Stein Variational Gaussian Processes

Thomas Pinder[1], Christopher Nemeth[1], David Leslie[1]

[1]Department of Mathematics and Statistics, Lancaster University, UK

September 28, 2020

Abstract

We show how to use Stein variational gradient descent (SVGD) to carry out inference in Gaussian process (GP) models with non-Gaussian likelihoods and large data volumes. Markov chain Monte Carlo (MCMC) is extremely computationally intensive for these situations, but the parametric assumptions required for efficient variational inference (VI) result in incorrect inference when they encounter the multi-modal posterior distributions that are common for such models. SVGD provides a non-parametric alternative to variational inference which is substantially faster than MCMC but unhindered by parametric assumptions. We prove that for GP models with Lipschitz gradients the SVGD algorithm monotonically decreases the Kullback-Leibler divergence from the sampling distribution to the true posterior. Our method is demonstrated on benchmark problems in both regression and classification, and a real air quality example with 11440 spatiotemporal observations, showing substantial performance improvements over MCMC and VI.

1 Introduction

Gaussian processes (GPs) are highly expressive, non-parametric distributions over continuous functions and are frequently employed in both regression and classification tasks (Rasmussen and Williams, 2006). In recent years, GPs have received significant attention in the machine learning community due to their successes in domains such as reinforcement learning (Deisenroth et al., 2015), variance reduction (Oates et al., 2017), and optimisation (Mockus, 2012). This recent blossoming has been facilitated by advances in inference methods, and especially by variational inference (VI) which provides a tractable approach to fitting GP models to large and/or non-Gaussian data sets (e.g. Hensman et al., 2013; Cheng and Boots, 2017).

While computationally efficient, VI typically relies on the practitioner placing a parametric constraint upon the approximating posterior distribution. Unfortunately, this assumption can often severely inhibit the quality of the approximate posterior should the true posterior not belong to the chosen family of probability distributions, as often happens with GPs (Havasi et al., 2018). In this work we propose the use of Stein Variational Gradient Descent (SVGD) (Liu and Wang, 2016), a non-parametric development of the VI approach, as an effective inference method for GP

Table 1: Features of key inference methods for GP models.

Reference	$p(\mathbf{y}\|\mathbf{f})$	Sparse	Approx. posterior	Hyperparams	Inference
Opper and Archambeau (2008)	Binary	✗	Gaussian	Point estimate	Variational
Titsias (2009)	Gaussian	✓	Gaussian	Point estimate	Variational
Nguyen and Bonilla (2014)	Any	✗	Gaussian mixture	Point estimate	Variational
Hensman et al. (2015)	Any	✓	True posterior	Marginalised	MCMC
This work	Any	✓	True posterior	Marginalised	SVGD

models. SVGD can be thought of as a particle-based approach, whereby particles are sequentially transformed until they become samples from an arbitrary variational distribution that closely approximates the posterior of interest.

The most common (asymptotically) exact inference method for GPs is Markov chain Monte-Carlo (MCMC). However, sampling can be problematic if the posterior distribution is non-convex as the sampler can become trapped in local modes (Rudoy and Wolfe, 2006). Additionally, MCMC does not enjoy the same computational scalability as VI, and for this reason it is impractical for modelling problems with a large number of observations.

SVGD can be considered a hybrid of VI and Monte Carlo approaches, yielding benefits over both. The first benefit is removing the parametric assumption used in VI. The result of this is that inference through SVGD allows a richer variational distribution to be learned. A second benefit is that we need only evaluate the posterior distribution's score function, not the costly full posterior as in MCMC, meaning SVGD enjoys greater scalability than MCMC (especially due to the subsampling trick in Section 4). Finally, through the use of a kernel function acting over the set of particles, SVGD encourages full exploration over the posterior space, meaning that we are able to better represent the uncertainty in multimodal posteriors. Table 1 shows the position of this work within the current literature.

Our article demonstrates how to use SVGD to fit GPs to both Gaussian and non-Gaussian data, including when computational scalability is addressed through an inducing point representation of the original data. We prove that the SVGD scheme reduces the Kullback-Leibler (KL) divergence to the target distribution on each iteration. We empirically demonstrate the performance of SVGD in a range of both classification and regression datasets, comparing against traditional VI and modern implementations of MCMC for GPs, including in a large-scale spatiotemporal model for air quality in the UK. We release, at `https://gitbhub.com/RedactedForReview`, code for reproducing the experiments in Section 5, and a general library for fitting GPs using SVGD based entirely upon GPFlow (Matthews et al., 2017) and TensorFlow (Abadi et al., 2016).

2 Stein Variational Gradient Descent

Stein's discrepancy The foundation for SVGD is Stein's identity (Stein, 1972). For an arbitrary (continuously differentiable) density of interest p,

$$\mathbb{E}_{\boldsymbol{\lambda}\sim p}\Big[\underbrace{\nabla_{\boldsymbol{\lambda}}\log p(\boldsymbol{\lambda}) + \nabla_{\boldsymbol{\lambda}}\phi(\boldsymbol{\lambda})}_{\text{Stein operator: } \mathcal{A}_p\phi(\boldsymbol{\lambda})}\Big] = 0, \tag{1}$$

where $\phi : \mathbb{R}^d \to \mathbb{R}^d$ is any smooth function. Stein's identity has found uses in a range of modern machine learning problems including variance reduction (Oates et al., 2017), model selection (Kanagawa et al., 2019) and generative modelling (Pu et al., 2017). In our application, p will be the posterior density of interest. We can identify the internals of the expectation in (1) as the *Stein operator*, a quantity we denote as $\mathcal{A}_p\phi(\boldsymbol{\lambda})$.

Stein's identity (1) can be used to give a notion of *distance* between any two probability distributions. Replacing the density p under which we evaluate the expectation in (1) with a second density q, (1) is 0 if and only if $p = q$. For a class of functions $\mathcal{F}$, the Stein discrepancy is defined to be

$$\sqrt{\mathbb{D}(q,p)} = \sup_{\phi\in\mathcal{F}} \mathbb{E}_{\boldsymbol{\lambda}\sim q}\left[\text{trace}(\mathcal{A}_p\phi(\boldsymbol{\lambda}))\right]. \tag{2}$$

Taking $\mathcal{F}$ to be the unit ball of the Reproducing Kernel Hilbert Space (RKHS) $\mathcal{H}^d$ of a positive definite kernel $\kappa(x, x')$, we can optimise (2) explicitly (Liu et al., 2016). This functional optimisation yields the closed form solution to (2)

$$\hat{\phi}(\boldsymbol{\lambda}) = \beta(\boldsymbol{\lambda})/\left\|\beta\right\|_{\mathcal{H}^d} \quad \text{where} \quad \beta(\boldsymbol{\lambda}) = \mathbb{E}_{\boldsymbol{\lambda}'\sim q}\left[\mathcal{A}_p\kappa(\boldsymbol{\lambda},\boldsymbol{\lambda}')\right]. \tag{3}$$

This explicit form of the maximiser is the critical enabler of SVGD.

Stein Variational Gradient Descent In SVGD, as in classical VI, we approximate the true posterior distribution p with a variational distribution q that minimises the KL divergence between p and q. The innovation in SVGD is that we assume no parametric form for q. From an arbitrary initial distribution q_0, SVGD iterates through a series of pushforward transformations that reduce the KL divergence between the target distribution and the distribution q_t.

In particular, the transformation is defined by considering a $\boldsymbol{\lambda} \sim q_t$, and a mapping $\mathcal{T}(\boldsymbol{\lambda}) = \boldsymbol{\lambda} + \epsilon\phi(\boldsymbol{\lambda})$ for an arbitrary function ϕ and perturbation magnitude ϵ. The transformed distribution q_{t+1} is the distribution of $\mathcal{T}(\boldsymbol{\lambda})$. Following Gorham and Mackey (2017) and Liu et al. (2016), we assume ϕ lives in the RKHS $\mathcal{H}^d$. Under this assumption, it is shown that

$$\nabla_\epsilon \,\text{KL}\left(q_{t+1}\|p\right)\big|_{\epsilon=0} = -\mathbb{E}_{\boldsymbol{\lambda}\sim q_t}\left[\text{trace}\left(\mathcal{A}_p\boldsymbol{\phi}(\boldsymbol{\lambda})\right)\right]. \tag{4}$$

Comparing (4) and (2) shows that using $\phi = \hat{\phi}$ from (3) maximally decreases the KL divergence.

To implement the recursion, we maintain a finite set of J samples that empirically represent q, referred to as *particles*. These particles $\Lambda = \{\boldsymbol{\lambda}^j\}_{j=1}^J$ are initially sampled

independently from q_0, which is typically taken to be the prior distribution.[1] The transformation $\mathcal{T}$ is then applied repeatedly to the set of particles, where at each stage the optimal $\hat{\phi}$ from (3) is estimated empirically using the particles $\Lambda_t = \{\boldsymbol{\lambda}_t^m\}_{j=1}^J$ at the t^{th} iteration:

$$\hat{\phi}_{\Lambda_t}(\boldsymbol{\lambda}) = \frac{1}{J}\sum_{j=1}^{J}\Big[\underbrace{\kappa(\boldsymbol{\lambda}_t^j, \boldsymbol{\lambda})\nabla_{\boldsymbol{\lambda}} \log p(\boldsymbol{\lambda}_t^j)}_{\text{Attraction}} + \underbrace{\nabla_{\boldsymbol{\lambda}}\kappa(\boldsymbol{\lambda}_t^j, \boldsymbol{\lambda})}_{\text{Repulsion}}\Big]. \tag{5}$$

When the process terminates after T iterations, each of the particles is a sample from a distribution q_T with low KL divergence from the target p, and we can use the particles in the same way as a standard Monte Carlo sample.

