one parameter to select automatically both the number of inducing points and their location. OIPS naturally takes into account the structure of the data while the performance trade-off and the expected number of IPs can be inferred.

Our main contributions are as follow :
- We develop an efficient online algorithm to automatically select the number and location of inducing points for a streaming GP.
- We give theoretical guarantees on the expected number of inducing points and the performance of the GP.

In section 2 we present existing methods to select inducing

points, as well as an online inference for GPs. We present our algorithm and its theoretical guarantees in section 3. We show our experiments in comparison with popular inducing points selection methods in section 4. Finally we summarize our findings and explore outlooks in section 5.

2. Background

2.1. Sparse Variational Gaussian Processes

Gaussian Processes: Given some training data $\mathcal{D} = \{X, \boldsymbol{y}\}$ where $X = \{x_i\}_{i=1}^N$ are the inputs $x_i \in \mathbb{R}^D$ and $\boldsymbol{y} = \{y_i\}_{i=1}^N$ are the labels, we want to compute the predictive distribution $p(y^*|D, x^*)$ for new inputs x^*. In order to do this we try to find an optimal distribution over a latent function f. We set the latent vector $\boldsymbol{f}$ as the realization of $f(X)$, where $f_i = f(x_i)$, and put a GP prior $\mathcal{GP}(\mu_0, k)$ on $\boldsymbol{f}$, with μ_0 the prior mean (set to 0 without loss of generality) and k a kernel function. In this work we are going to use an isotropic squared exponential kernel (**SE kernel**) : $k(x, x') = \exp(-||x - x'||^2/l^2)$, but it is generally applicable to all translation-invariant kernels. We then compute the posterior:

$$p(\boldsymbol{f}|\mathcal{D}) = \frac{\prod_{i=1}^N p(y_i|f_i)p(\boldsymbol{f})}{p(\mathcal{D})} \tag{1}$$

Where $p(\boldsymbol{f}) \sim \mathcal{N}(0, K_{XX})$ and K_{XX} is the kernel matrix evaluated on X (in later notation we use K_X instead of K_{XX}). For a Gaussian likelihood the posterior $p(\boldsymbol{f}|\mathcal{D})$ is known analytically in closed-form. Prediction and inference have nonetheless a complexity of $\mathcal{O}(N^3)$

Sparse Variational Gaussian Processes: When the likelihood is not Gaussian, there is no tractable solution for the posterior. One possible approximation is to use variational inference : a family of distributions over $\boldsymbol{f}$ is selected, e.g. the multivariate Gaussian $q(\boldsymbol{f}) = \mathcal{N}(\boldsymbol{m}, S)$, and one optimizes the variational parameters $\boldsymbol{m}$ and S by minimizing the negative ELBO, a proxy for the KL divergence $\mathrm{KL}(q(\boldsymbol{f})||p(\boldsymbol{f}|\mathcal{D}))$. However the computational complexity still grows cubically with the number of samples, and is therefore inadequate to large datasets.

Quinonero-Candela & Rasmussen (2005) and Titsias (2009) introduced the notion of sparse variational GPs (**SVGP**). One adds inducing variables $\boldsymbol{u}$ and their inducing locations $Z = \{Z_i\}_{i=1}^M$ to the model. In this work we restrict Z_i to be in the same domain as X_i but inter-domain approaches also exist (Hensman et al., 2017). The relation between $\boldsymbol{u}$ and $\boldsymbol{f}$ is given by the distribution $p(\boldsymbol{f}, \boldsymbol{u}) = p(\boldsymbol{f}|\boldsymbol{u})p(\boldsymbol{u})$ where

$$p(\boldsymbol{f}|\boldsymbol{u}) = \mathcal{N}(\boldsymbol{f}|K_{XZ}K_Z^{-1}\boldsymbol{u}, \widetilde{K}),\ p(\boldsymbol{u}) = \mathcal{N}(0, K_Z) \tag{2}$$

where $\widetilde{K} = K_X - K_{XZ}K_Z^{-1}K_{ZX}$

Then we approximate $p(\boldsymbol{f}, \boldsymbol{u})$ with the variational distribution $q(\boldsymbol{f}, \boldsymbol{u}) = p(\boldsymbol{f}|\boldsymbol{u})q(\boldsymbol{u})$ where $q(\boldsymbol{u}) = \mathcal{N}(\boldsymbol{\mu}, \Sigma)$ by optimizing $\mathrm{KL}(q(\boldsymbol{f}, \boldsymbol{u})||p(\boldsymbol{f}, \boldsymbol{u}|\mathcal{D}))$.

Note that if the likelihood is Gaussian, the optimal variational parameters $\boldsymbol{\mu}^*$ and Σ^* are known in closed-form. The only parameters left to optimize are the kernel parameters as well as selecting the number and the location of the inducing variables.

2.2. Inducing points selection methods

Titsias (2009) initially proposed to select the points location via a greedy selection : A small batch of data is randomly sampled, each sample is successively tested by adding it to the set of inducing points and evaluating the improvement on the ELBO. The sample bringing the best performance is added to the set of inducing points and the operation is repeated until the desired number of inducing points is reached. This greedy approach has the advantage of selecting a set which is already close to the optimum set but is extremely expensive and is not applicable to non-conjugate likelihoods as it relies on estimating the optimal bound.

The most popular approach currently is to use the k-means++ algorithm (Arthur & Vassilvitskii, 2007) and take the optimized clusters centers as inducing points locations. The clustering nature of the algorithm allows to have good coverage of the whole dataset. However the k-means algorithm have a complexity of $\mathcal{O}(NMDT)$ on the whole dataset where T is the number of k-means iterations. Another issue is that it might allocate multiple centers in a region of high density leading to very close inducing points and no significant performance improvement. It is also not applicable online and does not solve the problem of choosing the number of inducing points.

Another classical approach is to simply take a grid. For example Moreno-Muñoz et al. (2019) use a grid in an online setting by updating the bounds of a uniform grid. Using a grid is unfortunately limited a small number of dimensions and does not take into account the structure of the data.

