Sparse within Sparse Gaussian Processes using Neighbor Information

Gia-Lac Tran [1 2] **Dimitrios Milios** [1] **Pietro Michiardi** [1] **Maurizio Filippone** [1]

Abstract

Approximations to Gaussian processes (GPs) based on inducing variables, combined with variational inference techniques, enable state-of-the-art sparse approaches to infer GPs at scale through mini-batch based learning. In this work, we further push the limits of scalability of sparse GPs by allowing large number of inducing variables without imposing a special structure on the inducing inputs. In particular, we introduce a novel hierarchical prior, which imposes sparsity on the set of inducing variables. We treat our model variationally, and we experimentally show considerable computational gains compared to standard sparse GPs when sparsity on the inducing variables is realized considering the nearest inducing inputs of a random mini-batch of the data. We perform an extensive experimental validation that demonstrates the effectiveness of our approach compared to the state-of-the-art. Our approach enables the possibility to use sparse GPs using a large number of inducing points without incurring a prohibitive computational cost.

1. Introduction

Gaussian Processes (GPs) (Rasmussen & Williams, 2006) offer a powerful framework to perform inference over functions; being Bayesian, GPs provide rigorous uncertainty quantification and prevent overfitting. However, the applicability of GPs on large datasets is hindered by their computational complexity of $\mathcal{O}\left(N^3\right)$, where N is the training size. This issue has fuelled a considerable amount of research towards scalable GP methodologies that operate on a set of *inducing variables* (Quiñonero Candela & Rasmussen, 2005). In the literature, there is a plethora of approaches that offer different treatments of the inducing variables (Lawrence et al., 2002; Seeger et al., 2003; Snelson & Ghahramani, 2005; Naish-Guzman & Holden, 2007; Titsias, 2009; Hensman et al., 2013; Wilson & Nickisch, 2015; Hensman et al., 2015). Some of the more recent approaches, such as Scalable Variational Gaussian Processes (SVGPs) (Hensman et al., 2015), allow for the application of GPs to problems with millions of data-points. In most applications of scalable GPs, these are approximated using M inducing points, which results in a complexity of $\mathcal{O}\left(M^3\right)$. It has been shown recently by Burt et al. (2019) that it is possible to obtain an arbitrarily good approximation for a certain class of GP models (i.e. conjugate likelihoods, concentrated distribution for the training data) with M growing more slowly than N. However, the general case remains elusive and it is still possible that the required value for M may exceed a certain computational budget. Our result contributes to strengthen our belief that sparsity does not only enjoy desirable theoretical properties, but it also constitutes an extremely computationally efficient method in practice.

In this work, we push the limits of scalability and effectiveness of sparse GPs enabling a further reduction in complexity, which can be translated to higher accuracy by considering a larger set of inducing variables. The idea is to operate on a subset of H inducing points during training and prediction, with $H \ll M$, while maintaining a sparse approximation with M inducing variables. We formalize our strategy by imposing a sparsity-inducing structure on the prior over the inducing variables and by carrying out a variational formulation of this model. This extends the original SVGP framework and enables mini-batch based optimization for the variational objective. We then consider ways to select the set of H inducing points based on neighbor information; at training time, for a given mini-batch, we activate H out of M inducing variables considering the nearest inducing inputs to the samples in the mini-batch, whereas at test time we select inducing variables corresponding to the inducing inputs which are nearest to the test data-points. We name our proposal *Sparse within a Sparse* GP (SWSGP). SWSGP is characterized by a number of attractive features: (i) it improves significantly the prediction quality using a small number of neighboring inducing inputs, and (ii) it accelerates the training phase, especially when the total number of inducing points becomes large. We extensively validate these properties on a variety of regression and classification tasks. We also showcase SWSGP on a large scale classifica-

[1]Department of Data Science, Eurecom, France [2]Department of Computer Science, National University of Singapore, Singapore. Correspondence to: Gia-Lac Tran <tranlac@nus.edu.sg>.

Proceedings of the 38^{th} International Conference on Machine Learning, PMLR 139, 2021.

tion problem with $M = 100,000$; we are not aware of other approaches that can handle such a large set of inducing inputs without imposing some special structure on them (e.g., grid) or without considering one-dimensional inputs.

Hierarchical priors are often applied in Bayesian modeling to achieve compression and to improve flexibility (Molchanov et al., 2017; Louizos et al., 2017). To the best of our knowledge, this work is the first to explore these ideas as means to sparsify the inducing set in sparse GPs.

2. Related Work and Background

Sparse GPs that operate on inducing inputs have been extensively studied in the last 20 years (Csató & Opper, 2002; Lawrence et al., 2002; Snelson & Ghahramani, 2005; Quiñonero Candela & Rasmussen, 2005; Naish-Guzman & Holden, 2007). Many attempts on sparse GPs specified inducing inputs by satisfying certain criteria that produce an informative set of inducing variables (Csató & Opper, 2002; Lawrence et al., 2002; Seeger et al., 2003). A different treatment has been proposed by Titsias (2009), which involves formulating the selection of inducing inputs as optimization of a variational lower bound to the marginal likelihood. The variational framework was later extended so that stochastic optimization can be admitted, thus improving scalability for regression (Hensman et al., 2013) and classification (Hensman et al., 2015), and to deal with large-dimensional inputs (Panos et al., 2018). All the aforementioned methodologies share a computational complexity of $\mathcal{O}\left(M^3\right)$. Although there have been some attempts in the literature to infer the appropriate number of inducing points as well as the inducing inputs (Pourhabib et al., 2014; Burt et al., 2019), a large number of inducing variables is desirable to improve posterior approximation. In this work, we present a methodology that builds on the SVGP framework (Hensman et al., 2015) and reduces its complexity, thus increasing the potential of sparse GP application on even larger datasets and with a larger set of inducing variables.

A different approach to scalable GPs was introduced by Wilson & Nickisch (2015), namely Kernel Interpolation for Scalable Structured GPs (KISS-GP). This line of work involves arranging a large number of inducing inputs into a grid structure; this allows one to scale to very large datasets by means of fast linear algebra. The applicability of KISS-GP on higher-dimensional problems has been addressed by Wilson et al. (2015) by means of low-dimensional projections. A more recent extension allows for a constant-time variance prediction using Lanczos methods (Pleiss et al., 2018). Our work takes a different approach by keeping the GP prior intact, and by imposing sparsity on the set of inducing variables.

The local approximation of GPs inspired by the the concept of divide-and-conquer is also a practical solution to implement scalable GPs (Kim et al., 2005; Urtasun & Darrell, 2008; Datta et al., 2016; Park & Huang, 2016; Park & Apley, 2017). More recently, Liu & Liu (2019) proposed an amortized variational inference framework where the nearest training points are selected for inference. In our work, we use neighbor information in a different way, by incorporating it in a certain hierarchical structure of the auxiliary variables through a variational scheme.