Examining the update step in (5), it can be seen that the first term transports particles towards areas in the posterior distribution that represent high probability mass. Conversely, the second term is the derivative of the kernel function; a term that will penalise particles being too close to one another (see Appendix 7). In the case that $J = 1$, the summation in (5) disappears, and the entire scheme reduces to regular gradient based optimisation. Additionally, there is no danger of running SVGD with J too large as, by the *propagation of chaos* (Kac, 1976), the final distribution of the i^{th} particle is invariant to J as the number of iteration steps $T \to \infty$ (Liu and Wang, 2016).

Connection to variational inference Typically, in VI we minimise the KL divergence between a ξ-parameterised variational distribution $q_\xi(\boldsymbol{\lambda})$ and the target density:

$$\xi^\star = \arg\min_{\xi} \mathrm{KL}(q_\xi(\boldsymbol{\lambda})||p(\boldsymbol{\lambda})). \tag{6}$$

ξ often parameterises a family $\{q_\xi\}$ of Gaussian distributions. The resultant parameters $\xi^\star$ are then used to form the optimal variational distribution $q_\xi^\star(\boldsymbol{\lambda})$, used in place of the intractable $p(\boldsymbol{\lambda})$.

In a regular VI framework, the explicit form placed on q can be highly restrictive, particularly if the true posterior density is not well approximated by the variational family selected. The nonparametric approach of SVGD, given by (5), allows for a more flexible representation of the posterior geometry, and not just the Gaussian distribution captured in regular VI. An additional advantage is that SVGD only requires evaluation of the posterior's score function, a quantity that is invariant to the normalisation constant and can be unbiasedly approximated in large data settings.

Related SVGD work SVGD has been used in the context of fitting Bayesian logistic regression and Bayesian neural networks (Liu et al., 2017). Further, SVGD was used in the context of variational autoencoders (VAE) to model the latent space (Pu et al., 2017). By relaxing the Gaussian assumption that is typically made of the

[1] In the asymptotic limit, the final particle values are invariant to the initial distribution that particles are initialised from (Papamakarios et al., 2019)

latent space, it was possible to learn a more complex distribution over the latent space of the VAE. Further examples of the applications of SVGD can be found in reinforcement learning (Liu et al., 2017), Bayesian optimisation (Gong et al., 2019), and in conjunction with deep learning (Grathwohl et al., 2020). Theoretical analysis has also established connections between SVGD and the overdamped Langevin diffusion (Duncan et al., 2019), and black-box variational inference (Chu et al., 2020).

This article leverages the effective and efficient SVGD optimisation framework to address the computational and multi-modality challenges endemic in GP inference.

3 Gaussian Processes

Consider data $(X, \mathbf{f}, \mathbf{y}) = \{\mathbf{x}_i, f_i, y_i\}_{i=1}^N$ where $\mathbf{x}_i \in \mathbb{R}^d$, and $y_i \in \mathbb{R}$ is a stochastic observation depending on $f_i = f(x_i)$ for some latent function f. Let $k_{\boldsymbol{\theta}}(\mathbf{x}, \mathbf{x}')$ be a positive definite kernel function parameterised by a set of hyperparameters $\boldsymbol{\theta}$ with resultant Gram matrix $K_{\mathbf{xx}} = k_{\boldsymbol{\theta}}(\mathbf{x}, \mathbf{x}')$. Following standard practice in the literature and assuming a zero mean function, we can posit the hierarchical GP framework as

$$p(\mathbf{y} \,|\, \mathbf{f}, \boldsymbol{\theta}) = \prod_{i=1}^{N} p(y_i \,|\, f_i, \boldsymbol{\theta}), \qquad \mathbf{f} \,|\, X, \boldsymbol{\theta} \sim \mathcal{N}(0, K_{\mathbf{xx}}), \qquad \boldsymbol{\theta} \sim p_0. \tag{7}$$

From the generative model in (7), we can see that the posterior distribution of a GP is

$$p(\mathbf{f}, \boldsymbol{\theta} \,|\, \mathbf{y}) = \frac{1}{C} p(\mathbf{y} \,|\, \mathbf{f}, \boldsymbol{\theta}) p(\mathbf{f} \,|\, \boldsymbol{\theta}) p_0(\boldsymbol{\theta}), \tag{8}$$

where C denotes the unknown normalisation constant of the posterior. Often we are interested in using the posterior to make new function predictions $f^\star$ for test data $X^\star$,

$$p\left(\mathbf{f}^\star | \mathbf{y}\right) = \iint p\left(\mathbf{f}^\star | \mathbf{f}, \boldsymbol{\theta}\right) p(\mathbf{f}, \boldsymbol{\theta} | \mathbf{y}) \, \mathrm{d}\boldsymbol{\theta} \, \mathrm{d}\mathbf{f}. \tag{9}$$

When the likelihood $p(y_i \,|\, f_i, \boldsymbol{\theta})$ is Gaussian the posterior predictive distribution conditional on $\boldsymbol{\theta}$ is analytically available as we can marginalise $\mathbf{f}$ out from (9), and inference methods focus on $\boldsymbol{\theta}$. For non-Gaussian likelihoods, we must approximate the integral over $\mathbf{y}$ using alternative approaches. Some of the most common methods are Laplace approximations (Williams and Barber, 1998), expectation-propagation (Minka, 2001), MCMC (Murray et al., 2010), or VI (Opper and Archambeau, 2008).

Even when $\mathbf{f}$ can be marginalised, computing (9) requires the inversion of $K_{\mathbf{xx}}$, an operation that requires $\mathcal{O}(N^3)$ computation. To reduce this cost when N is large, one can introduce a set of *inducing points* $Z = \{\mathbf{z}_i\}_{i=1}^M$ that live in the same space as X, such that $M \ll N$ (Snelson and Ghahramani, 2006). The fundamental assumption made in the majority of sparse frameworks (see Quiñonero-Candela and Rasmussen, 2005) is that the elements of $\mathbf{f}$ and $\mathbf{f}^\star$ are conditionally independent given $\mathbf{u} = f(Z)$. Inference within this model only requires inversion of the Gram matrix $K_{\mathbf{ZZ}} = k_{\boldsymbol{\theta}}(Z, Z)$, which reduces the cost for posterior inference from $\mathcal{O}(N^3)$ to $\mathcal{O}(NM^2)$. Inference for sparse GP models has thus far either required $\boldsymbol{\theta}$ to be fixed so that VI can be deployed (e.g. Titsias, 2009; Hensman et al., 2013), or has

used extremely computationally-intensive MCMC schemes (e.g. Hensman et al., 2015). A full review of scalable GPs can be found in Liu et al. (2019).

We demonstrate that SVGD is able to retain the ability to carry out joint inference over $\mathbf{f}$, $\boldsymbol{\theta}$ and, where necessary, $\mathbf{u}$ without incurring the large overheads of MCMC schemes. In the rest of this article we do not explicitly discuss inference in sparse GPs, but all our results apply equally to sparse formulations as to the full GP models discussed; the sparse approximation simply corresponds to a variant GP (Quiñonero-Candela and Rasmussen, 2005).

4 Stein Variational Gaussian Processes

Gaussian data Recalling the posterior distribution (8) of a GP, we seek to approximate this distribution using a finite set of particles learned through SVGD. When the data likelihood is Gaussian, we are able to analytically integrate out f and use SVGD to learn the kernel hyperparameters and observation noise σ^2, which comprise $\boldsymbol{\theta}$. This means that each particle $\boldsymbol{\lambda}^j$ represents a sampled $\boldsymbol{\theta}$ value. SVGD is useful even in these cases, as estimating kernel parameters such as the lengthscale can be particularly challenging for VI and MCMC. This is due to their unidentifiable nature that often manifests itself through a multimodal marginal posterior distribution. SVGD is able to efficiently capture these modes, and consequently gives more realistic posterior predictions (Palacios and Steel, 2006; Gelfand et al., 2010).

Non-Gaussian data In the general GP setup in (7), where $y_i \,|\, f_i, \boldsymbol{\theta}$ is non-Gaussian, we are required to learn the latent values $\mathbf{f}$ of the GP in addition to $\boldsymbol{\theta}$. To decouple the strong dependency that exists between $\mathbf{f}$ and $\boldsymbol{\theta}$, we centre (or 'whiten') the GP's covariance matrix such that $\mathbf{f} = L_{\boldsymbol{\theta}}\boldsymbol{\nu}$, where $L_{\boldsymbol{\theta}}$ is the lower Cholesky decomposition of the Gram matrix $K_{\mathbf{xx}}$, and $\nu_i \sim \mathcal{N}(0, 1)$. Applying such a transformation has been shown to enhance the performance of inferential schemes in the GP setting (Murray et al., 2010; Hensman et al., 2015; Salimbeni and Deisenroth, 2017). Once whitened, we can use the joint posterior distribution $p(\boldsymbol{\theta}, \boldsymbol{\nu} \,|\, X, \mathbf{y})$ as the target distribution for SVGD and *post hoc* deterministically transform the posterior samples to give $p(\boldsymbol{\theta}, \mathbf{f} \,|\, X, \mathbf{y})$.