2.3. Online Variational Gaussian Process Learning

(Bui et al., 2017) developed a streaming algorithm for GPs (**SSVGP**) based the inducing points approach of (Titsias, 2009). The method consists in recursively optimizing the variational distribution $q_t(\boldsymbol{u}_t, \boldsymbol{f})$ for each new batch of data $\mathcal{D}_t$ given the previous variational distribution $q_{t-1}(\boldsymbol{u}_{t-1}, \boldsymbol{f})$. q_t initially approximates the posterior :

$$p(\boldsymbol{u}_t, \boldsymbol{f}|\mathcal{D}_{1:t}) = \frac{p(\mathcal{D}_t|\boldsymbol{f})p(\mathcal{D}_{1:(t-1)}|\boldsymbol{f})p(\boldsymbol{u}_t, \boldsymbol{f}|\theta_t)}{p(\mathcal{D}_{1:t})} \tag{3}$$

where θ_t are the set of hyper-parameters. Since $D_{1:(t-1)}$ is not accessible anymore, the likelihood on previously seen

data is approximated using the previous variational approximation $q_{t-1}(\boldsymbol{u}_{t-1})$ and the previous hyper-parameters θ_{t-1}:

$$p(\mathcal{D}_{1:(t-1)}|\boldsymbol{f}) \approx \frac{q_{t-1}(\boldsymbol{u}_{t-1})p(\mathcal{D}_{1:(t-1)})}{p(\boldsymbol{u}_{t-1}|\theta_{t-1})}.$$

The distribution approximated by q_t is in the end:

$$q_t(\boldsymbol{u}_t, \boldsymbol{f}|\mathcal{D}_{1:t}) \approx \frac{p(\mathcal{D}_t|\boldsymbol{f})q_{t-1}(\boldsymbol{u}_{t-1})p(\boldsymbol{u}_t, \boldsymbol{f}|\theta_t)}{p(\boldsymbol{u}_{t-1}|\theta_{t-1})} \frac{p(\mathcal{D}_{1:(t-1)})}{p(\mathcal{D}_{1:t})} \quad (4)$$

The optimization of the (bound on the) KL divergence between the two distributions for each new batch will preserve the information of $\mathcal{D}_{1:(t-1)}$ via q_{t-1} and ensure a smooth transition of the hyper-parameters, including the number of inducing points. We give all technical details including the hyper-parameter derivatives and the ELBO in full form in appendix A.

3. Algorithm

The idea of our algorithm is that to give a good approximation, a large majority of the samples should be "close" (in the reproducing kernel Hilbert space (RKHS)) to the set Z of IPs locations. Additionally, Z should be as diverse as possible, since IP degeneracy will not improve the approximation. This intuition is supported by previous works:
- Bauer et al. (2016) showed that the most substantial improvement obtained by adding a new inducing point was through the reduction of the uncertainty of $q(\boldsymbol{f})$, which decreases quadratically with K_{XZ}.
- Burt et al. (2019) showed that the quality of the approximation made with inducing points is bounded by the norm of $Q_X = K_X - K_{XZ}K_Z^{-1}K_{ZX}$.
Therefore by ensuring that K_{XZ} and $|K_Z|$ are sufficiently large, we can expect an improvement on the approximation of the non-sparse problem.

3.1. Adding New Inducing Points

A simple yet efficient strategy is to verify that for each new data point x seen during training, there exists a close inducing point. We first compute $K_{xZ} = [k(x, Z_1), \dots, k(x, Z_M)]$. If the maximum value of K_{xZ} is smaller than a threshold parameter ρ, the sample is added to the set of IPs Z. If not, the algorithm passes on to the next sample. We summarize all steps in Algorithm 1.

The streaming nature of the algorithm makes it perfectly suited for an online learning setting : it needs to see samples only once, whereas other algorithms like k-means need to parse all the data multiple times before converging. It is fully deterministic for a given sequence of samples and therefore convergence guarantees are given under some conditions. This approach was previously explored in a different context by Csató & Opper (2002), but was limited to small datasets.

Algorithm 1 Online Inducing Point Selection (**OIPS**)

Input: sample x, set of inducing points $Z = \{Z_j\}_{j=1}^M$, acceptance threshold $0 < \rho < 1$, kernel function k
$d \leftarrow \max_j(k(x, Z_j))$
if $d < \rho$ **then**
  $\{Z_j\} \leftarrow \{Z_j\} \bigcup x$
  $M \leftarrow M + 1$
end if
return $\{Z_j\}$

The extra cost of the algorithm is virtually free since K_{XZ} needs to be computed for the variational updates of the model.

One of our claims is that our algorithm is model and data agnostic. The reason is that as kernel hyper-parameters are optimized, the acceptance condition changes as well

Note that this method can be interpreted as a half-greedy approach of a sequential sampling of a determinantal point process (Kulesza & Taskar, 2012). In appendix B, we show that for the same number of points, the probability of our selected set is higher than the one of a k-DPP.

3.2. Theoretical guarantees

The final size of Z is depending on many factors: the selected threshold ρ, the chosen kernel, the structure of the data (distribution, sparsity, etc) and the number of points seen. However by having some weak assumptions on the data we can prove a bound on the expected number of inducing points as well as on the quality of the variational approximation.

Expected number of inducing points : Since the selection process is directly depending on the data, it is impossible to give an arbitrary bound. However by adding assumptions on the distribution of x one can

Theorem 1. *Given a dataset i.i.d and uniformly distributed, i.e.* $x \sim \mathcal{U}(0, a)^D$*, and a SE kernel with length-scale* $l^D \ll 1$*, the expected number of selected inducing points* M *after parsing* N *points is*

$$\mathbb{E}[M|N] \leq \frac{a^D - (a^D - \alpha)^{N+1}}{\alpha}, \quad (5)$$

where $\alpha = \left(\frac{l\sqrt{-D\log\rho}}{2}\right)^D$.