2.1. Scalable Variational Gaussian Processes

Consider a supervised learning problem with inputs $\mathbf{X} = (\mathbf{x}_1, \ldots, \mathbf{x}_N)^\top$ associated with labels $\mathbf{y} = (y_1, \ldots, y_N)^\top$. Given a set of latent variables $\mathbf{f} = (f_1, \ldots, f_N)^\top$, GP models assume that labels are stochastic realizations based on $\mathbf{f}$ and a likelihood function $p(\mathbf{y} \mid \mathbf{f})$. In SVGPs, the set of inducing points is characterized by inducing inputs $\mathbf{Z} = (\mathbf{z}_1, \ldots, \mathbf{z}_M)^\top$ and inducing variables $\mathbf{u} = (u_1, \ldots, u_M)^\top$. Regarding $\mathbf{f}$ and $\mathbf{u}$, we have the following joint prior:

$$p(\mathbf{f}, \mathbf{u}) = \mathcal{N}\left(0, \begin{bmatrix} \mathbf{K_X} & \mathbf{K_{X,Z}} \\ \mathbf{K_{Z,X}} & \mathbf{K_Z} \end{bmatrix}\right), \tag{1}$$

where $\mathbf{K_X}$, $\mathbf{K_Z}$ and $\mathbf{K_{X,Z}}$ are covariance matrices evaluated at the inputs indicated by the subscripts. The posterior over inducing variables is approximated by a variational distribution $q(\mathbf{u}) = \mathcal{N}(\mathbf{u} \mid \mathbf{m}, \mathbf{S})$, while keeping the exact conditional $p(\mathbf{f} \mid \mathbf{u})$ intact, that is $q(\mathbf{f}, \mathbf{u}) = p(\mathbf{f} \mid \mathbf{u})q(\mathbf{u})$. The variational parameters $\mathbf{m}$ and $\mathbf{S}$, as well as the inputs $\mathbf{Z}$, are optimized by maximizing a lower bound on the marginal likelihood $p(\mathbf{y} \mid \mathbf{X}) = \int p(\mathbf{y} \mid \mathbf{f})p(\mathbf{f} \mid \mathbf{X})d\mathbf{f}$. The lower bound on $\log p(\mathbf{y} \mid \mathbf{X})$ can be obtained by considering the form of $q(\mathbf{f}, \mathbf{u})$ above and by applying Jensen's inequality:

$$\mathrm{E}_{q(\mathbf{f})}\left[\log p(\mathbf{y} \mid \mathbf{f})\right] - \mathrm{KL}\left(q(\mathbf{u}) \,\|\, p(\mathbf{u})\right). \tag{2}$$

The approximate posterior $q(\mathbf{f})$ can be computed by integrating out $\mathbf{u}$: $q(\mathbf{f}) = \int q(\mathbf{u})\, p(\mathbf{f} \,|\mathbf{u})\, d\mathbf{u}$. Thanks to the Gaussian form of $q(\mathbf{u})$, $q(\mathbf{f})$ can be computed analytically:

$$q(\mathbf{f}) = \mathcal{N}\left(\mathbf{f} \mid \mathbf{Am},\ \mathbf{K_X} + \mathbf{A}\left(\mathbf{S} - \mathbf{K_Z}\right)\mathbf{A}\right), \tag{3}$$

where $\mathbf{A} = \mathbf{K_{X,Z}}\mathbf{K_Z^{-1}}$. When the likelihood factorizes over training points, the lower bound can be re-written as:

$$\textstyle\sum_{i=1}^{N} \mathrm{E}_{q(f_i)}\left[\log p(y_i \mid f_i)\right] - \mathrm{KL}\left(q(\mathbf{u}) \,\|\, p(\mathbf{u})\right). \tag{4}$$

Each term of the one-dimensional expectation of the log-likelihood can be computed by Gauss-Hermite quadrature for any likelihoods (and analytically for the Gaussian likelihood). The $\mathrm{KL}\left(q(\mathbf{u}) \,\|\, p(\mathbf{u})\right)$ term can be computed analytically given that $q(\mathbf{u})$ and $p(\mathbf{u})$ are both Gaussian. To maintain positive-definiteness of $\mathbf{S}$ and perform unconstrained optimization, $\mathbf{S}$ is parametrized as $\mathbf{S} = \mathbf{L}\mathbf{L}^T$, with $\mathbf{L}$ lower triangular.

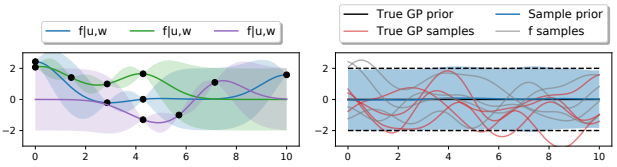


Figure 1. The choice of inducing points does not affect the prior samples drawn from $p(\mathbf{f})$. Left: visualizations of $\mathbf{f} \mid \mathbf{u}, \mathbf{w}$ for different samples of $\mathbf{w}$. Right: comparison of the marginalized (w.r.t. $\mathbf{u}, \mathbf{w}$) prior over $\mathbf{f}$, against the true $p(\mathbf{f})$.

3. Sparse Within Sparse GP

We present a novel formulation of sparse GPs, which permits the use of a random subset of the inducing points with little loss in performance. We introduce a set of binary random variables $\mathbf{w} \in \{0,1\}^M$ to govern the inclusion of inducing inputs $\mathbf{Z}$ and the corresponding variables $\mathbf{u}$. We then employ these random variables to define a hierarchical structure on the prior as follows:

$$p(\mathbf{u} \mid \mathbf{w}) = \mathcal{N}(\mathbf{0}, \mathbf{D_w K_Z D_w}), \tag{5}$$

where $\mathbf{D_w} = \operatorname{diag}(\mathbf{w})$, and $\mathbf{w} \sim p(\mathbf{w})$. Although the marginalized prior $p(\mathbf{u})$ is not Gaussian, it is possible to use the joint $p(\mathbf{u}, \mathbf{w}) = p(\mathbf{u} \mid \mathbf{w})\, p(\mathbf{w})$ within a variational scheme. We thus consider a random subset of the inducing points during the evaluation of the prior in the variational scheme that follows; no inducing points are permanently removed. Regarding $p(\mathbf{w})$, we consider an implicit distribution: its analytical form is unknown, but we can draw samples from it. Later, we will consider $p(\mathbf{w})$ based on the nearest inducing inputs to random mini-batches of data.