SVGD requires evaluation of the score function of the density at the current particle values to evaluate (5). Using automatic differentiation, this is a trivial task. However, it is accompanied by a computational cost that scales quadratically with N since we can no longer marginalise $\mathbf{f}$. For this reason, in datasets surpassing several thousand datapoints, we estimate the score function using subsampled mini-batches $\Psi \subset \{1, \ldots, N\}$. The end result of this is a score function approximation that can be written as

$$\nabla_{\boldsymbol{\theta},\boldsymbol{\nu}} \log p(\boldsymbol{\theta}, \boldsymbol{\nu} \,|\, X, \mathbf{y}) \approx \nabla_{\boldsymbol{\theta},\boldsymbol{\nu}} \log p_0(\boldsymbol{\theta}, \boldsymbol{\nu}) + \frac{N}{|\Psi|} \sum_{i \in \Psi} \nabla_{\boldsymbol{\theta},\boldsymbol{\nu}} \log p(\mathbf{y}_i \,|\, \boldsymbol{\theta}, \nu_i). \tag{10}$$

Posterior predictions Once a set of particles Λ has been learned, we can use the value of each particle to make predictive inference. Recalling that the primary moti-

vation of SVGD is to enable accurate predictive inference in posterior distributions with complex and multimodal geometries, naively taking the mean particle value for each parameter as the final estimate of each parameter in the GP could become problematic when the posterior is complex. For this reason, to obtain a predictive mean and variance, we sample K times from the predictive posterior of the GP for each of the J particles and compute the mean and variance of the predictive samples. To elucidate this notion, the procedure is summarised in Algorithm 2 .

Algorithm 1 Pseudocode for fitting a Gaussian process using T iterations of SVGD.

Require: Base distribution q_0. Target distribution $p(\boldsymbol{\lambda} \mid X, \mathbf{y})$ where $\boldsymbol{\lambda} = \{\boldsymbol{\theta}, \boldsymbol{\nu}\}$.
Create $\Lambda_0 = \{\boldsymbol{\lambda}_0^j\}_{j=1}^J$ where $\boldsymbol{\lambda}_0^j \overset{\text{iid}}{\sim} q_0$.

for t in 1:T **do**
  for j in 1:J **do**
    $\boldsymbol{\lambda}_t^j \leftarrow \boldsymbol{\lambda}_{t-1}^j + \epsilon \hat{\phi}_{\Lambda_{t-1}}(\boldsymbol{\lambda}_{t-1}^j)$ (see (5))
  end for
  $\Lambda_t = \{\boldsymbol{\lambda}_t^j\}_{j=1}^J$
end for
return Λ_T

Algorithm 2 Pseudocode for predictive inference over test inputs $X^\star$.

Require: Learned set of particles $\{\boldsymbol{\lambda}_j\}_{j=1}^J$
Initialise `sample`= {}
for j in 1:J **do**
  Set $(\boldsymbol{\theta}, \boldsymbol{\nu}) = \boldsymbol{\lambda}^j$
  for k in 1:K **do**
    Sample $\mathbf{y}^* \sim p(\cdot \mid X^*, \boldsymbol{\theta}, \boldsymbol{\nu})$
    Append $\mathbf{y}^*$ to `sample`
  end for
end for
return `mean(sample), var(sample)`

Optimisation guarantees We would like to show that the SteinGP in Algorithm 1 iteratively improves the posterior approximation. Consider the variational distributions q_t of the particles at time t. We show that the Kullback-Leibler divergence between these distributions and the target p decreases monotonically as t increases.

Theorem 1 *Consider SVGD in a general model with* $\log p(\boldsymbol{\lambda})$ *at least twice continuously differentiable and* $\nabla \log p(\boldsymbol{\lambda})$ *is smooth with Lipschitz constant* L*. The Kullback-Leibler divergence between the target distribution* p *and its SVGD approximation* q_t *at iteration* t *is monotonically decreasing, satisfying*

$$KL(q_{t+1}||p) - KL(q_t||p) \leq -\epsilon \mathbb{D}(q_t, p)^2 (1 - \mathbb{E}_{\boldsymbol{\lambda} \sim q_t} \left[\epsilon L \kappa(\boldsymbol{\lambda}, \boldsymbol{\lambda}')/2 + 2\nabla_{\boldsymbol{\lambda}, \boldsymbol{\lambda}'} \kappa(\boldsymbol{\lambda}, \boldsymbol{\lambda}')\right]) \tag{11}$$

It can be shown that for the GP models considered in this paper the gradients are Lipschitz smooth and therefore Theorem 1 holds. Additionally, Theorem 8 of Gorham and Mackey (2017) establishes weak convergence for a sequence of probability measures $(q_t)_{t \geq 1}$, where $q_t \implies p$ if $\mathbb{D}(q_t, p) \to 0$, and so it follows from Theorem 1, that $q_t \implies p$ as $t \to \infty$.

5 Experiments

Implementation All code for this work is written in TensorFlow (Abadi et al., 2016) and extends the popular Gaussian process library GPFlow (Matthews et al., 2017). Code to replicate experiments can be found at `https://github.com/RedactedForReview`. A demonstration of the code is given in Appendix 9.

Benchmark datasets For the datasets used in Section 5.1 and Section 5.2, models are fitted using 70% of the full dataset, whilst the remaining 30% is used for model assessment. The partitioning of data, hyperparameter initialisation and computing environment used is the same for all models. For each dataset we standardise the inputs and outputs to zero mean and unit standard deviation.

SVGD implementation details Throughout our experiments we use an RBF kernel to specify the SVGD step (5) (which is an independent choice from the kernel within the GP model). The RBF kernel's variance is set to 1, and we select the lengthscale at each iteration of SVGD using the median rule as in Liu and Wang (2016). We run experiments with $J = 2$, 5, 10 and 20 particles (labelled SteinGP2 through to SteinGP20).

Further experimental details can be found in Appendix 10.

5.1 Regression Data

We assess the performance of our proposed model on 13 publicly available UCI datasets. The number of observations in each dataset ranges from 23 to 4898 and in dimension from 4 to 128. Full details can be found in Appendix 11. Due to the relatively small size of each dataset, we make no sparse assumptions in this experiment.

We benchmark SteinGP against two comparative modes of inference: a variational GP (VI) whereby natural gradients are used to learn variational parameters, and a maximum likelihood (ML) approach whereby the latent function has been marginalised analytically. For both the VI and ML schemes we use the Adam optimiser (Kingma and Ba, 2015) with a learning rate of 0.01[2]. For all models, an Automatic Relevance Determination (ARD) squared exponential kernel is used with lengthscale values initialised to the square root of the data's dimensionality.

In Table 2 we report the test log-likelihood for each model on the held-out test data where a significant difference exists between one or more of the three inference methods. In all five datasets, a SteinGP achieves the highest test log-likelihood, showing inference through SVGD is not prone to overfitting. In the remaining eight datasets there is no negligible difference between the methods (Table 6 in Appendix 12). Further, from Table 7 in Appendix 12, we can see that a SteinGP with 2 particles converges fastest in 7 of the 13 datasets, and even with 20 particles

[2]We found a learning rate of 0.01 to give faster convergence than the recommended learning rate of 0.001 with no detriment to predictive quality.

Table 2: Mean test log-likelihoods (larger is better) over 5 independent data splits, with bold values indicating the best performing method. Our SteinGP with 2, 5, 10 and 20 particles is compared against a GP fitted using VI and maximum likelihood (ML). For brevity, only the datasets where there is a significant difference between the best performing and one, or more, alternative methods are reported here. The full table can be found in Table 6 in Appendix 12.

Dataset	SteinGP2	SteinGP5	SteinGP10	SteinGP20	VI	ML
Airfoil	$\mathbf{0.06 \pm 0.04}$	0.06 ± 0.04	0.05 ± 0.06	0.05 ± 0.05	0.03 ± 0.03	0.03 ± 0.03
Challenger	-1.53 ± 0.45	-1.52 ± 0.43	$\mathbf{-1.46 \pm 0.32}$	-1.53 ± 0.41	-1.51 ± 0.3	-1.51 ± 0.3
Concreteslump	$\mathbf{1.08 \pm 0.39}$	1.07 ± 0.41	1.06 ± 0.4	1.08 ± 0.39	0.13 ± 1.14	0.13 ± 1.14
Gas	0.88 ± 0.11	0.88 ± 0.11	$\mathbf{0.89 \pm 0.1}$	0.88 ± 0.11	0.79 ± 0.11	0.79 ± 0.11
Parkinsons	4.12 ± 0.05	4.12 ± 0.05	$\mathbf{4.14 \pm 0.03}$	4.13 ± 0.06	3.95 ± 0.04	3.95 ± 0.04
Winewhite	0.56 ± 0.05	0.57 ± 0.05	$\mathbf{0.57 \pm 0.05}$	0.57 ± 0.05	0.49 ± 0.05	0.55 ± 0.05

SteinGP20 is generally only one order of magnitude slower than basic maximum likelihood.

5.2 Classification

We further test our proposed mode of inference using GP models fitted to six benchmark binary classification datasets. The number of observations in each dataset ranges from 100 to 2201 and in dimension from 4 to 45. As discussed in Section 4, we are now tasked with learning the latent function's values along with the GP hyperparameters, and maximum likelihood is no longer a viable approach.