The proof is given in the appendix C. As $N \to \infty$, this bound will converge to a^D/α which is the estimated number of overlapping hyper-spheres of radius $l\sqrt{-D\log\rho_{in}}$ to fill a hypercube of dimension D with side length a. This can be used as an upper bound for any data lying in a compact domain. This confirms the intuition that the number

of selected inducing points will grow faster with larger dimensions and a larger ρ and with smaller lengthscales.

Expected performance on regression : Burt et al. (2019) derived a convergence bound for the inducing points approach of (Titsias, 2009). Even if they show this bound in an offline setting, their bound is still relevant for online problems. They show that when Z is sampled via a k-DPP process (Kulesza & Taskar, 2011), i.e. a determinantal point process conditioned on a fixed set size, the difference between the ELBO and the log evidence $\log p(\mathcal{D})$ is bounded by

$$\mathbb{E}_Z \left[\|K_X - Q_X\| \right] \leq (M+1) \sum_{i=M+1}^{N} \lambda_i(K_X) \quad (6)$$

where $\lambda_i(K_X)$ is the i-th largest eigenvalue of K_X and $Q_X = K_{XZ}K_Z^{-1}K_{ZX}$ is the Nyström approximation of K_X.

We derive a similar bound when using our algorithm instead of k-DPPsampling:

Theorem 2. *Let Z be the set of inducing points locations of size M selected via Algorithm 1 on the dataset X of size N.*

$$\|K_X - Q_X\| \leq (N-M)\left(1 - \frac{\rho^2}{1+M(M-1)\rho}\right) \quad (7)$$

where K_X is the kernel matrix on X and Q_X is the Nyström approximation of K_X using the subset Z

The proof and an empirical comparison are given in the appendix D.

4. Experiments

In this section we get a quick look on how our algorithm performs in different settings compared to approaches described in section 2.2. We compare the online model **SSVGP** described in section 2 with different IP selection techniques. We select from the first batch via k-means and then optimize them (**k-means/opt**), select them via our algorithm and optimize them (**OIPS/opt**), select them via our algorithm but don't optimize them (**OIPS**) and finally create a **Grid** that we adapt according to new bounds. We consider 3 different toy datasets, from which two are displayed in figure 2. The dataset A is a uniform time series and the output function is a noisy sinus. The dataset B is an irregular time-series, with a gap in the inputs. The output function is also a noisy sinus. Dataset C inputs are random samples from an isotropic multivariate 3D Gaussian and the output function is given by $\sin(||x||)/||x||$. All datasets contain 200 training points and 200 test points. For all experiments we use an isotropic SE kernel with fixed parameters. For datasets A and B, **Grid** and k**-means** has 25 IPs while **OIPS** converged to around 20 IPs. For dataset C, **Grid** has 10^3 IPs, k**-means** 50, and both **OIPS** converged to 10 IPs Figure 2 shows the evolution on the average negative log likelihood on test data after every batch has been seen. On a uniform time-series context all methods are pretty much equivalent. The presence of a gap, blocks the optimization of IP locations and impede inference of future points. Whereas the grid suffers from being in high-dimensions and All details on the datasets, different training methods, hyper-parameters and optimization parameters used are to be found in appendix E.

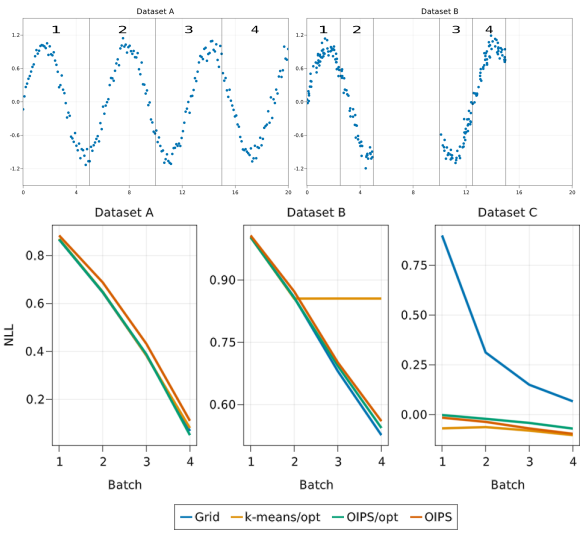


Figure 2: Toy datasets A and B, divived in 4 batches. Average Negative Test Log-Likelihood on a test set in function of number of batches seen. In a uniform streaming setting all methods perform similarly but having a gap blocks the convergence of a simple position optimization whereas in a non-compact situation the adaptive grid suffers in performance.

5. Conclusion

We presented a new algorithm, OIPS, able to select inducing points automatically for a GP in an online setting. The theoretical bounds derived outperforms the previous work based on DPPs. There is yet to improve the selection process to make it robust to outliers and to variations of the hyper-parameters. Using for instance a threshold on the median or a mean on the k-nearest IPs could help to avoid picking adversarial points such as outliers. We have only considered regression but our algorithm is also compatible with non-conjugate likelihoods. Using augmentations approaches (Wenzel et al., 2019; Galy-Fajou et al., 2019), same performance can be attained. Finally the most interesting improvement would be to use a non-stationary kernel (Remes et al., 2017) and be able to automatically adapt the number of inducing points across the dataset.

References

Arthur, D. and Vassilvitskii, S. k-means++: The advantages of careful seeding. In *Proceedings of the eighteenth annual ACM-SIAM symposium on Discrete algorithms*, pp. 1027–1035. Society for Industrial and Applied Mathematics, 2007.

Bauer, M., van der Wilk, M., and Rasmussen, C. E. Understanding probabilistic sparse gaussian process approximations. In *Advances in neural information processing systems*, pp. 1533–1541, 2016.

Belabbas, M.-A. and Wolfe, P. J. Spectral methods in machine learning and new strategies for very large datasets. *Proceedings of the National Academy of Sciences*, 106 (2):369–374, 2009.