Remarks on the prior over $\mathbf{f}$ Our strategy simply assumes a certain structure on the auxiliary variables, but it *has no effect* on the prior over $\mathbf{f}$; the latter remains unchanged. Let $\mathcal{I}$ and $\mathcal{J}$ bet the sets of indices such that $\mathbf{w}_\mathcal{I} = \mathbf{1}$ and $\mathbf{w}_\mathcal{J} = \mathbf{0}$. Given an appropriate ordering, the conditional $\mathbf{u} \mid \mathbf{w}$ is effectively the element-wise product $[\mathbf{u}_\mathcal{I}, \mathbf{u}_\mathcal{J}]^\top = \mathbf{u} \circ \mathbf{w}$. This reduces the variances and covariances of some elements of $\mathbf{u}$ to zero yielding a distribution of this form:

$$p(\mathbf{f}, \mathbf{u} \mid \mathbf{w}) = \mathcal{N}\left(\begin{bmatrix} \mathbf{0} \\ \mathbf{0} \\ \mathbf{0} \end{bmatrix}, \begin{bmatrix} \mathbf{K_X} & \mathbf{K_{X,Z_\mathcal{I}}} & \mathbf{0} \\ \mathbf{K_{Z_\mathcal{I},X}} & \mathbf{K_{Z_\mathcal{I}}} & \mathbf{0} \\ \mathbf{0} & \mathbf{0} & \mathbf{0} \end{bmatrix} \right) \tag{6}$$

The rows and columns of $\mathbf{u}_\mathcal{J}$ can simply be ignored. Regardless of the value of $\mathbf{w}$, the conditional $\mathbf{f}, \mathbf{u}_\mathcal{I} \mid \mathbf{w}$ is always a Gaussian marginal, as it is a subset of Gaussian variables. The marginalized $p(\mathbf{f}, \mathbf{u}) = \int p(\mathbf{f}, \mathbf{u} \mid \mathbf{w})\, p(\mathbf{w}) d\mathbf{w}$ is mixture of Gaussian densities, where the marginal over $\mathbf{f}$ is the same for every component of the mixture.

The effect on $\mathbf{f}$ is demonstrated in Figure 1, where we sample from the (non-Gaussian) marginalized prior $p(\mathbf{u})$ in two steps: first we consider an arbitrary random subset $\mathbf{u}_\mathcal{I}$, and then we sample from $p(\mathbf{u}_\mathcal{I}) \equiv p(\mathbf{u} \mid \mathbf{w})$. Finally, $\mathbf{f}$ samples are drawn from $p(\mathbf{f} \mid \mathbf{u}_\mathcal{I})$, which only involves the selected inducing variables $\mathbf{u}_\mathcal{I}$. Following Eq. (1), the conditional $\mathbf{f} \mid \mathbf{u}$ is normally-distributed with mean $\mathbf{m}_{\mathbf{f}|\mathbf{u}_\mathcal{I}} = \mathbf{K_{X,Z_\mathcal{I}}} \mathbf{K}_{\mathbf{Z}_\mathcal{I}}^{-1} \mathbf{u}_\mathcal{I}$ and covariance $\mathbf{S}_{\mathbf{f}|\mathbf{u}_\mathcal{I}} = \mathbf{K_X} - \mathbf{K_{X,Z_\mathcal{I}}} \mathbf{K}_{\mathbf{Z}_\mathcal{I}}^{-1} \mathbf{K_{Z_\mathcal{I},X}}$. These conditionals can be seen for different samples of $\mathbf{u}, \mathbf{w}$ in the left side of Figure 1, while in the right side we compare the marginalized prior over $\mathbf{f}$ against the true GP prior.

Of course, although the prior remains unchanged, that is not the case for the posterior approximation. It is well known that the choice of inducing inputs has an effect on the variational posterior (Titsias, 2009; Burt et al., 2019). Our choice to impose a hierarchical structure to the inducing variables through $\mathbf{w}$ effectively changes the model compared to SVGP, and we adapt the variational scheme accordingly.

3.1. Lower Bound on Marginal Likelihood

By introducing $\mathbf{u}, \mathbf{w}$ and using Jensen's inequality, the lower bound on $\log p(\mathbf{y})$ can be obtained as follows

$$\mathrm{E}_{q(\mathbf{u},\mathbf{w})} \log p(\mathbf{y} \mid \mathbf{u}, \mathbf{w}) - \mathrm{KL}\left(q(\mathbf{u}, \mathbf{w}) \,\|\, p(\mathbf{u}, \mathbf{w})\right), \tag{7}$$

where we choose the variational distribution q to reflect the hierarchical structure of the prior, i.e. $q(\mathbf{u}, \mathbf{w}) = q(\mathbf{u} \mid \mathbf{w})\, p(\mathbf{w})$. This choice enforces sparsity over the approximate posterior q; the variational parameters are shared among the conditionals $q(\mathbf{u} \mid \mathbf{w})$, for which we assume:

$$q(\mathbf{u} \mid \mathbf{w}) = \mathcal{N}(\mathbf{u} \mid \mathbf{D_w m}, \mathbf{D_w S D_w}) \tag{8}$$

By maximizing the variational bound, we aim to obtain a q that performs well under a sparsified inducing set. We continue by applying Jensen's inequality on $p(\mathbf{y} \mid \mathbf{u}, \mathbf{w})$, obtaining:

$$\log p(\mathbf{y} \mid \mathbf{u}, \mathbf{w}) \geq \mathrm{E}_{p(\mathbf{f}|\mathbf{u},\mathbf{w})} \log p(\mathbf{y} \mid \mathbf{f}) \tag{9}$$

The bound we describe in Equations (7) and (9) is the same as in SVGP, but with a different variational distribution. In our case, the variational distribution imposes sparsity for $\mathbf{u}$ by means of $\mathbf{w}$. We can now substitute (9) into (7), obtaining a bound where we expand $q(\mathbf{u}, \mathbf{w})$ as $q(\mathbf{u} \mid \mathbf{w})\, p(\mathbf{w})$. By making this assumption, we get the following evidence lower bound $\mathcal{L}_{\text{ELBO}}$:

$$\sum_{n=1}^{N} \mathrm{E}_{p(\mathbf{w})} \Big[\mathrm{E}_{q(\mathbf{u}|\mathbf{w})} \mathrm{E}_{p(f_n|\mathbf{u},\mathbf{w})} \log p(y_n \mid f_n) - \frac{1}{N} \mathrm{KL}\left(q(\mathbf{u} \mid \mathbf{w}) \,\middle\|\, p(\mathbf{u} \mid \mathbf{w}) \right) \Big] \tag{10}$$

Recall that $p(\mathbf{w})$ is implicit: although we do not make any particular assumptions about its analytical form, we can

draw samples from it. Using MC sampling from $p(\mathbf{w})$, we can obtain the approximation $\tilde{\mathcal{L}}_{\text{ELBO}}$:

$$\sum_{n=1}^{N}\Big[\mathrm{E}_{q(\mathbf{u}|\tilde{\mathbf{w}}^{(n)})}\mathrm{E}_{p(f_n|\mathbf{u},\tilde{\mathbf{w}}^{(n)})}\log p(y_n|f_n) - \frac{1}{N}\mathrm{KL}\left(q\left(\mathbf{u}\left|\tilde{\mathbf{w}}^{(n)}\right.\right)\middle\|p\left(\mathbf{u}\left|\tilde{\mathbf{w}}^{(n)}\right.\right)\right)\Big], \tag{11}$$

where $\tilde{\mathbf{w}}^{(n)}$ is sampled from $p(\mathbf{w})$.