We now compare our SteinGP method to a variational approach equivalent to that used in Section 5.1 that learns a point-estimate of the GP hyperparameters $\boldsymbol{\theta}$ and a variational approximation of the latent values $\mathbf{f}$, and to an Hamiltonian Monte-Carlo (HMC) MCMC sampler that jointly learns $\boldsymbol{\theta}$ and $\mathbf{f}$. We start the HMC sampler near the maximum a posteriori (MAP) estimate which is found using Adam (Kingma and Ba, 2015). 5000 posterior samples are generated, with the first 2000 discarded as burn-in, and the remaining 3000 thinned by a factor of 10. Trace plots are used in each case to assess convergence of the predictive distribution.

Figure 1 shows the predictive accuracy and expected calibration error (ECE, Naeini et al., 2015) for each method on the six datasets. Subject to observation noise, it can be seen in Figure 1 that in all but one case (mammographic), the predictive accuracy of a SteinGP is equal to or better than the VI and HMC inferred GPs, and the calibration of the SVGD-inferred posteriors is better for that example. Further, this powerful predictive accuracy comes at a significantly reduced computational cost compared to MCMC (Table 8 in Appendix 12).

5.3 Fully Bayesian Sparse Gaussian Process

We demonstrate the scalability of our model on a large scale regression problem with 11440 observations coming from 130 point sensors of air quality in the Automatic

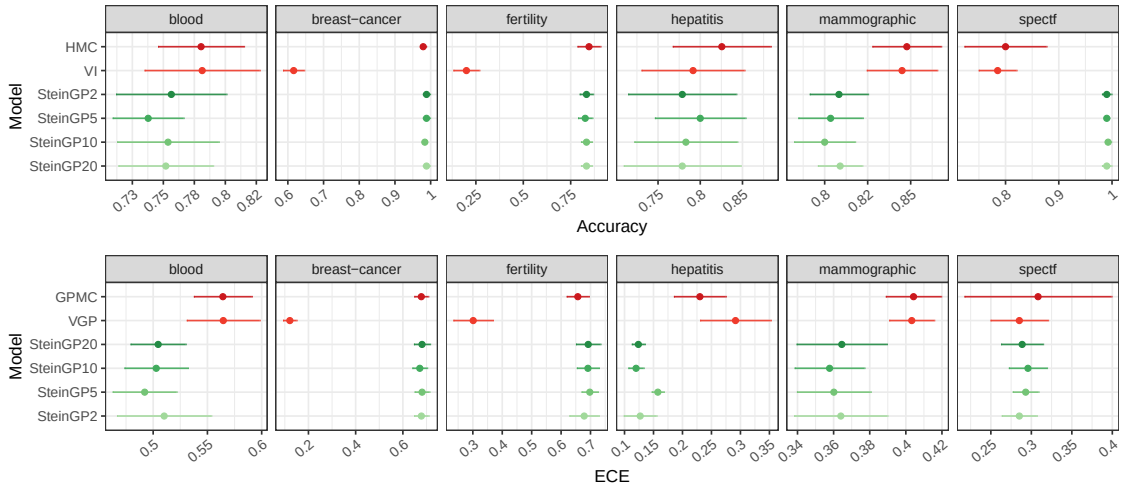

Figure 1: Predictive accuracy (top) and Expected Calibration Error (ECE, Naeini et al., 2015) of GP models fitted using HMC, VI and SVGD (SteinGP) where the proceeding integer denotes the number of particles used. Results are averaged over five independent partitions of the data, with error bars depicting a 95% confidence interval. For ECE, lower numbers are better.

Urban and Rural Network (AURN).[3] We infer the full posterior distribution of the spatiotemporal response surface of nitrogen oxide (NOx) levels in the UK from February 1$^{\text{st}}$ to February 22$^{\text{nd}}$ 2019. Data corresponding to NOx levels on February 6$^{\text{th}}$ are held out as test data, and we assess the predictive performance of each model on this data in Table 3. Each data point consists of a spatial location and timestamp. The GP model uses a Matern kernel with a smoothness of 3/2. All hyperparameters have Gamma priors with shape and scale values of 1 and 2 respectively. We use two different sparse models (see Section 3), fixing a set of either 300 or 800 inducing points in advance by using k-means clustering of the spatiotemporal locations.

We compare our method against HMC (Hensman et al., 2015). In both models we jointly infer the latent values of the GP and the GP hyperparameters. Conventional variational approaches (see Table 1) treat the GP's hyperparameters as fixed point values, whereas we would like to learn full posteriors in order to properly report the uncertainty in the predictions.

To improve the effectiveness of the HMC sampler, we optimise the marginal log-likelihood using the Adam optimiser for 300 iterations and start the MCMC from the MAP estimate. 10000 posterior samples are then drawn with the first 2000 discarded as burn-in and the remaining 8000 samples thinned by a factor of 5. Trace plots can be found in Appendix 12.1. Table 3 shows the significant computational improvements that can be achieved through a SteinGP, with only minimal reduction in model fit.

A depiction of the spatiotemporal surface learned through the sparse SteinGP can be seen in Figure 2, where we interpolate over a uniform grid the inferred f_i values on February 6$^{\text{th}}$. Comparative plots for the equivalent MCMC inferred surface,

[3]Crown 2019 copyright Defra via uk-air.defra.gov.uk, licensed under the Open Government Licence (OGL).

Table 3: Predictive performance of SteinGP with the proceeding integer denoting the number of particles used compared against the MCMC scheme (HMC). M=300 or 800 inducing points were used for each model, and metrics are reported on 5 independent initialisations of Z.

Metric	M	SteinGP2	SteinGP5	SteinGP10	SteinGP20	HMC
Log-likelihood		-1.62 ± 0.04	-1.61 ± 0.02	-1.6 ± 0.01	-1.61 ± 0.01	$\mathbf{-1.59} \pm 0.02$
RMSE	300	2359.63 ± 39.14	2336.93 ± 48.4	2358.54 ± 19.63	$\mathbf{2336.81} \pm 26.1$	2380.26 ± 104.32
Runtime (seconds)		$\mathbf{12.96} \pm 0.67$	26.83 ± 1.22	49.49 ± 1.79	94.88 ± 2.34	1959 ± 15.73
Log-likelihood		-1.41 ± 0.04	-1.42 ± 0.03	-1.40 ± 0.03	-1.38 ± 0.02	$\mathbf{-1.32} \pm 0.01$
RMSE	800	2297.64 ± 76.39	2304.19 ± 35.36	2282.42 ± 54	2287.33 ± 30.06	$\mathbf{1945.67} \pm 10.04$
Runtime (seconds)		$\mathbf{16.5} \pm 0.19$	36.42 ± 0.39	70.4 ± 1.21	137.87 ± 3.33	4802.55 ± 2.49

and predictive uncertainty plots, can be found in Appendix 12.2.

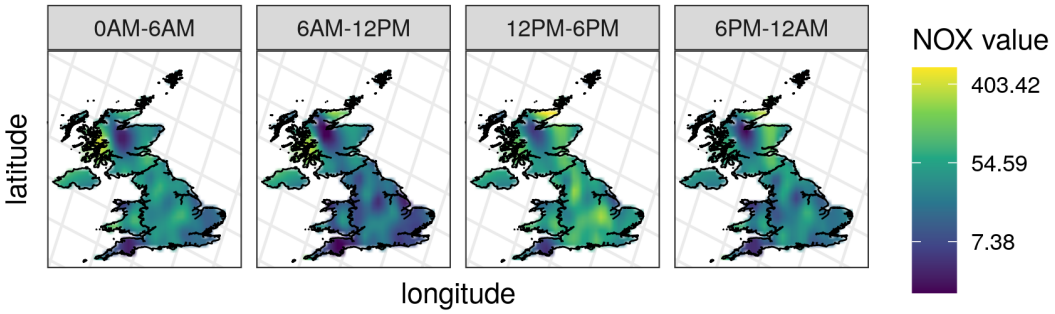

Figure 2: Spatiotemporal interpolation for February 6$^{\text{th}}$ using a SteinGP with 20 particles and 800 inducing points. Predictions were made over a uniform grid consisting of 500 points per timestamp.

6 Discussion

We have shown that SVGD can be used to provide comparable inferential quality to the gold standard HMC sampler in GP inference, while only requiring a computational cost comparable to VI. The ability to carry out joint inference over latent function values and kernel hyperparameters allows for a full and proper consideration of uncertainty in the inference.

For simple problems where the true posterior is Gaussian, very few SVGD particles are required to achieve strong inference, as can be seen in the experiments carried out in Section 5.1. Even in larger experiments, a small number of particles results in surprisingly strong inferential performance, at a fraction of the cost of MCMC.

References

Martín Abadi, Ashish Agarwal, Paul Barham, Eugene Brevdo, Zhifeng Chen, Craig Citro, Greg S. Corrado, Andy Davis, Jeffrey Dean, Matthieu Devin, Sanjay Ghemawat, Ian Goodfellow, Andrew Harp, Geoffrey Irving, Michael Isard, Yangqing Jia, Rafal Jozefowicz, Lukasz Kaiser, Manjunath Kudlur, Josh Levenberg, Dan Mane, Rajat Monga, Sherry Moore, Derek Murray, Chris Olah, Mike Schuster, Jonathon Shlens, Benoit Steiner, Ilya Sutskever, Kunal Talwar, Paul Tucker, Vincent Vanhoucke, Vijay Vasudevan, Fernanda Viegas, Oriol Vinyals, Pete Warden, Martin Wattenberg, Martin Wicke, Yuan Yu, and Xiaoqiang Zheng. TensorFlow: Large-Scale Machine Learning on Heterogeneous Distributed Systems. *arXiv:1603.04467 [cs]*, March 2016. arXiv: 1603.04467.