Bui, T. D., Nguyen, C., and Turner, R. E. Streaming sparse gaussian process approximations. In *Advances in Neural Information Processing Systems*, pp. 3299–3307, 2017.

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

Csató, L. and Opper, M. Sparse on-line gaussian processes. *Neural computation*, 14(3):641–668, 2002.

Galy-Fajou, T., Wenzel, F., Donner, C., and Opper, M. Multi-class gaussian process classification made conjugate: Efficient inference via data augmentation. *arXiv preprint arXiv:1905.09670*, 2019.

Hensman, J., Durrande, N., and Solin, A. Variational fourier features for gaussian processes. *The Journal of Machine Learning Research*, 18(1):5537–5588, 2017.

Kulesza, A. and Taskar, B. k-dpps: Fixed-size determinantal point processes. In *Proceedings of the 28th International Conference on Machine Learning (ICML-11)*, pp. 1193–1200, 2011.

Kulesza, A. and Taskar, B. Determinantal point processes for machine learning. pp. 1–120, 2012. ISSN 1935-8237. doi: 10.1561/2200000044. URL http://arxiv.org/abs/1207.6083%0Ahttp://dx.doi.org/10.1561/2200000044. ZSCC: 0000516 arXiv: 1207.6083 ISBN: 9781601986283.

Moreno-Muñoz, P., Artés-Rodríguez, A., and Álvarez, M. A. Continual multi-task gaussian processes. *arXiv preprint arXiv:1911.00002*, 2019.

Quinonero-Candela, J. and Rasmussen, C. E. A Unifying View of Sparse Approximate Gaussian Process Regression. *Journal of Machine Learning Research*, 6:1939–1959, 2005. ZSCC: NoCitationData[s0].

Remes, S., Heinonen, M., and Kaski, S. Non-stationary spectral kernels. In *Advances in Neural Information Processing Systems*, pp. 4642–4651, 2017.

Roberts, S., Osborne, M., Ebden, M., Reece, S., Gibson, N., and Aigrain, S. Gaussian processes for time-series modelling. *Philosophical Transactions of the Royal Society A: Mathematical, Physical and Engineering Sciences*, 371(1984):20110550, February 2013. ISSN 1364-503X, 1471-2962. doi: 10.1098/rsta.2011.0550. URL https://royalsocietypublishing.org/doi/10.1098/rsta.2011.0550.

Stewart, G. W. and guang Sun, J. *Matrix Perturbation Theory*. Academic Press, 1990.

Titsias, M. Variational learning of inducing variables in sparse gaussian processes. In *Artificial Intelligence and Statistics*, pp. 567–574, 2009.

Wenzel, F., Galy-Fajou, T., Donner, C., Kloft, M., and Opper, M. Efficient gaussian process classification using pòlya-gamma data augmentation. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 33, pp. 5417–5424, 2019.

Williams, C. K. and Rasmussen, C. E. *Gaussian processes for machine learning*, volume 2. MIT press Cambridge, MA, 2006.

A. Derivations online GPs

A.1. ELBO

Following Bui et al. (2017), the ELBO for variational inference is defined as :

$$\begin{aligned}\mathcal{L} = &- \mathrm{KL}\left(q_t(\boldsymbol{u}_t)||p(\boldsymbol{u}_t|\theta_t)\right) + \mathbb{E}_{q_t(\boldsymbol{u}_t,\boldsymbol{f}_t)}\left[\log p(y_t|\boldsymbol{f}_t)\right] \\ &- \mathrm{KL}(q_t(\boldsymbol{u}_{t-1})||q_{t-1}(\boldsymbol{u}_{t-1})) \\ &+ \mathrm{KL}(q_t(\boldsymbol{u}_{t-1})||p(\boldsymbol{u}_{t-1}|\theta_{t-1}))\end{aligned}$$

The terms of the first line correspond to a classical SVGP problem and the second line express the KL divergence with the previous variational posterior. The distributions are defined as :

$$\begin{aligned}q_t(\boldsymbol{u}_t) =& \mathcal{N}\left(\boldsymbol{\mu}_t, \Sigma_t\right) \\ p(\boldsymbol{u}_t|\theta_t) =& \mathcal{N}\left(0, K_{Z_t}\right) \\ q_t(\boldsymbol{u}_{t-1}) =& \int p(\boldsymbol{u}_{t-1}|\boldsymbol{u}_t) q_t(\boldsymbol{u}_t) d\boldsymbol{u}_t \\ =& \mathcal{N}\left(\kappa_{Z_{t-1}Z_t}\boldsymbol{\mu}_t, \widetilde{K}_{Z_{t-1}}\right) \\ \widetilde{K}_{Z_{t-1}} =& K_{Z_{t-1}} + \kappa_{Z_{t-1}Z_t}\Sigma_t\kappa_{Z_{t-1}Z_t}^\top \\ &- K_{Z_{t-1}Z_t}K_{Z_t}^{-1}K_{Z_tZ_{t-1}} \\ q_{t-1}(\boldsymbol{u}_{t-1}) =& \mathcal{N}\left(\boldsymbol{\mu}_{t-1}, \Sigma_{t-1}\right) \\ p(\boldsymbol{u}_{t-1}||\theta_{t-1}) =& \mathcal{N}(0, \underbrace{K'_{Z_{t-1}}}_{\text{Given } \theta_{t-1}})\end{aligned}$$

The first terms ares

$$\begin{aligned}&\mathrm{KL}(q_t(\boldsymbol{u}_t)||p(\boldsymbol{u}_t|\theta_t) = \\ &\frac{1}{2}\left(\log|K_{Z_t}| - \log|\Sigma_t| - M_t\right. \\ &\left. + \mathrm{tr}(K_{Z_t}^{-1}\Sigma_t) + \boldsymbol{\mu}_t^\top K_{Z_t}^{-1}\boldsymbol{\mu}_t\right)\end{aligned}$$