Sampling from the set of inducing points. Recall that any sample $\tilde{\mathbf{w}}$ from $p(\mathbf{w})$ is a binary vector, i.e. $\mathbf{w} \in \{0,1\}^M$. In case all elements of $\mathbf{w}$ are set to one, our approach recovers the original SVGP with computational cost of $\mathcal{O}(M^3)$ coming from computing $p(f_n|\mathbf{u},\tilde{\mathbf{w}}=\mathbf{1})$ and $\mathrm{KL}(q(\mathbf{u}|\mathbf{w})\|p(\mathbf{u}|\mathbf{w}))$ in the ELBO. When any $\tilde{w}_i$ is set to zero, the entries of the i-th row and i-th column of the covariance matrix in $p(\mathbf{u}|\mathbf{w})$ and $q(\mathbf{u}|\mathbf{w})$ are zero. This means that the i-th variable becomes unnecessary, so we get rid of i-th row and column in these matrices, and also eliminate the i-th element in mean vectors of $q(\mathbf{u}|\mathbf{w})$ and $p(\mathbf{u}|\mathbf{w})$. This is equivalent to selecting a set of active inducing points in each training iteration.

3.2. H-nearest Inducing Inputs

Despite the fact that $p(\mathbf{w})$ is an implicit distribution, we have been able to define and calculate a variational bound, assuming we can sample from $p(\mathbf{w})$. We shall now describe our sampling strategy, which relies on neighbor information of random mini-batches.

We introduce $\mathbf{Z}_{\mathbf{x}}^H$ as the set of H-nearest inducing inputs. Intuitively, the prediction for an unseen data $\mathbf{x}$ using $\mathbf{Z}_{\mathbf{x}}^H$ is a good approximation of the prediction using all M inducing points, that is $\mathbf{Z}_{\mathbf{x}}^M$. This can be verified by looking at the predictive mean, which is expressed as a linear combination of kernel functions evaluated between training points and a test point, as in Eq. (3). The majority of the contribution is given by the inducing points with the largest kernel values, so we can use this as a criterion to establish whether an inducing input is "close" to an input vector (the effect of different kernels on the definition of nearest neighbors is explored in the supplement). With this intuition, $p(\mathbf{w})$ becomes a deterministic function $w(\mathbf{x})$ indicating which inducing inputs are activated. For mini-batch based training, the value of $\mathbf{w}$ remains random, as it depends on the elements $\mathbf{x}$ that are selected in the random mini-batch; this materializes the sampling from the implicit distribution $p(\mathbf{w})$. The maximization of the ELBO in the setting described is summarized in Algorithm 1 (SWSGP).

Predictive distribution Contrary to what happens at training time, where mini-batches of data are drawn randomly, at test time the inputs of interest are not random; we need to describe the predictive distribution in terms of the deterministic function $w(\mathbf{x})$. In fact, if we would like to approximate the predictive distribution at $\mathbf{x}_n$ using H-nearest inducing inputs to $\mathbf{x}$, i.e. $\mathbf{Z}_{\mathbf{x}_n}^H$, then $w(\mathbf{x}) = \left[w_{\mathbf{x}}^{(1)}...w_{\mathbf{x}}^{(M)}\right]^T$ where,

$$w_{\mathbf{x}}^{(m)} = \begin{cases} 1 & \text{if } \mathbf{z}_m \in \mathbf{Z}_{\mathbf{x}}^H \\ 0 & \text{else} \end{cases}, \text{ with } m = 1,...,M \tag{12}$$

We extract the relevant elements using $w(\mathbf{x})$; for the mean, we have $\mathbf{m}_{w(x_i)} = \mathbf{D}_{w(x_i)}\mathbf{m}$, and for the covariance we select the appropriate rows and columns using $\mathbf{S}_{w(\mathbf{x}_i)} = \mathbf{D}_{w(\mathbf{x}_i)}\mathbf{S}\mathbf{D}_{w(\mathbf{x}_i)}$. The approximate posterior over f_i given $w(\mathbf{x}_i)$, i.e. $q(f_i|w(\mathbf{x}_i))$ is:

$$\mathcal{N}\Big(f_i|\mathbf{A}_{\mathbf{x}_i}\mathbf{m}_{w(\mathbf{x}_i)}, \mathbf{K}_{\mathbf{x}_i} + \mathbf{A}_{\mathbf{x}_i}\left(\mathbf{S}_{w(\mathbf{x}_i)} - \mathbf{K}_{\mathbf{Z}_{\mathbf{x}_i}^H}\right)\mathbf{A}_{\mathbf{x}_i}^\top\Big), \tag{13}$$

where $\mathbf{A}_{\mathbf{x}_i} = \mathbf{K}_{\mathbf{x}_i,\mathbf{Z}_{\mathbf{x}_i}^H}\mathbf{K}_{\mathbf{Z}_{\mathbf{x}_i}^H}^{-1}$.

Algorithm 1 Sparse within sparse GP (SWSGP)

Require: $\mathcal{D}$, H, M.
Ensure: The optimum of trainable parameters $\boldsymbol{\theta}$.
Initialize $\boldsymbol{\theta}$ including kernel's parameters, $\mathbf{Z}$, $\mathbf{m}$ and $\mathbf{L}$ which can be used to construct $\mathbf{S}$, i.e. $\mathbf{S} = \mathbf{L}\mathbf{L}^T$.
while stopping criteria is False **do**
  ELL $\leftarrow 0$ and KL $\leftarrow 0$.
  Sample mini-batch B of size n_{B} from $\mathcal{D}$.
  for $(\mathbf{x}_i, y_i) \in$ B **do**
    Find $\mathbf{Z}_{\mathbf{x}_i}^H$, i.e. the H-nearest $\mathbf{Z}$ to $\mathbf{x}_i$.
    Compute $w(\mathbf{x}_i)$ using $\mathbf{Z}_{\mathbf{x}_i}^H$ as in (12).
    Extract $\mathbf{m}_{w(\mathbf{x}_i)}$ and $\mathbf{S}_{w(\mathbf{x}_i)}$ from $\mathbf{m}$ and $\mathbf{L}$.
    Compute $q(f_i|w(\mathbf{x}_i))$ as in (13).
    ELL $\leftarrow$ ELL $+ \mathrm{E}_{q(f_i|w(\mathbf{x}_i))}\log p(y_i|f_i)$.
    KL $\leftarrow$ KL $+ \mathrm{KL}\left(q\left(\mathbf{u}_{w(\mathbf{x}_i)}\right)\|p\left(\mathbf{u}_{w(\mathbf{x}_i)}\right)\right)$
  end for
  $\tilde{\mathcal{L}}_{\text{ELBO}} \leftarrow \frac{N}{n_{\text{B}}}\text{ELL} - \frac{1}{n_{\text{B}}}\text{KL}$.
  Update $\boldsymbol{\theta}$ using the derivative of $\tilde{\mathcal{L}}_{\text{ELBO}}$.
end while

Limitations What we presented allows one to compute predictive distributions for individual test points following Eq. (13). As a result, it is not possible to obtain a full covariance across predictions over multiple test points. While most performance and uncertainty quantification metrics do not consider covariances, this might be a limitation in applications where covariances are essential. One way to work around this is to consider the union of nearest neighbors at test time; however, this would be inconsistent with

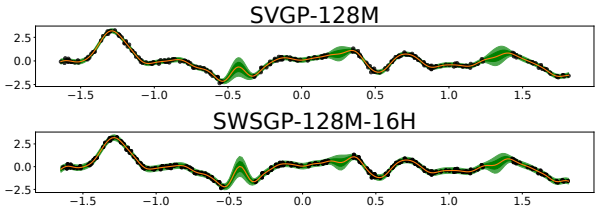


Figure 2. Visualization of posterior distribution of SVGP and SWSGP. In both cases, we consider 128 inducing points; in terms of our scheme (SWSGP) we use 16 neighbors.