A. Berlinet and Christine Thomas-Agnan. *Reproducing kernel Hilbert spaces in probability and statistics*. Kluwer Academic, Boston, 2004. ISBN 978-1-4020-7679-4.

David C. Carslaw and Karl Ropkins. openair — An R package for air quality data analysis. *Environmental Modelling & Software*, 27-28:52–61, January 2012.

Ching-An Cheng and Byron Boots. Variational Inference for Gaussian Process Models with Linear Complexity. In I. Guyon, U. V. Luxburg, S. Bengio, H. Wallach, R. Fergus, S. Vishwanathan, and R. Garnett, editors, *Advances in Neural Information Processing Systems 30*, pages 5184–5194. Curran Associates, Inc., 2017.

Casey Chu, Kentaro Minami, and Kenji Fukumizu. The equivalence between Stein variational gradient descent and black-box variational inference. In *arXiv:2004.01822 [cs, stat]*, April 2020. arXiv: 2004.01822.

Marc Peter Deisenroth, Dieter Fox, and Carl Edward Rasmussen. Gaussian Processes for Data-Efficient Learning in Robotics and Control. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 37(2):408–423, February 2015.

A. Duncan, N. Nuesken, and L. Szpruch. On the geometry of Stein variational gradient descent. *arXiv:1912.00894 [cs, math, stat]*, December 2019. arXiv: 1912.00894.

Alan E. Gelfand, Peter Diggle, Peter Guttorp, and Montserrat Fuentes. *Handbook of spatial statistics*. Chapman & Hall/CRC handbooks of modern statistical methods. CRC Press, Boca Raton, Fla., 2010. ISBN 978-1-4200-7287-7.

Chengyue Gong, Jian Peng, and Qiang Liu. Quantile Stein Variational Gradient Descent for Batch Bayesian Optimization. In *International Conference on Machine Learning*, pages 2347–2356, May 2019.

Jackson Gorham and Lester Mackey. Measuring Sample Quality with Stein's Method. *Proceedings of the 34th International Conference on Machine Learning*, 70, December 2017.

Will Grathwohl, Kuan-Chieh Wang, Jorn-Henrik Jacobsen, David Duvenaud, and Richard Zemel. Cutting out the Middle-Man: Training and Evaluating Energy-Based Models without Sampling. *arXiv:2002.05616 [cs, stat]*, February 2020. arXiv: 2002.05616.

Marton Havasi, José Miguel Hernández-Lobato, and Juan José Murillo-Fuentes. Inference in Deep Gaussian Processes using Stochastic Gradient Hamiltonian Monte Carlo. *arXiv:1806.05490 [cs, stat]*, June 2018. arXiv: 1806.05490.

James Hensman, Nicolo Fusi, and Neil D. Lawrence. Gaussian Processes for Big Data. *Uncertainty in Artificial Intelligence*, 2013. arXiv: 1309.6835.

James Hensman, Alexander G. de G. Matthews, Maurizio Filippone, and Zoubin Ghahramani. MCMC for Variationally Sparse Gaussian Processes. *Advances in Neural Information Processing Systems*, June 2015. arXiv: 1506.04000.

Mark Kac. *Probability and related topics in physical sciences*. Number Volume 1. A in Lectures in applied mathematics. Am. Mathematical Soc, Providence, RI, 2. printing edition, 1976. ISBN 978-0-8218-0047-8. OCLC: 223864829.

Heishiro Kanagawa, Wittawat Jitkrittum, Lester Mackey, Kenji Fukumizu, and Arthur Gretton. A Kernel Stein Test for Comparing Latent Variable Models. *arXiv*, July 2019.

Diederik P. Kingma and Jimmy Ba. Adam: A Method for Stochastic Optimization. *International Conference for Learning Representations*, 2015. arXiv: 1412.6980.

Haitao Liu, Yew-Soon Ong, Xiaobo Shen, and Jianfei Cai. When Gaussian Process Meets Big Data: A Review of Scalable GPs. *arXiv:1807.01065 [cs, stat]*, April 2019. arXiv: 1807.01065.

Qiang Liu and Dilin Wang. Stein Variational Gradient Descent: A General Purpose Bayesian Inference Algorithm. *Advances in Neural Information Processing Systems 29*, 2016.

Qiang Liu, Jason D. Lee, and Michael I. Jordan. A Kernelized Stein Discrepancy for Goodness-of-fit Tests and Model Evaluation. *arXiv:1602.03253 [stat]*, July 2016. arXiv: 1602.03253.

Yang Liu, Prajit Ramachandran, Qiang Liu, and Jian Peng. Stein Variational Policy Gradient. *arXiv:1704.02399 [cs]*, April 2017. arXiv: 1704.02399.

Alexander G. de G. Matthews, Mark van der Wilk, Tom Nickson, Keisuke Fujii, Alexis Boukouvalas, Pablo Le{\'o}n-Villagr{\'a}, Zoubin Ghahramani, and James Hensman. GPflow: A Gaussian Process Library using TensorFlow. *Journal of Machine Learning Research*, 18(40):1–6, 2017.

Thomas P. Minka. *A family of algorithms for approximate bayesian inference*. PhD Thesis, Massachusetts Institute of Technology, USA, 2001. AAI0803033.

Jonas Mockus. *Bayesian Approach to Global Optimization: Theory and Applications*. Springer, Dordrecht, 2012. ISBN 978-94-009-0909-0. OCLC: 851374758.

Iain Murray, Ryan Adams, and David MacKay. Elliptical slice sampling. In *Proceedings of the Thirteenth International Conference on Artificial Intelligence and Statistics*, pages 541–548, March 2010.

Mahdi Pakdaman Naeini, Gregory F. Cooper, and Milos Hauskrecht. Obtaining Well Calibrated Probabilities Using Bayesian Binning. *Proceedings of the AAAI Conference on Artificial Intelligence*, 2015:2901–2907, January 2015.

Trung V Nguyen and Edwin V Bonilla. Automated Variational Inference for Gaussian Process Models. In Z. Ghahramani, M. Welling, C. Cortes, N. D. Lawrence, and K. Q. Weinberger, editors, *Advances in Neural Information Processing Systems 27*, pages 1404–1412. Curran Associates, Inc., 2014.

Chris J. Oates, Mark Girolami, and Nicolas Chopin. Control functionals for Monte Carlo integration. *Journal of the Royal Statistical Society: Series B (Statistical Methodology)*, 79(3):695–718, 2017.

Manfred Opper and Cédric Archambeau. The Variational Gaussian Approximation Revisited. *Neural Computation*, 21(3):786–792, September 2008.

M. Blanca Palacios and Mark F. J. Steel. Non-Gaussian Bayesian Geostatistical Modeling. *Journal of the American Statistical Association*, 101(474):604–618, June 2006.

George Papamakarios, Eric Nalisnick, Danilo Jimenez Rezende, Shakir Mohamed, and Balaji Lakshminarayanan. Normalizing Flows for Probabilistic Modeling and Inference. *arXiv:1912.02762 [cs, stat]*, December 2019. arXiv: 1912.02762.

Yuchen Pu, Zhe Gan, Ricardo Henao, Chunyuan Li, Shaobo Han, and Lawrence Carin. VAE Learning via Stein Variational Gradient Descent. *Advances in Neural Information Processing Systems*, page 10, April 2017.

Joaquin Quiñonero-Candela and Carl Edward Rasmussen. A Unifying View of Sparse Approximate Gaussian Process Regression. *J. Mach. Learn. Res.*, 6:1939–1959, December 2005.

Carl Edward Rasmussen and Christopher K. I. Williams. *Gaussian processes for machine learning*. MIT Press, 2006. ISBN 978-0-262-18253-9. OCLC: ocm61285753.

Daniel Rudoy and Patrick J. Wolfe. Monte Carlo Methods for Multi-Modal Distributions. In *2006 Fortieth Asilomar Conference on Signals, Systems and Computers*, pages 2019–2023, October 2006. ISSN: 1058-6393.

Hugh Salimbeni and Marc Deisenroth. Doubly Stochastic Variational Inference for Deep Gaussian Processes. In *Advances in Neural Information Processing Systems 30*, pages 4588–4599. Curran Associates, Inc., 2017.

Edward Snelson and Zoubin Ghahramani. Sparse Gaussian Processes using Pseudo-inputs. *Advances in Neural Information Processing Systems 18*, pages 1257–1264, 2006.

Charles Stein. A bound for the error in the normal approximation to the distribution of a sum of dependent random variables. *Proceedings of the Sixth Berkeley Symposium on Mathematical Statistics and Probability, Volume 2: Probability Theory*, 1972.

Michalis Titsias. Variational Learning of Inducing Variables in Sparse Gaussian Processes. In *Artificial Intelligence and Statistics*, pages 567–574, April 2009.

C.K.I. Williams and D. Barber. Bayesian classification with Gaussian processes. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 20(12):1342–1351, December 1998.

Ding-Xuan Zhou. Derivative reproducing properties for kernel methods in learning theory. *Journal of Computational and Applied Mathematics*, 220(1):456–463, October 2008.