And for $p(\boldsymbol{y}_t|\boldsymbol{f}_t) = \prod_{i=1}^B \mathcal{N}(y_i|f_i, \sigma)$. The expected log-likelihood is given by L

$$\begin{aligned}\mathbb{E}_{q_t(\boldsymbol{u}_t,\boldsymbol{f}_t)}\left[\log p(\boldsymbol{y}_t|\boldsymbol{f}_t)\right] = &-\frac{B}{2}\log 2\pi\sigma^2 \\ &- \frac{1}{2\sigma^2}\sum_{i=1}^B (y_i - \kappa_{X_iZ_t}\mu_t)^2 + \widetilde{K} + \kappa_{X_iZ_t}\Sigma_t\kappa_{X_iZ_t}^\top\end{aligned}$$

Writing the second terms fully we get :

$$\begin{aligned}&\mathrm{KL}(q_t(\boldsymbol{u}_{t-1})||p(\boldsymbol{u}_{t-1}|\theta_{t-1})) = \\ &\frac{1}{2}\left(\log|K'_{Z_{t-1}}| - \log|\widetilde{K}_{t-1}| - M_{t-1}\right. \\ &+ \mathrm{tr}((K'_{Z_{t-1}})^{-1}\widetilde{K}_{Z_{t-1}}) \\ &\left.+(\kappa_{Z_{t-1}Z_t}\boldsymbol{\mu}_t)^\top(K'_{Z_{t-1}})^{-1}\kappa_{Z_tZ_{t-1}}\boldsymbol{\mu}_t\right) \\ &\mathrm{KL}(q_t(\boldsymbol{u}_{t-1})||q_{t-1}(\boldsymbol{u}_{t-1})) = \\ &\frac{1}{2}\left(\log|\Sigma_{t-1}| - \log|\widetilde{K}_{Z_{t-1}}| - M_{t-1}\right. \\ &+ \mathrm{tr}(\Sigma_{t-1}^{-1}\widetilde{K}_{Z_{t-1}}) \\ &\left.+(\boldsymbol{\mu}_{t-1} - \kappa_{Z_tZ_{t-1}}\boldsymbol{\mu}_t)^\top\Sigma_{t-1}^{-1}(\boldsymbol{\mu}_{t-1} - \kappa_{Z_tZ_{t-1}}\boldsymbol{\mu}_t)\right)\end{aligned}$$

Subtracting the second term to the first we get:

$$\begin{aligned}&\mathrm{KL}_{t:t-1} = \\ &\mathrm{KL}(q_t(\boldsymbol{u}_{t-1})||p(\boldsymbol{u}_{t-1}|\theta_{t-1})) - \mathrm{KL}(q_t(\boldsymbol{u}_t)||q_{t-1}(\boldsymbol{u}_{t-1})) \\ &= \frac{1}{2}\left(\log|K'_{Z_{t-1}}| - \log|\Sigma_{t-1}| - \mathrm{tr}((\Sigma_{t-1}^{-1} - (K'_{Z_{t-1}})^{-1})\widetilde{K}_{Z_{t-1}})\right. \\ &- \boldsymbol{\mu}_{t-1}^\top\Sigma_{t-1}^{-1}\boldsymbol{\mu}_{t-1} + 2\boldsymbol{\mu}_{t-1}\Sigma_{t-1}^{-1}\kappa_{Z_{t-1}Z_t}\boldsymbol{\mu}_t \\ &\left.-(\kappa_{Z_{t-1}Z_t}\boldsymbol{\mu}_t)^\top(\Sigma_{t-1}^{-1} - (K'_{Z_{t-1}})^{-1})(\kappa_{Z_{t-1}Z_t}\boldsymbol{\mu}_t)\right) \\ &= \frac{1}{2}\left(\log|K'_{Z_{t-1}}| - \log|\Sigma_{t-1}| - \mathrm{tr}(D_{t-1}^{-1}\widetilde{K}_{t-1})\right. \\ &- \boldsymbol{\mu}_{t-1}^\top\Sigma_{t-1}^{-1}\boldsymbol{\mu}_{t-1} + 2\boldsymbol{\mu}_{t-1}\Sigma_{t-1}^{-1}\kappa_{Z_{t-1}Z_t}\boldsymbol{\mu}_t \\ &\left.-(\kappa_{Z_{t-1}Z_t}\boldsymbol{\mu}_t)^\top D_{t-1}^{-1}(\kappa_{Z_{t-1}Z_t}\boldsymbol{\mu}_t)\right)\end{aligned}$$

Where $D_t = \left(\Sigma_t^{-1} - K_{Z_t}^{-1}\right)^{-1}$.

Taking the derivative of $\mathcal{L}$ given $\boldsymbol{\mu}_t$ and Σ_t gives us directly the optimal solution for Gaussian regression:

$$\begin{aligned}\Sigma_t^* =& \left(\sigma^{-2}\kappa_{X_tZ_t}^\top\kappa_{X_tZ_t} + \kappa_{Z_{t-1}Z_t}^\top D_{t-1}^{-1}\kappa_{Z_{t-1}Z_t} + K_{Z_t}^{-1}\right)^{-1} \\ \boldsymbol{\mu}_t^* =& \Sigma_t\left(\kappa_{X_tZ_t}^\top\sigma^{-2}\boldsymbol{y}_t + \kappa_{Z_{t-1}Z_t}^\top\Sigma_{t-1}\boldsymbol{\mu}_{t-1}\right)\end{aligned}$$

Rewritten in natural parameters terms:

$$\begin{aligned}\eta_1^t =& \kappa_{X_tZ_t}^\top\sigma^{-2}\boldsymbol{y}_t + \kappa_{Z_{t-1}Z_t}^\top\eta_1^{t-1} \\ \eta_2^t =& -\frac{1}{2}\left(\kappa_{X_tZ_t}^\top\sigma^{-2}I\kappa_{X_tZ_t}\right. \\ &\left.+\kappa_{Z_{t-1}Z_t}^\top\left(-2\eta_2^{t-1} - K_{Z_{t-1}}^{-1}\right)\kappa_{Z_{t-1}Z_t} + K_{Z_t}^{-1}\right)\end{aligned}$$