Eq. (13) and the training procedure should be modified accordingly considering the union of nearest neighbors for training points within mini-batches. The considerations suggest a further limitation of SWSGP, that is that the predictive distribution becomes dependent on the number of test points that are consdered within a batch at test time. Nevertheless, the model is built and trained to deal with sparsity at test time, and this kind of inconsistency appears only when testing the model in conditions which are different compared to those at training time. Besides, the experiments show that SWSGP performs extremely well against various state-of-the-art competitors on a wide range of experiments.

One-dimensional regression example. We visualize the posterior distribution for a synthetic dataset generated on a one-dimensional input space. We execute SVGP and SWSGP, and depict the posterior distributions of these two methods by showing the predictive means (orange lines) and the 95% credible intervals (shaded areas) in Figure 2. We consider identical settings for the two methods (i.e. 128 inducing points, kernel parameters, likelihood variance) and a neighbor area of 16 for SWSGP; a full account of the setup can be found in the supplement. We see that although the models are different, the predictive distributions appear remarkably similar. A more extensive evaluation follows in Section 4.

3.3. Complexity

The computational cost of SWSGP is dominated by lines 6, 8 and 9 in Algorithm 1. For each data-point $(\mathbf{x}_i, y_i)$ in mini-batch B, we need to find the H nearest inducing neighbors $\mathbf{Z}^H_{\mathbf{x}_i}$ for n_B points in line 6, where $n_\mathrm{B} = |\mathrm{B}|$; this contributes to the worst-case complexity by $\mathcal{O}\left(n_\mathrm{B} M H\right)$.

In line 8, we extract relevant parameters from $\mathbf{m}$ and $\mathbf{L}$. We focus on the cost of extracting $\mathbf{S}_{w(\mathbf{x}_i)}$ from $\mathbf{L}$. Similar to SVGP (Section 2.1), we consider $\mathbf{S} = \mathbf{L}\mathbf{L}^T$, where $\mathbf{L}$ is lower triangular. We extract $\mathbf{L}_{w(\mathbf{x}_i)} = \mathbf{D}_{w(\mathbf{x}_i)}\mathbf{L}$ which contains the rows of $\mathbf{L}$. Then, we compute $\mathbf{S}_{w(\mathbf{x}_i)}$ by $\mathbf{S}_{w(\mathbf{x}_i)} = \mathbf{L}_{w(\mathbf{x}_i)}\mathbf{L}^T_{w(\mathbf{x}_i)}$. The computational complexity of selecting the variational parameters is $\mathcal{O}\left(n_\mathrm{B} M H^2\right)$.

Finally, the computation of approximating the predictive distribution in line 9 requires $\mathcal{O}\left(n_\mathrm{B} H^3\right)$. The overall complexity for SWSGP in the general case is $\mathcal{O}\left(n_\mathrm{B} M H + n_\mathrm{B} M H^2 + n_\mathrm{B} H^3\right)$, which is a significant improvement over the $\mathcal{O}\left(M^3\right)$ complexity of standard SVGP, assuming that $n_\mathrm{B}, H \ll M$. If we choose $\mathbf{S}$ to be diagonal, the total complexity reduces to $\mathcal{O}\left(n_\mathrm{B} M H + n_\mathrm{B} H^3\right)$; if we additionally consider $\mathbf{Z}$ to be fixed, the computational cost is $O(n_\mathrm{B} H^3)$. In the experiments of Section 4 we explore all these settings.

4. Experiments

We conduct experiments to evaluate SWSGP on a variety of experimental conditions. Our approach is denoted by SWSGP-M-H, where M inducing points are used and H determines how many neighbors are selected. The locations of the inducing points are optimized, unless stated otherwise. We introduce SVGP-M and SVGP-M-H as competitors; SVGP-M uses M inducing points. SVGP-M-H, instead, refers to SVGP using M inducing points at training time and H-nearest inducing inputs at test time. We also use SVGP-16M using 16 inducing points as a reference. The comparison is carried out on some UCI data sets for regression and classification, i.e., POWERPLANT, KIN, PROTEIN, EEG, CREDIT and SPAM. We also consider larger scale data sets, such as, WAVE, QUERY and the AIRLINE data or images classification on MNIST. The task for the AIRLINE data set is the classification of whether flights are to subject to a delay, and we follow the same setup as in Hensman et al. (2013) and Wilson et al. (2016). Regarding MNIST, we consider the binary classification setup of separating odd and even digits (pixels normalized between 0 and 1), and we use a Bernoulli likelihood with probit inverse link function. We use the Matérn-$^5/_2$ kernel in all cases except for the AIRLINE dataset, where the sum of a Matérn-$^3/_2$ and a linear kernel is used, similar to Hensman et al. (2015). All models are trained using the Adam optimizer (Kingma & Ba, 2015) with a learning rate of 0.001 and a mini-batch size of 64. The likelihood for regression and binary classification are set to Gaussian and probit function, respectively. In regression tasks, we report the test root mean squared error (RMSE) and the test mean negative log-likelihood (MNLL), whereas we report the test error rate (ERR) and MNLL in classification tasks. The results are averaged over 10 folds. All models in section 4.1 are trained over $300,000$ iterations.

4.1. Impact of M and H

We begin our evaluation by investigating the behavior of SWSGP-M-H with respect to M and H. Figure 3 shows that SWSGP-M-H consistently outperforms SVGP-M-H. This suggests that including neighbor information at prediction time, combined with the use of a larger set of inducing points alone is not enough to obtain competitive performance, and that only thanks to the sparsity-inducing prior over latent

variables, this yields improvements. Crucially, the performance metrics obtained by SWSGP are comparable with those obtained by SVGP-M, while at each iteration only a small subset of H out of M inducing points are updated, carrying a significant complexity reduction.