Supplementary Material

7 Particle repulsion

Reminding ourselves of the update step (5) of SVGD:

$$\hat{\hat{\phi}}_{\Lambda_t}(\boldsymbol{\lambda}) = \frac{1}{J}\sum_{j=1}^{J}\left[\underbrace{\kappa(\boldsymbol{\lambda}_t^j, \boldsymbol{\lambda})\nabla_{\boldsymbol{\lambda}} \log p(\boldsymbol{\lambda}_t^j)}_{\text{Attraction}} + \underbrace{\nabla_{\boldsymbol{\lambda}}\kappa(\boldsymbol{\lambda}_t^j, \boldsymbol{\lambda})}_{\text{Repulsion}}\right],$$

we demonstrate how particles are repelled from one another. If we take $\kappa(\cdot,\cdot)$ to be the radial basis function (a valid kernel when computing Kernel Stein Discrepancy (KSD) (Gorham and Mackey, 2017)), then we have

$$\kappa(\boldsymbol{\lambda}, \boldsymbol{\lambda}') = \exp\left(\frac{\|\boldsymbol{\lambda} - \boldsymbol{\lambda}'\|^2}{-\ell^2}\right) \tag{12}$$

whereby

$$\begin{aligned}\nabla_{\boldsymbol{\lambda}}\kappa(\boldsymbol{\lambda}, \boldsymbol{\lambda}') &= -\frac{2\boldsymbol{\lambda} - 2\boldsymbol{\lambda}'}{\ell^2}\exp\left(\frac{\|\boldsymbol{\lambda} - \boldsymbol{\lambda}'\|^2}{-\ell^2}\right) \\ &= -\frac{2(\boldsymbol{\lambda} - \boldsymbol{\lambda}')}{\ell^2}\kappa(\boldsymbol{\lambda}, \boldsymbol{\lambda}'). \end{aligned} \tag{13}$$

Should particles be densely clustered, then the resultant Gram matrix will be dense. This will lead to a larger quantity being computed upon evaluation of (13), compared to when particles are sufficiently far from one another.

8 Proof of Theorem 1

SVGD maps particles using $\boldsymbol{\lambda}_{t+1} = \mathcal{T}(\boldsymbol{\lambda}_t) = \boldsymbol{\lambda}_t + \epsilon\hat{\phi}(\boldsymbol{\lambda})$. Denote the corresponding mapping of densities by $q_{t+1} = T(q_t)$. We have that

$$\begin{aligned}\mathrm{KL}(q_{t+1}||p) - \mathrm{KL}(q_t||p) &= \mathrm{KL}(T(q_t)||p) - \mathrm{KL}(q_t||p) \\ &= \mathrm{KL}(q_t||T^{-1}(p)) - \mathrm{KL}(q_t||p) \\ &= \mathbb{E}_{\boldsymbol{\lambda}\sim q_t}\left[\log q_t(\boldsymbol{\lambda}) - \log T^{-1}(p)(\boldsymbol{\lambda})\right] - \mathbb{E}_{\boldsymbol{\lambda}\sim q_t}\left[\log q_t(\boldsymbol{\lambda}) - \log p(\boldsymbol{\lambda})\right] \\ &= \mathbb{E}_{\boldsymbol{\lambda}\sim q_t}\left[\log p(\boldsymbol{\lambda}) - \log T^{-1}(p)(\boldsymbol{\lambda})\right]. \end{aligned} \tag{14}$$

Under the change of variable formula for densities, we have

$$T^{-1}(p)(\boldsymbol{\lambda}) = p(\mathcal{T}(\boldsymbol{\lambda})) \cdot |\det(\nabla_{\boldsymbol{\lambda}}\mathcal{T}(\boldsymbol{\lambda}))|$$

which allows us to rewrite (14) as

$$\mathbb{E}_{\boldsymbol{\lambda}\sim q_t}\left[\log p(\boldsymbol{\lambda}) - \log p(\boldsymbol{\lambda} + \epsilon\hat{\phi}(\boldsymbol{\lambda})) - \mathrm{logdet}(\nabla_{\boldsymbol{\lambda}}\mathcal{T}(\boldsymbol{\lambda}))\right]. \tag{15}$$

Assuming that $\nabla_{\boldsymbol{\lambda}} \log p(\boldsymbol{\lambda})$ is Lipschitz smooth with constant L and $\log p(\boldsymbol{\lambda}) \in C^2$, a second order Taylor series approximation of A about $\boldsymbol{\lambda}$ (assuming $\epsilon \ll 1$) lets us bound the first two terms in (15) by

$$\log p(\boldsymbol{\lambda}) - \log p(\boldsymbol{\lambda} + \epsilon\hat{\phi}(\boldsymbol{\lambda})) \leq -\epsilon\nabla_{\boldsymbol{\lambda}} \log p(\boldsymbol{\lambda})^\intercal \hat{\phi}(\boldsymbol{\lambda}) + \frac{L\epsilon^2}{2}\hat{\phi}(\boldsymbol{\lambda})^\intercal \hat{\phi}(\boldsymbol{\lambda}).$$

Noting the definition of the Stein operator from (1), we have that $-\epsilon\nabla_{\boldsymbol{\lambda}} \log p(\boldsymbol{\lambda})^\intercal \hat{\phi}(\boldsymbol{\lambda}) = \text{trace}(-\epsilon\mathcal{A}_p\hat{\phi}(\boldsymbol{\lambda}) - \epsilon\nabla_\lambda\hat{\phi}(\boldsymbol{\lambda}))$. Note also that $\mathcal{T}(\boldsymbol{\lambda}) = \boldsymbol{\lambda} + \epsilon\hat{\phi}(\boldsymbol{\lambda})$, and therefore $\nabla_{\boldsymbol{\lambda}}\mathcal{T}(\boldsymbol{\lambda}) = I + \epsilon\nabla_{\boldsymbol{\lambda}}\hat{\phi}(\boldsymbol{\lambda})$, which gives

$$\text{logdet}(\nabla_{\boldsymbol{\lambda}}\mathcal{T}(\boldsymbol{\lambda})) = \text{logdet}(I + \epsilon\nabla_{\boldsymbol{\lambda}}\hat{\phi}(\boldsymbol{\lambda})) \leq \epsilon\sum_{i=1}^{d} e_i = \epsilon\nabla_{\boldsymbol{\lambda}}\hat{\phi}(\boldsymbol{\lambda}), \tag{16}$$

where $e_1, \ldots, e_d$ are the eigenvalues of $\nabla_{\boldsymbol{\lambda}}\hat{\phi}(\boldsymbol{\lambda})$. Using these identities, (15) becomes

$$\underbrace{-\epsilon\mathbb{E}_{\boldsymbol{\lambda}\sim q_t}\left[\mathcal{A}_p\hat{\phi}(\boldsymbol{\lambda})\right]}_{B} \underbrace{- \mathbb{E}_{\boldsymbol{\lambda}\sim q_t}\left[\underbrace{2\epsilon\nabla_{\boldsymbol{\lambda}}\hat{\phi}(\boldsymbol{\lambda})}_{C1} + \underbrace{\frac{L}{2}(\epsilon\hat{\phi}(\boldsymbol{\lambda}))^2}_{C2}\right]}_{C}. \tag{17}$$

By definition of the Stein discrepancy (2), $B = -\epsilon\mathbb{D}(q_t, p)^2$, and (14) becomes

$$\text{KL}(q_{t+1}||p) - \text{KL}(q_t||p) \leq -\epsilon\mathbb{D}(q_t, p)^2 + C. \tag{18}$$

Based on this, we must now show that C is bounded, which we can do by considering each term individually.

C2: We can bound this term using the properties of the RKHS (Berlinet and Thomas-Agnan, 2004). As $\hat{\phi} = (\hat{\phi}_1, \ldots, \hat{\phi}_d)'$ and $\hat{\phi}_i \in \mathcal{H}_0 \implies \hat{\phi} \in \mathcal{H}^d$ then

$$\begin{aligned}
\left\|\hat{\phi}(\boldsymbol{\lambda})\right\|_2^2 &= \sum_{i=1}^{d} \hat{\phi}_i(\boldsymbol{\lambda})^2 \\
&= \sum_{i=1}^{d} \left(\langle\hat{\phi}_i(\cdot), \kappa(\boldsymbol{\lambda}, \cdot)\rangle_{\mathcal{H}_0}\right)^2 \text{ which follows from the RKHS properties} \\
&\leq \sum_{i=1}^{d} \left\|\hat{\phi}\right\|_{\mathcal{H}_0}^2 \|\kappa(\boldsymbol{\lambda}, \cdot)\|_{\mathcal{H}_0}^2 \text{ by Cauchy-Schwarz} \\
&= \left\|\hat{\phi}\right\|_{\mathcal{H}^d}^2 \kappa(\boldsymbol{\lambda}, \boldsymbol{\lambda}) \\
&= \mathbb{D}(q_t, p)^2\kappa(\boldsymbol{\lambda}, \boldsymbol{\lambda}) \text{ which follows by (3)}
\end{aligned} \tag{19}$$

C1: We upper bound $2\epsilon\nabla_{\boldsymbol{\lambda}}\hat{\phi}(\boldsymbol{\lambda})$ with the matrix norm $2\epsilon||\nabla_{\boldsymbol{\lambda}}\hat{\phi}(\boldsymbol{\lambda})||_F$ and the same RKHS property used in C2.