A.2. Hyper-parameter derivatives

Given θ a kernel hyperparameter and $J_{\square\square} = \frac{dK_{\square\square}}{d\theta}$ the derivatives are given by:

$$\begin{aligned}
\frac{d\mathrm{KL}_{t:t-1}}{d\theta_t} =& -\frac{1}{2}\mathrm{tr}\left(D_{t-1}^{-1}\frac{d\widetilde{K}_{Z_{t-1}}}{d\theta_t}\right) \\
&+ \boldsymbol{\mu}_{t-1}\Sigma_{t-1}^{-1}\frac{d\kappa_{Z_{t-1}Z_t}}{d\theta_t}\boldsymbol{\mu}_t \\
&- (\kappa_{Z_{t-1}Z_t}\boldsymbol{\mu}_t)^\top D_{t-1}^{-1}(\frac{d\kappa_{Z_{t-1}Z_t}}{d\theta_t}\boldsymbol{\mu}_t) \\
\frac{d\kappa_{Z_{t-1}Z_t}}{d\theta_t} =& \frac{dK_{Z_{t-1}Z_t}}{d\theta_t}K_{Z_t}^{-1} + K_{Z_tZ_{t-1}}\frac{dK_{Z_t}^{-1}}{d\theta_t} \\
=& (J_{Z_tZ_{t-1}} - \kappa_{Z_tZ_{t-1}}J_{Z_t})K_{Z_t}^{-1} = \iota_{Z_{t-1}Z_t} \\
\frac{d\widetilde{K}_{Z_{t-1}}}{d\theta_t} =& \frac{dK_{Z_{t-1}}}{d\theta_t} + 2\frac{d\kappa_{Z_{t-1}Z_t}}{d\theta_t}\Sigma_t\kappa_{Z_tZ_{t-1}}^\top \\
&- \frac{d\kappa_{Z_{t-1}Z_t}}{d\theta_t}K_{Z_tZ_{t-1}} - \kappa_{Z_{t-1}Z_t}\frac{dK_{Z_tZ_{t-1}}}{d\theta_t} \\
=& J_{Z_{t-1}} + 2\iota_{Z_{t-1}Z_t}\Sigma_t\kappa_{Z_{t-1}Z_t}^\top \\
&- \iota_{Z_{t-1}Z_t}K_{Z_tZ_{t-1}} - \kappa_{Z_{t-1}Z_t}J_{Z_tZ_{t-1}}
\end{aligned}$$

$$\frac{d\mathrm{KL}(q_t(\boldsymbol{u}_t)||p(\boldsymbol{u}_t|\theta_t)}{d\theta_t}$$

Special derivative given the variance :

$$\frac{dKL_a}{dv} = -\frac{1}{2}\left(\mathrm{tr}\left(D_a^{-1}\left[\frac{1}{v}(K_{aa} - K_{ab}K_{bb}^{-1}K_{ba})\right]\right)\right)$$

A.3. Comparison with SVI

If we take the special case where inducing points do not change between iterations, then $\kappa_{Z_{t-1}Z_t} = I$ and $K_{Z_{t-1}} = K_{Z_t}$. The updates become

$$\begin{aligned}
\eta_1^t =& \kappa_{X_tZ_t}^\top\sigma^{-2}\boldsymbol{y}_t + \eta_1^{t-1} \\
\eta_2^t =& -\frac{1}{2}\left(\kappa_{X_tZ_t}^\top\sigma^{-2}\kappa_{X_tZ_t} + \left(-2\eta_2^{t-1} - K_{Z_t}^{-1}\right) + K_{Z_t}^{-1}\right) \\
=& -\frac{1}{2}\kappa_{X_tZ_t}^\top\sigma^{-2}\kappa_{X_tZ_t} + \eta_2^{t-1}
\end{aligned}$$

Compared to the SVI updates:

$$\begin{aligned}
\eta_1^t =& \eta_1^{t-1} + \rho\left(\frac{N}{|B|}\left(\kappa_{X_tZ_t}^\top\sigma^{-2}\boldsymbol{y}_t\right) - \eta_1^{t-1}\right) \\
\eta_2^t =& \eta_2^{t-1} + \rho\left(-\frac{1}{2}\left(\frac{N}{|B|}\kappa_{X_tZ_t}^\top\sigma^{-2}\kappa_{X_tZ_t} + K_{Z_t}^{-1}\right) - \eta_2^{t-1}\right)
\end{aligned}$$

If we ignore ρ by setting it as 1:

$$\begin{aligned}
\eta_1^t =& \frac{N}{|B|}\left(\kappa_{X_tZ_t}^\top\sigma^{-2}\boldsymbol{y}_t\right) \\
\eta_2^t =& -\frac{1}{2}\left(\frac{N}{|B|}\kappa_{X_tZ_t}^\top\sigma^{-2}\kappa_{X_tZ_t} + K_{Z_t}^{-1}\right)
\end{aligned}$$

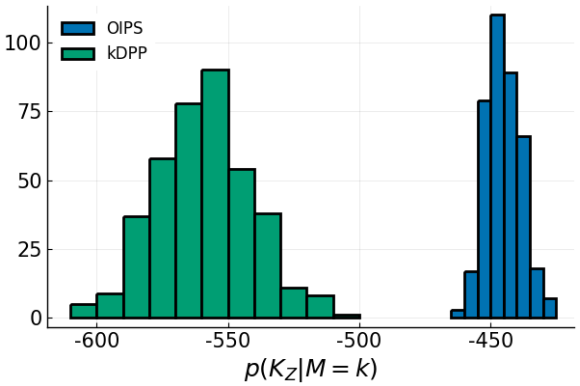


Figure 3: Histogram of $p(Z|k = M)$ for the OIPS algorithm and k-DPPsampling

We forget completely the previous η_1.
To make it directly comparable to streaming:

SVI

$$\begin{aligned}
\eta_1^{t+1} =& (1-\rho)\eta_1^t + \rho\left(\frac{N}{|B|}\left(\kappa_f^\top\sigma^{-2}y\right)\right) \\
\eta_2^{t+1} =& (1-\rho)\eta_2^t + -\frac{1}{2}\rho\left(\frac{N}{|B|}\kappa_f^\top\sigma^{-2}\kappa_f + K_{bb}^{-1}\right) \\
\eta_1^t =& (1-\rho)^t\eta_0 + \sum_{i=1}^{t}(1-\rho)^{i-1}\rho\frac{N}{|B|}\kappa_f^\top\sigma^{-2}y^i
\end{aligned}$$

Streaming

$$\begin{aligned}
\eta_1^{t+1} =& \eta_1^t + \kappa_f^\top\sigma^{-2}y \\
\eta_2^{t+1} =& \eta_2^t - \frac{1}{2}\kappa_f^\top\sigma^{-2}\kappa_f
\end{aligned}$$

B. Deterministic algorithm as a DPP half-greedy sampling

We proceed to a simple experiment, where given a dataset, Abalone ($N = 4177, D = 7$), we repeatedly shuffle the data. We apply algorithm 1 parsing all the data to get the subset Z_{OIPS}. We use the resulting number of inducing points k as a parameter to sample from a k-DPP and obtain Z_{kDPP}. We compute the probabilities of $\log p(Z_{OIPS}|M = k)$ and $\log p(Z_{kDPP}|M = k)$ and report the histogram of the probabilities on figure 3 One can observe that the probability given by the OIPS algorithm is consistently higher as well as more narrow then the sampling. This can be explained by the fact that we deterministically constrain all the points to have a certain distance from each other and therefore put a deterministic limit on the determinant of K_Z.

C. Proof Theorem 1 : Bound on the number of points

Algorithm 1 can be interpreted as filling a domain with closed balls, where balls intersections are allowed but no center can be inside another ball. For a SE kernel we can compute the radius r (in euclidean space) of these balls :

$$\begin{aligned} k(x, x') &= \rho_{in} \\ \exp\left(-\frac{||x - x'||^2}{h^2}\right) &= \rho_{in} \\ ||x - x'||^2 &= -h^2 \log \rho_{in} \\ r &= h\sqrt{-\log \rho_{in}} \end{aligned}$$

We can bound the volume of the union of the balls by the union of inscribed hypercubes. The length of an inscribed hypercube in an hypersphere of radius r is $l = r\sqrt{D}/2$. Since the volume of the hypercube is defined to be smaller, this gives us an upper bound on the expected number of inducing points. Defining as K_n the number of inducing points at time n, the probability of having a point outside of the union of all k hypercubes is

$$\begin{aligned} p(K_{n+1} = k + 1 | K_n = k) &= \max\left(a^D - \sum_{i=1}^{k} l^D\right) \\ &= \max\left(a^D - kl^D, 0\right) \\ p_k^+ &= \max\left(a^D - k\alpha, 0\right) \end{aligned}$$

Where $\alpha = \left(\frac{r\sqrt{D}}{2}\right)^D$, is the volume of one hypercube and therefore the probability of a new sample to appear in it.

The probability of keeping the same number of points is

$$\begin{aligned} p(K_{n+1} = k | K_n = k) &= \min\left(\sum_{i=1}^{k} l^D, 1\right) \\ p_k^= &= \min(k\alpha, 1) \end{aligned}$$

We now consider the problem as a Markov chain where the state p is represented by a vector $\{p_i\}_{i=1}^N$ where $p_i = 1$ if there are i inducing points. The transition matrix P is given by :

$$P = \begin{pmatrix} p_1^= & 0 & 0 & 0 \\ p_1^+ & p_2^= & 0 & 0 \\ 0 & p_2^+ & \ddots & 0 \\ 0 & 0 & \ddots & 0 \\ 0 & 0 & p_{N-1}^+ & p_N^= \end{pmatrix}$$

If we define that we start with inducing points the initial state is $p^1 = \{1, 0, \dots, 0\}^\top$, the probability of having k balls after n steps is $p(K_n = k|p^1) = \left(P^n p^1\right)_k$ while the expected number of pointsis given by $\sum_k k \cdot p(K_n = k|p^1)$.

These sequence can be complex to compute. Instead we can approximate the final expectation by recursively computing the update given the expectation at the previous step:

$$\begin{aligned} &\mathbb{E}_{p(K_{n+1}|K_n = \mathbb{E}[K_n])}\left[K_{n+1}\right] \\ &= \mathbb{E}\left[K_n\right]\mathbb{E}\left[K_n\right]\alpha + (\mathbb{E}\left[K_n\right] + 1)(a^D - \mathbb{E}\left[K_n\right]\alpha) \\ &= a^D\mathbb{E}\left[K_n\right] + a^D - \mathbb{E}\left[K_n\right]\alpha = a^D + \mathbb{E}\left[K_n\right](a^D - \alpha) \end{aligned}$$

This is an arithmetico-geometric suite and given the original condition $\mathbb{E}\left[K_0\right] = 1$ and since $\alpha < a^D$ we can get a closed form solution for $\mathbb{E}\left[K_n\right]$:

$$\begin{aligned} \mathbb{E}\left[K_n\right] &= (a^D - \alpha)^n\left(1 - \frac{a^D}{\alpha}\right) + \frac{a^D}{\alpha} \\ &= \frac{a^D - (a^D - \alpha)^{n+1}}{\alpha} \end{aligned}$$

C.1. Empirical Comparison

We show the realization of this bound on uniform data with 3 dimensions, $\rho = 0.7$ and $l = 0.3$ on figure 4.