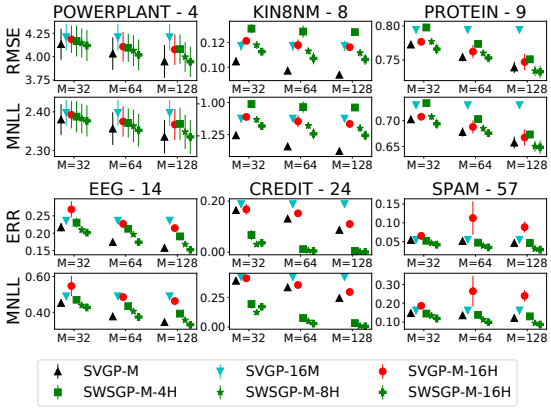


Figure 3. Evaluation of SWSGP on UCI data sets with various configuration for M and H. The title of each sub-figure follows the format of [name of data set]-[data dimensions]. The black up-triangles are for SVGP with M inducing points. The cyan down-triangles are for SVGP with 16 inducing points. The red circles are for SVGP training with M inducing points and the prediction at an unseen data $\mathbf{x}$ are made by $\mathbf{Z}_{\mathbf{x}}^{H}$. The green squares, stars and plus are for SWSGP with H of 4, 8 and 16 respectively. In these experiments, M varies from 32 to 128, as shown on horizontal axes. The standard deviation of the error metrics over 10 folds is represented by vertical bars; they are very small for most configurations.

4.2. Running Time

In Table 1, we report the training and testing times of SWSGP and SVGP In SVGP, we set $M = 256$ for EEG, and 1024 for MNIST, i.e. SVGP-256M and SVGP-1024M. In our approach, we use the same M and we set H to 4, i.e. SWSGP-256M-4H and SWSGP-1024M-4H. In Table 1, we stress that t_1 and t_2 in SWSGP take into account the computation of finding neighbors inducing inputs for each data-point. In SVGP, we assume that $\mathbf{K}_{\mathbf{Z}}^{-1}$ is pre-computed and saved after the training phase. Therefore, the computational cost to evaluate the predictive distribution on a single test point is $\mathcal{O}\left(M^2\right)$. The time t_2 in SVGP refers to the execution time of carrying out predictions with the complexity of $\mathcal{O}\left(M^2\right)$.

The results in Tab 1 show a consistent improvement at test time compared to SVGP across all values of H and M. At training time, the results show a trend dependent on the number M of inducing points. Not surprisingly, SWSGP offers limited improvements when M is small. Considering EEG in which M is set to 256, SVGP is faster than SWSGP in terms of training time. This is because the inversion of a 256×256 matrix requires less time than finding the neighbors and inverting several 4×4 matrices. However, Tab 1 shows dramatic speedups compared to SVGP when the number of inducing points M is large. When $M = 1024$ on MNIST, SWSGP-1024M-4H is faster than SVGP-1024M at training time. This is due to the inversion of the 1024×1024 kernel matrix being a burden for SVGP, whereas SWSGP deals with much cheaper computations. In addition, we show the corresponding ERR and MNLL of each model when we train SVGP and SWSGP on EEG and MNIST. On the EEG data set, our method is comparable with SVGP. On MNIST, SWSGP reaches high accuracies significantly faster compared to SVGP.

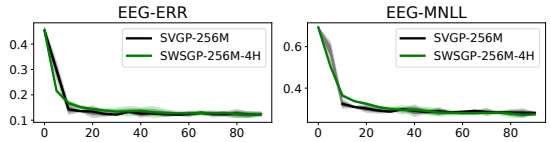
Table 1. Comparison of running time between SVGP and SWSGP. The running time for a training iteration is denoted by t_1. The testing time for an unseen example is denoted by t_2. Times are in milliseconds. The corresponding evaluated metrics ,i.e. ERR and MNLL are also shown in terms of training time. These figures are averaged over 5 folds. The models for EEG and MNIST are trained in 90 minutes and 6 hours respectively.

(a) Running times for SVGP-256M and SWSGP-256M-4H on EEG.

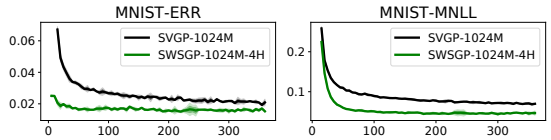
Configuration	t_1(ms)	t_2(ms)	ERR	MNLL
SVGP-256M	**21.42**	1.43	**0.17**	**0.34**
SWSGP-256M-4H	26.18	**0.56**	0.18	0.36

(b) Running times for SVGP-1024M and SWSGP-1024M-4H on MNIST.

Configuration	t_1(ms)	t_2(ms)	ERR	MNLL
SVGP-1024M	516	21.6	0.02	0.066
SWSGP-1024M-4H	**233**	**1.77**	**0.016**	**0.05**



(c) ERR and MNLL over training time (in minutes) on EEG.



(d) ERR and MNLL over training time (in minutes) on MNIST.

4.3. Large-scale Sparse GP Modeling with a Huge Number of Inducing Points

Here we show that SWSGP allows one to use sparse GPs with a massive number of inducing points without incurring a prohibitive computational cost. We employ several large-scale data sets, i.e. PROTEIN, EEG, WAVE, QUERY and

AIRLINE with 5 millions training samples. We test SWSGP with a large number of inducing points, ranging from $3,000$ to $100,000$. In this case, we keep the inducing locations fixed for SWSGP. We have attempted to run SVGP with such large values of M without success (out of memory in a system with 32GB of RAM). Therefore, as a baseline we execute SVGP-128M and SVGP-512M and report the results of SVGP with the configuration in Hensman et al. (2015).

In SWSGP, we impose a diagonal matrix $\mathbf{S}$ in the variational distribution $q(\mathbf{u} \mid \mathbf{w})$, and we fix the position of the inducing inputs during training. By fixing the inducing inputs, we can operate with pre-computed information about which inducing inputs are neighbors of training inputs. Thanks to these settings, SWSGP's training phase requires $\mathcal{O}\left(n_{\mathrm{B}} H^3\right)$ operations only, where n_{B} is the mini-batch size. Due to the appropriate choice of H and n_{B}, and the computational cost being independent of M, unlike SVGP, we can successfully run SWSGP with an unprecedented number of inducing point, e.g. $M = 100,000$.

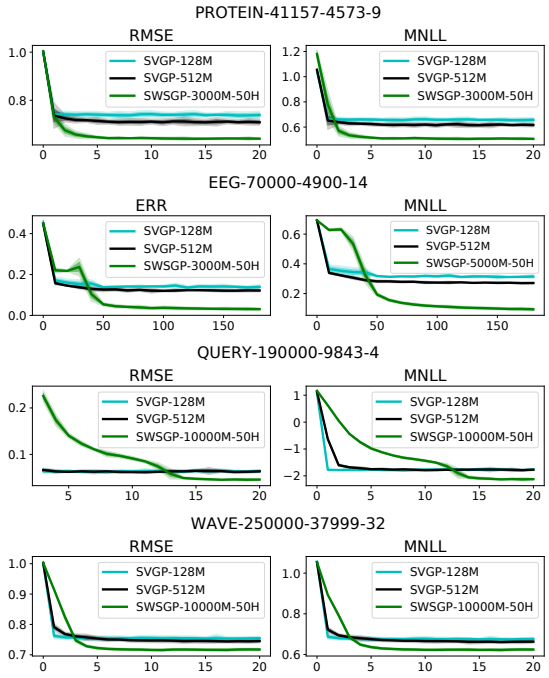


Figure 4. SWSGP with large number of inducing points. The figure shows the progression of RMSE (ERR) and MNLL over time. Horizontal axes indicate the running times in minute. The title of each sub-figure follows the format of [name of data set]-[training size]-[testing size]-[data dimensions].