$$
\begin{aligned}
\left\|\nabla_{\boldsymbol{\lambda}}\hat{\phi}(\boldsymbol{\lambda})\right\|_F &= \sqrt{\sum_{i=1}^{d}\sum_{j=1}^{d}\left(\frac{\partial\hat{\phi}_i(\boldsymbol{\lambda})}{\partial\boldsymbol{\lambda}_j}\right)^2} \quad \text{from definition above and the Frobenius norm} \\
&\leq \sum_{i=1}^{d}\sum_{j=1}^{d}\left(\frac{\partial\hat{\phi}_i(\boldsymbol{\lambda})}{\partial\boldsymbol{\lambda}_j}\right)^2 \\
&= \sum_{i=1}^{d}\sum_{j=1}^{d}\left(\langle\hat{\phi}_i(\cdot), \partial\kappa(\boldsymbol{\lambda},\cdot)/\partial\boldsymbol{\lambda}_j\rangle_{\mathcal{H}_0}\right)^2 \quad \text{by Theorem 1 of Zhou (2008)} \\
&\leq \sum_{i=1}^{d}\sum_{j=1}^{d}\left\|\hat{\phi}_i\right\|_{\mathcal{H}_0}^2 \|\partial\kappa(\boldsymbol{\lambda},\cdot)/\partial\boldsymbol{\lambda}_j\|_{\mathcal{H}_0}^2 \quad \text{by Cauchy-Schwarz} \\
&= \left\|\hat{\phi}\right\|_{\mathcal{H}^d}^2 \nabla_{\boldsymbol{\lambda},\boldsymbol{\lambda}'}\kappa(\boldsymbol{\lambda},\boldsymbol{\lambda}') \\
&= \mathbb{D}(q_t,p)^2\nabla_{\boldsymbol{\lambda},\boldsymbol{\lambda}'}\kappa(\boldsymbol{\lambda},\boldsymbol{\lambda}').
\end{aligned}
$$

Finally, putting terms C1 and C2 together, (18) now becomes

$$
\begin{aligned}
\mathrm{KL}(q_{t+1}||p) - \mathrm{KL}(q_t||p) &\leq -\epsilon\mathbb{D}(q_t,p)^2 + \frac{\epsilon^2 L}{2}\mathbb{D}(q_t,p)^2\mathbb{E}_{\boldsymbol{\lambda}\sim q_t}\left[\kappa(\boldsymbol{\lambda},\boldsymbol{\lambda})\right] \\
&\quad + 2\epsilon\mathbb{D}(q_t,p)^2\mathbb{E}_{\boldsymbol{\lambda}\sim q_t}\left[\nabla_{\boldsymbol{\lambda},\boldsymbol{\lambda}'}\kappa(\boldsymbol{\lambda},\boldsymbol{\lambda})\right] \\
&= -\epsilon\mathbb{D}(q_t,p)^2\left(1 - \mathbb{E}_{\boldsymbol{\lambda}\sim q_t}\left[\epsilon L\kappa(\boldsymbol{\lambda},\boldsymbol{\lambda})/2 + 2\nabla_{\boldsymbol{\lambda},\boldsymbol{\lambda}'}\kappa(\boldsymbol{\lambda},\boldsymbol{\lambda})\right]\right).
\end{aligned}
$$

9 Demo implementation

The accompanying code to this paper has been designed to integrate with GPFlow 2.0, and can be run through the following commands.

```
from steingp import SteinGPR, RBF, Median, SVGD
import numpy as np
import gpflow

# Build data
X = np.random.uniform(-5, 5, 100).reshape(-1,1)
y = np.sin(x)

# Define model
kernel = gpflow.kernels.SquaredExponential()
model = SteinGPR((X, y), kernel)

# Fit
opt = SVGD(RBF(bandwidth=Median()), n_particles=5)
opt.run(iterations = 1000)

# Predict
Xtest = np.linspace(-5, 5, 500).reshape(-1, 1)
theta = opt.get_particles()
posterior_samples = model.predict(Xtest, theta, n_samples=5)
```

10 Training details

GP kernels For the regression and classification experiments in Section 5.1 and 5.2, an ARD RBF kernel was used, while a Matern32 kernel was used for sparse experiments in Section 5.3. In all experiments, the kernel's lengthscale for each dimension was initialised at the square root of the data's dimensionality. Further, the kernel's variance was initialised to equal 1.0.

SVGD kernels In all experiments we use an RBF kernel to compute (5). The kernel's lengthscale is estimated at each iteration of the optimisation procedure using the median rule, as per Liu and Wang (2016).

Likelihoods For the Gaussian likelihood functions used in Section 5.1, the variance parameter was initialised to 1.0. The same Gaussian likelihood is used in 5.3 but no initialisation is required due to the use of priors.

Particle initialisation The number of particles used in the SVGD scheme is explicitly reported in the respective results. Particle initialisation is carried out by making a random draw from the respective parameter's prior distributions where applicable, otherwise a draw is made from the uniform distribution on [0,1].

Prior distributions For GP models fitted using either HMC or SVGD we place the same priors on all parameters. Unless explicitly specified, a Gamma distribution parameterised with a unit shape parameter and a scale parameter of 2 is used as the prior for all lengthscale and variance parameters.

Parameter constraints For all parameters where positivity is a constraint (i.e. variance), the softplus transformation is applied with a clipping of 10^{-6}, as is the default in GPFlow. Optimisation is then conducted on the constrained parameter, however, we report the re-transformed parameter i.e. the unconstrained representation.

Optimisation For all variational models, natural gradients are used to optimise the variational parameters with a stepsize of 0.1. For the kernel hyperparameters and likelihood parameters (where applicable) in both the variational models and models fitted using maximum likelihood, the Adam optimiser (Kingma and Ba, 2015) was used. In Section 5.1 a step-size parameter of 0.01 is used, instead of the default recommendation of 0.001, as this was found to give faster optimisation at no detriment to the model's predictive accuracy. However, for the classification experiments in Section 5.2 we found that using the recommended stepsize of 0.001 was necessary for convergence.

Data availability All datasets used in Section 5.1 and 5.2 are available at `https://github.com/RedactedForReview`. The air quality data used in Section 5.3 was gathered using the `openaiR` package (Carslaw and Ropkins, 2012) using a script available at `https://github.com/RedactedForReview`.

11 Dataset descriptions

Table 4: The number of observations and corresponding dimensionality of each dataset used in Section 5.1

Dataset	N	Dimension
airfoil	1503	5
autompg	392	7
boston	506	13
challenger	23	4
concrete	1030	8
concreteslump	103	7
gas	2565	128
machine	209	7
parkinsons	5875	20
servo	167	4
skillcraft	3338	19
winered	1599	11
winewhite	4898	11

Table 5: The number of observations, observation dimensionality, and proportion of positive labels in each dataset used in Section 5.2

Dataset	N	Dimension	Positive proportion
breast-cancer	286	10	29.72%
blood	748	5	23.8%
mammographic	961	6	46.31%
spectf	267	45	79.4%
hepatitis	155	20	79.35%
fertility	100	10	12.0%

12 Additional experimental results

Table 6: Full set of test log-likelihood values for the datasets used in Section 5.1.

dataset	SteinGP2	SteinGP5	SteinGP10	SteinGP20	VI	ML
airfoil	$\mathbf{0.06 \pm 0.04}$	0.06 ± 0.04	0.05 ± 0.06	0.05 ± 0.05	0.03 ± 0.03	0.03 ± 0.03
autompg	-0.39 ± 0.09	-0.39 ± 0.09	-0.39 ± 0.09	-0.4 ± 0.09	$\mathbf{-0.39 \pm 0.07}$	-0.39 ± 0.07
boston	-0.3 ± 0.12	$\mathbf{-0.28 \pm 0.11}$	-0.3 ± 0.12	-0.3 ± 0.13	-0.31 ± 0.13	-0.31 ± 0.13
challenger	-1.53 ± 0.45	-1.52 ± 0.43	$\mathbf{-1.46 \pm 0.32}$	-1.53 ± 0.41	-1.51 ± 0.3	-1.51 ± 0.3
concrete	-0.25 ± 0.07	-0.25 ± 0.07	-0.25 ± 0.07	-0.25 ± 0.07	$\mathbf{-0.24 \pm 0.05}$	-0.24 ± 0.05
concreteslump	$\mathbf{1.08 \pm 0.39}$	1.07 ± 0.41	1.06 ± 0.4	1.08 ± 0.39	0.13 ± 1.14	0.13 ± 1.14
gas	0.88 ± 0.11	0.88 ± 0.11	$\mathbf{0.89 \pm 0.1}$	0.88 ± 0.11	0.79 ± 0.11	0.79 ± 0.11
machine	$\mathbf{-0.51 \pm 0.09}$	-0.52 ± 0.08	-0.52 ± 0.08	-0.52 ± 0.08	-0.52 ± 0.07	-0.52 ± 0.07
parkinsons	4.12 ± 0.05	4.12 ± 0.05	$\mathbf{4.14 \pm 0.03}$	4.13 ± 0.06	3.95 ± 0.04	3.95 ± 0.04
servo	-0.48 ± 0.04	-0.43 ± 0.05	-0.41 ± 0.11	-0.41 ± 0.21	-0.39 ± 0.1	$\mathbf{-0.39 \pm 0.1}$
skillcraft	$\mathbf{-0.99 \pm 0.02}$	-0.99 ± 0.02	-0.99 ± 0.02	-0.99 ± 0.02	-1.01 ± 0.02	-1.01 ± 0.02
winered	-1.17 ± 0.03	-1.17 ± 0.03	-1.17 ± 0.03	-1.17 ± 0.03	$\mathbf{-1.16 \pm 0.03}$	-1.16 ± 0.03
winewhite	0.56 ± 0.05	0.57 ± 0.05	$\mathbf{0.57 \pm 0.05}$	0.57 ± 0.05	0.49 ± 0.05	0.55 ± 0.05