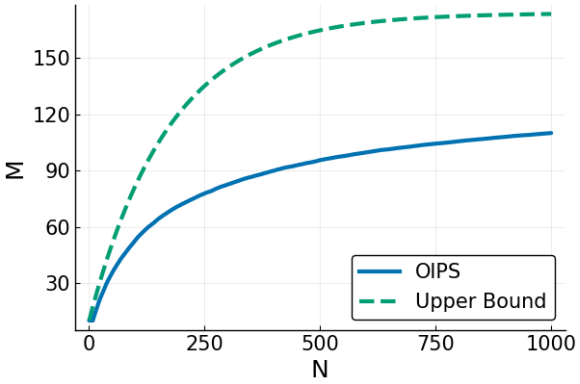


Figure 4: Bound on the number of inducing points accepted M given the number of seen points N vs the empirical estimation

D. Proof theorem 2 : Bounding the ELBO

We follow the approach of Burt et al. (2019) and Belabbas & Wolfe (2009). Burt et al. (2019) showed that the error between the ELBO and the log evidence was bounded by $\|K_X - K_{XZ}K_Z^{-1}K_{ZX}\|$. Where $\|\cdot\|$ is the Froebius norm. Using a k-DPP sampling (Kulesza & Taskar, 2011), they were able to show a bound on the expectation of this norm. We follow similar calculations with our deterministic algorithm for fixed kernel parameters. Let be K_X the kernel matrix of the full dataset and K_Z the submatrix given

the set of points $\{Z_i\}_{i=1}^M$. The Schur complement of K_{ZZ}, $S_C(K_{ZZ})$ in K_{XX} is given by $K_X - K_{XZ}K_Z^{-1}K_{ZX}$. Following a similar approach then Belabbas & Wolfe (2009) we bound the norm by the trace:

$$\|S_C(K_{ZZ})\| = \sqrt{\sum_{j=1}^{N-M} \overline{\lambda}_j} \leq \sum_{j=1}^{N-M} \overline{\lambda}_j = \text{tr}(S_C(K_{ZZ}))$$

Using the definiton of $S_C(K_{ZZ})$ we get :

$$\text{tr}(S_C(K_{ZZ})) = \sum_{i=1}^{N-M} K_{X_i} - K_{X_iZ}K_Z^{-1}K_{ZX_i}$$

where every element of the sum is a scalar. Taking $W^\top \overline{\Lambda} W$ the eigendecomposition of K_Z^{-1}, $w_i = WK_{X_iZ}$ and assuming a kernel variance v of 1 (although generalizable to all variances) and a translation invariant kernel such that $k(x, x) = 1$ we get :

$$K_{X_i} - K_{X_iZ}K_Z^{-1}K_{ZX_i} = 1 - w_i^\top \Lambda w_i = 1 - \sum_{j=1}^{M} \overline{\lambda_j}(w_i)_j^2$$
$$\leq 1 - \overline{\lambda}_{\min}\|w_i\|^2 = 1 - \overline{\lambda}_{\min}\|K_{X_iZ}\|^2 \leq 1 - \overline{\lambda}_{\min}\rho^2$$

Where we used the fact that at least X_i was close enough to at least one Z_j such that $k(X_i, Z_j) > \rho$. For clarity we replace $\overline{\lambda}_{\min} = \lambda_{\max}^{-1}$ where $\lambda_{\max}$ is the largest eigenvalue of K_Z. When summing over the trace we get the final bound :

$$\|K_X - K_{XZ}K_Z^{-1}K_{ZX}\| \leq (N - M)\left(1 - \frac{\rho^2}{\lambda_{\max}}\right)$$

Now by construction all off-diagonal terms of K_Z are smaller than ρ. Using the equality (Stewart & guang Sun, 1990)

$$|\lambda_i(A) - \lambda_i(B)| \leq \|A - B\|, \quad \forall i = 1, \ldots, N$$

We get that

$$|\lambda_{\max}(K_Z) - 1| \leq \|K_Z - I\|_2 = \sqrt{\sum_{i \neq j} (K_Z)_{ij}^2}$$
$$\leq M(M-1)\rho$$

Assuming $\lambda_{\max}(K_Z) \geq 1$, we get

$$\lambda_{\max}(K_Z) \leq 1 + M(M-1)\rho_{out}$$

Getting then the final bound :

$$\|K_X - Q_X\| \leq (N - M)\left(1 - \frac{\rho^2}{1 + M(M-1)\rho}\right)$$

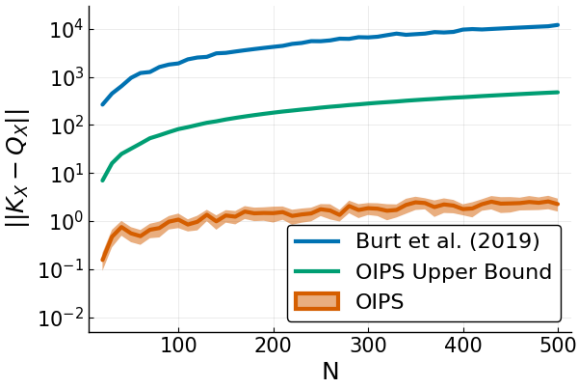


Figure 5: Evaluation of the $\|K_X - Q_X\|$ given the OIPS algorithm and computation of the bound from Burt et al. (2019) given in equation 6 and our bound given in equation 7

D.1. Empirical Comparison

These bounds are difficult to compare due to the different parameters characterizing them. Nevertheless we give an example by comparing the bound and the empirical value on toy data drawn uniformly in 3 dimensions in figure 5. For each N we ran our algorithm and input the required M in the bounds as the resulting number of selected inducing points. We show in the section 4 the empirical effect on the accuracy and on the number of points given the choice of ρ.

E. Experiments parameters

For every problem we use an isotropic Squared Exponential Kernel :

$$k(\mathbf{x}, \mathbf{x}') = v \exp\left(-\frac{\|\mathbf{x} - \mathbf{x}'\|^2}{h^2}\right)$$

Where h is initialized by taking the median of the lower triangular part of the pairwise distance matrix of the first subset of points and fixed for the rest of the training. Future work will involve working with kernel parameter optimization as well. We fix the noise of the Gaussian likelihood to $\sigma^2 = 0.01$.

IPs were optimized via ADAM ($\alpha = 10^{-2}$).