In AIRLINE experiments, by setting H and the mini-batch size n to 100 and 64 respectively, in about 24 hours of training we could run SWSGP-100,000M-100H for one million iterations. The ERR and MNLL of SWSGP-100,000M-100H evaluated on the test set are 21% and 0.48, respectively, while the ERR and MNLL of SVGP-200M published in Hensman et al. (2015) are about 34% and 0.61, respectively. To the best of our knowledge, SWSGP is the first to enable sparse GPs with such a large set of inducing points without imposing a grid structure on the inducing inputs. In addition, in Figure 4 we also show that SWSGP using $3,000$ or $5,000$ or $10,000$ inducing points outperforms SVGP using 128 and 512 inducing points on other data sets.

We conclude by reporting comparisons with other GP-based models. In particular, we compare against the Stochastic Variational Deep Kernel Learning (SVDKL) (Wilson et al., 2016) and the Deep GP approximated with random features (DGP-RBF) (Cutajar et al., 2017). In the former, KISS-GP is trained on top of a deep neural network which is optimized during training, and in the latter the layers of a deep GP are approximated as parametric models using random feature expansions. Both competitors feature mini-batch based learning, so this represents a challenging test for SWSGP. The results in Table 2 show that SWSGP is comparable with these competitors on various data sets. We believe that this is a remarkable result obtained by our shallow SWSGP, supporting the conclusions of previous works showing that advances in kernel methods can result in performance which are competitive with deep learning approaches (see, e.g., Rudi et al. (2017)).

Table 2. Comparison of SWSGP, KISS-GP (Wilson & Nickisch, 2015), SVDKL (Wilson et al., 2016) and DGP-RBF (Cutajar et al., 2017). The results SWSGP and KISS-GP are averaged over 5 folds. In order to deal with the difficulties of KISS-GP to handle large-dimensional input spaces, we followed (Wilson et al., 2015), and we linearly projected the inputs to a two-dimensional space using a linear transformation that is learned at training time, and used a grid of size 100.

(a) AIRLINE

Method	ERR	MNLL
SWSGP-100kM-100H	$\mathbf{0.210 \pm 0.012}$	0.48 ± 0.015
SVDKL	0.22	$\mathbf{0.46}$
DGP-RBF	$\mathbf{0.21}$	$\mathbf{0.46}$

(b) POWERPLANT

Method	RMSE	MNLL
SWSGP-64M-4H	$\mathbf{4.095 \pm 0.145}$	$\mathbf{2.371 \pm 0.038}$
KISS-GP	4.459 ± 0.355	3.074 ± 0.037

(c) PROTEIN

Method	RMSE	MNLL
SWSGP-64M-4H	$\mathbf{0.773 \pm 0.002}$	$\mathbf{0.702 \pm 0.003}$
KISS-GP	0.816 ± 0.001	0.904 ± 0.017

4.4. Comparison to Local GPs

We finally demonstrate that SWSGP behaves differently from other approaches that use local approximations of GPs. We consider two well-established approaches of local GPs proposed by Kim et al. (2005) and Urtasun & Darrell (2008). Following Liu et al. (2020), we refer to these methods as *Inductive* GPs and *Transductive* GPs, respectively. We run all methods on two regression data sets: POWERPLANT and KIN. We set the number of local experts to 64, and we use the same number of inducing points for SWSGP (with H either 4 or 8). As the size of POWERPLANT and KIN are approximately $7,000$, we set the number of training points governed by a local expert to 100. For the local GP approaches, we choose 64 locations in the input space using the K-means algorithm, and for each location we choose 100 neighboring points; we then train the corresponding local GP expert. For the testing phase, inductive GPs simply rely on the nearest local experts to an unseen point $\mathbf{x}_*$. For transductive GPs, we use 100 neighbors of $\mathbf{x}_*$ and the nearest local expert to make predictions. In Table 3, we summarize RMSE and MNLL for all methods; SWSGP clearly outperforms the local GP approaches in terms of MNLL.

Table 3. Comparison with Local GP approximations.

Method	POWERPLANT RMSE | MNLL	KIN RMSE | MNLL
SWSGP-64-4	4.27 | 2.41	0.11 | −1.27
SWSGP-64-8	**4.24** | **2.40**	0.10 | **-1.38**
Inductive GPs	9.93 | 38.38	0.13 | −0.40
Transductive GPs	6.17 | 18.78	**0.09** | −0.65

4.5. Joint Predictive Covariances

In this section, we propose an extension of SWSGP where the training procedure is modified by considering the union of nearest neighbors for training points within mini-batches. We refer to this as SWSGP-U, where U stands for "union". In the experiments, we execute SWSGP-U-512M-128m where the total number of inducing points is 512, and the number of active inducing points in each training iteration is fixed to 128. To illustrate the effectiveness of the modification, we compare SWSGP-U-512M-128m against SVGP-512M and full GPs. Due to the computational complexity of full GPs, we only use $2,000$ training samples and fix the computational budget to six hours. In the evaluation phase, we randomly group the test set based on the specified mini-batch size. Next, the joint predictive distributions of each testing batch are obtained, which we use to compute RMSE and MNLL. The results in Figure 5 are averaged by repeating the process 10 times. The results indicate that SWSGP-U works well

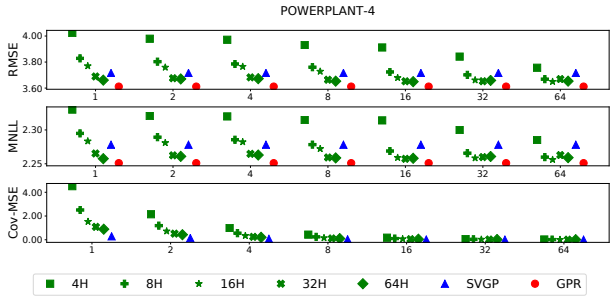


Figure 5. SWSGP-U on POWERPLANT. The horizontal axis shows the mini-batch size of the testing set. We evaluate SWSGP-U with various numbers of nearest inducing points from 4 to 64, i.e. $4H$, $8H$, $16H$, $32H$, and $64H$. The green markers show SWSGP-U. For example, in order to plot the green squares $4H$, we firstly find the 4-nearest inducing points of each sample within a testing batch. Then, we use the union of these nearest inducing points to make a joint prediction for all samples in the testing batch.

when using an adequate number of nearest inducing points. In the figure, we also assess the quality of the covariance approximation of SWSGP-U and SVGP by reporting the mean square error with respect to the covariance of full GPs. These results show that SWSGP-U provides a good approximation of joint predictive covariances.

5. Conclusions

Sparse approaches that rely on inducing points have met with success in reducing the complexity of GP regression and classification. However, these methods are limited by the number of inducing inputs that is required to obtain an accurate approximation of the true GP model. A large number of inducing inputs is often necessary in cases of very large datasets, which marks the limits of practical applications for most GP-based approaches.