Table 7: Computational runtimes reported in seconds for each model assessed in Section 5.1.

dataset	SteinGP2	SteinGP5	SteinGP10	SteinGP20	VI	ML
airfoil	39.96 ± 1.25	74.33 ± 2.51	129.81 ± 5.36	240.74 ± 2.93	61.48 ± 5.11	$\mathbf{21 \pm 1.87}$
autompg	$\mathbf{10.09 \pm 1.49}$	19.91 ± 3.53	36.11 ± 10.17	63.93 ± 8.31	29.02 ± 0.46	11.97 ± 0.21
boston	24.5 ± 1.7	49.22 ± 7.06	75.34 ± 7.17	151.41 ± 22.14	40.29 ± 1.81	$\mathbf{14.05 \pm 0.64}$
challenger	9.58 ± 0.31	16.40 ± 0.5	28.42 ± 0.38	52.7 ± 0.63	8.4 ± 0.03	$\mathbf{2.55 \pm 0.07}$
concrete	$\mathbf{9.67 \pm 1.18}$	65.21 ± 0.97	85.99 ± 21.05	93.73 ± 29.91	39.82 ± 0.81	13.89 ± 0.33
concreteslump	32.37 ± 1.91	56.66 ± 5.66	95.8 ± 10.16	175.98 ± 14.09	35.92 ± 22.12	$\mathbf{12.51 \pm 8.01}$
gas	$\mathbf{52.82 \pm 0.31}$	107.62 ± 1.29	199.79 ± 2.27	384.25 ± 3.74	245.96 ± 8.47	78.63 ± 3.21
machine	18.23 ± 1.5	29.56 ± 2.34	50.07 ± 3.13	97.68 ± 7.14	27.98 ± 0.93	$\mathbf{9.7 \pm 0.34}$
parkinsons	$\mathbf{201.64 \pm 1.14}$	468.34 ± 1.49	908.08 ± 2.13	1783.14 ± 4.22	2926.05 ± 3.04	675.27 ± 1.47
servo	$\mathbf{6.98 \pm 0.73}$	42.76 ± 21.21	38.15 ± 23.22	61.65 ± 15.05	23.94 ± 1.52	8.21 ± 0.56
skillcraft	68.63 ± 1.24	145.9 ± 0.67	273.3 ± 0.57	530.47 ± 1.92	190.17 ± 5.35	$\mathbf{53.96 \pm 1.38}$
wine	$\mathbf{15.73 \pm 0.77}$	61.99 ± 15.93	69.02 ± 7.51	100.62 ± 9.2	48.45 ± 4.25	17.46 ± 1.31
winered	16.93 ± 3.07	46.44 ± 8.93	55.55 ± 8.86	95.14 ± 16.87	28.56 ± 2.52	$\mathbf{9.83 \pm 0.91}$
winewhite	$\mathbf{142.04 \pm 0.59}$	320.33 ± 1.46	616.94 ± 1.08	1210.34 ± 1.65	1232.24 ± 7.84	315.16 ± 1.63

Table 8: Computational runtime in seconds for a variational GP (VI), GP fitted using HMC, and a SteinGP with 2, 5, 10 and 20 particles. Results are averaged over 5 random partitions of the data.

dataset	SteinGP2	SteinGP5	SteinGP10	SteinGP20	VI	HMC
blood	57.71 ± 0.26	111.41 ± 2.06	202.47 ± 2.35	381.57 ± 2.89	33.19 ± 1.42	171.49 ± 0.67
breast-cancer	12 ± 0.49	24.01 ± 1.27	38.3 ± 0.68	70.21 ± 0.9	8.82 ± 0.07	141.35 ± 0.71
fertility	9.91 ± 0.37	18.9 ± 0.45	33.2 ± 0.62	60.82 ± 1.8	8.71 ± 0.09	141.6 ± 0.42
hepatitis	13.4 ± 0.47	25.84 ± 0.99	44.56 ± 1.06	81.65 ± 2.06	8.78 ± 0.05	141.44 ± 0.63
mammographic	57.69 ± 0.59	111.91 ± 1.67	206.66 ± 4.12	391.31 ± 5.97	42.78 ± 2.19	190.2 ± 0.75
spectf	16.09 ± 0.56	30.87 ± 0.97	53.99 ± 1.49	102.61 ± 3.61	9.16 ± 0.23	142.41 ± 0.52

12.1 AURN MCMC Chains

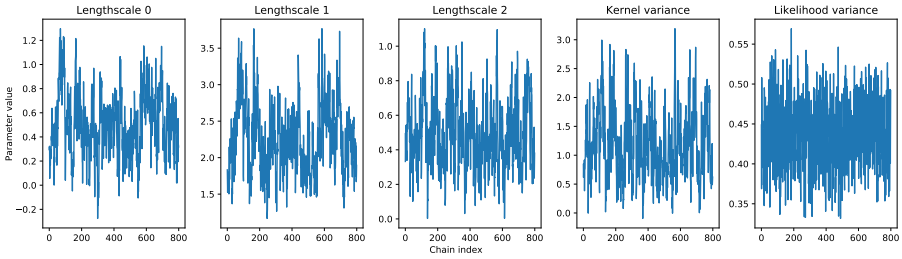

Figure 3: MCMC chains for the latitudinal, longitudinal and temporal lengthscales, and the kernel and likelihood variance parameters used in the sparse examples of Section 5.3.

12.2 Additional AURN results

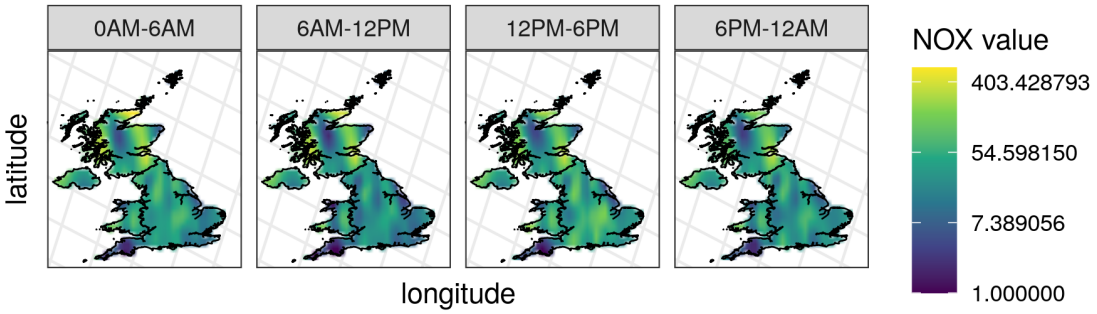

Figure 4: Spatiotemporal predictions for February 3rd. Each panel from left to right shows a mean prediction from 6 hour chronological increments across the day from a HMC sparse GP.

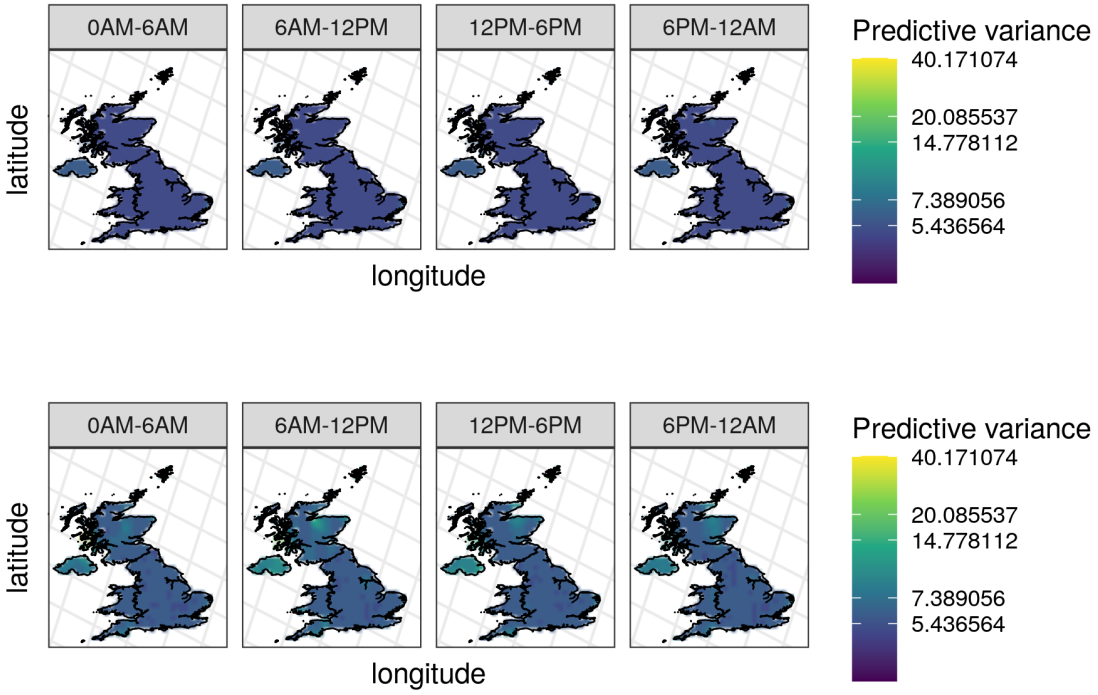

Figure 5: The predictive uncertainties from the sparse GP methods demonstrated in Section 5.3 using 800 inducing points. The upper row shows uncertainties arising from a HMC approach, and lower row corresponds to uncertainties from the SteinGP.