In this work, we further improve the computational gains of sparse GPs. We proposed SWSGP, a novel methodology that imposes a hierarchical and sparsity-inducing effect on the prior over the inducing variables. This has been realized as a conditional GP given a random subset of the inducing points, which is defined as the nearest neighbors of random mini-batches of data. We have developed an appropriate variational bound which can be estimated in an unbiased way by means of mini-batches. We have performed an extensive experimental campaign that demonstrated the superior scalability properties of SWSGP compared to the state-of-the-art.

Acknowledgments MF gratefully acknowledges support from the AXA Research Fund and the Agence Nationale de la Recherche (grant ANR-18-CE46-0002 and ANR-19-P3IA-0002).

References

Burt, D., Rasmussen, C. E., and Van Der Wilk, M. Rates of convergence for sparse variational Gaussian process regression. In *Proceedings of the 36th International Conference on Machine Learning*, volume 97 of *Proceedings of Machine Learning Research*, pp. 862–871. PMLR, 2019.

Csató, L. and Opper, M. Sparse on-line gaussian processes. *Neural Computation*, 14(3):641–668, 2002. ISSN 0899-7667.

Cutajar, K., Bonilla, E. V., Michiardi, P., and Filippone, M. Random feature expansions for deep Gaussian processes. In *Proceedings of the 34th International Conference on Machine Learning*, volume 70 of *Proceedings of Machine Learning Research*, pp. 884–893. PMLR, 2017.

Datta, A., Banerjee, S., Finley, A., and Gelfand, A. On nearest-neighbor gaussian process models for massive spatial data: Nearest-neighbor gaussian process models. *Wiley Interdisciplinary Reviews: Computational Statistics*, 8, 08 2016.

Hensman, J., Fusi, N., and Lawrence, N. D. Gaussian processes for big data. In *Proceedings of the 29th Conference on Uncertainty in Artificial Intelligence*, pp. 282–290. AUAI Press, 2013.

Hensman, J., Matthews, A., and Ghahramani, Z. Scalable Variational Gaussian Process Classification. In *Proceedings of the 18th International Conference on Artificial Intelligence and Statistics*, volume 38 of *Proceedings of Machine Learning Research*, pp. 351–360. PMLR, 2015.

Kim, H.-M., Mallick, B., and Holmes, C. Analyzing nonstationary spatial data using piecewise gaussian processes. *Journal of the American Statistical Association*, 100:653–668, 02 2005.

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

Lawrence, N. D., Seeger, M., and Herbrich, R. Fast Sparse Gaussian Process Methods: The Informative Vector Machine. In *Advances in Neural Information Processing Systems 15*, pp. 625–632. MIT Press, 2002.

Liu, H., Ong, Y., Shen, X., and Cai, J. When gaussian process meets big data: A review of scalable gps. *IEEE Transactions on Neural Networks and Learning Systems*, 31:4405–4423, 2020.

Liu, L. and Liu, L. Amortized variational inference with graph convolutional networks for gaussian processes. In *Proceedings of the 22nd International Conference on Artificial Intelligence and Statistics*, volume 89 of *Proceedings of Machine Learning Research*, pp. 2291–2300. PMLR, 2019.

Louizos, C., Ullrich, K., and Welling, M. Bayesian compression for deep learning. In *Advances in Neural Information Processing Systems 30*, pp. 3288–3298. Curran Associates, Inc., 2017.

Molchanov, D., Ashukha, A., and Vetrov, D. Variational dropout sparsifies deep neural networks. In *Proceedings of the 34th International Conference on Machine Learning*, volume 70 of *Proceedings of Machine Learning Research*, pp. 2498–2507. PMLR, 2017.

Naish-Guzman, A. and Holden, S. The generalized FITC approximation. In *Advances in Neural Information Processing Systems 20*, pp. 1057–1064. Curran Associates Inc., 2007. ISBN 978-1-60560-352-0.

Panos, A., Dellaportas, P., and Titsias, M. K. Fully scalable gaussian processes using subspace inducing inputs. *CoRR*, abs/1807.02537, 2018.

Park, C. and Apley, D. Patchwork kriging for large-scale gaussian process regression. *Journal of Machine Learning Research*, 19, 01 2017.

Park, C. and Huang, J. Z. Efficient computation of gaussian process regression for large spatial data sets by patching local gaussian processes. *J. Mach. Learn. Res.*, 17(1): 6071–6099, January 2016. ISSN 1532-4435.

Pleiss, G., Gardner, J., Weinberger, K., and Wilson, A. G. Constant-time predictive distributions for Gaussian processes. In *Proceedings of the 35th International Conference on Machine Learning*, volume 80 of *Proceedings of Machine Learning Research*, pp. 4114–4123. PMLR, 2018.

Pourhabib, A., Liang, F., and Ding, Y. Bayesian site selection for fast Gaussian process regression. *Institute of Industrial Engineers Transactions*, 46(5):543–555, 2014.

Quiñonero Candela, J. and Rasmussen, C. E. A unifying view of sparse approximate Gaussian process regression. *Journal of Machine Learning Research*, 6:1939–1959, 2005. ISSN 1532-4435.

Rasmussen, C. E. and Williams, C. *Gaussian Processes for Machine Learning*. MIT Press, 2006.

Rudi, A., Carratino, L., and Rosasco, L. FALKON: An Optimal Large Scale Kernel Method. In *Advances in Neural Information Processing Systems 30*, pp. 3888–3898. Curran Associates, Inc., 2017.

Seeger, M., Williams, C. K. I., and Lawrence, N. D. Fast forward selection to speed up sparse Gaussian process regression. In *Artificial Intelligence and Statistics 9*, 2003.

Snelson, E. and Ghahramani, Z. Sparse Gaussian Processes using Pseudo-inputs. In *Advances in Neural Information Processing Systems 18*, pp. 1257–1264. MIT Press, 2005.

Titsias, M. K. Variational Learning of Inducing Variables in Sparse Gaussian Processes. In *Proceedings of the 12th International Conference on Artificial Intelligence and Statistics*, volume 5 of *Proceedings of Machine Learning Research*, pp. 567–574. PMLR, 2009.

Urtasun, R. and Darrell, T. Sparse probabilistic regression for activity-independent human pose inference. In *2008 IEEE Conference on Computer Vision and Pattern Recognition*, pp. 1–8, 2008.

Wilson, A. and Nickisch, H. Kernel Interpolation for Scalable Structured Gaussian Processes (KISS-GP). In *Proceedings of the 32nd International Conference on Machine Learning*, volume 37 of *Proceedings of Machine Learning Research*, pp. 1775–1784. PMLR, 2015.

Wilson, A., Dann, C., and Nickisch, H. Thoughts on massively scalable gaussian processes. *ArXiv*, abs/1511.01870, 2015.

Wilson, A. G., Hu, Z., Salakhutdinov, R. R., and Xing, E. P. Stochastic Variational Deep Kernel Learning. In Lee, D. D., Sugiyama, M., Luxburg, U. V., Guyon, I., and Garnett, R. (eds.), *Advances in Neural Information Processing Systems 29*, pp. 2586–2594. Curran Associates, Inc., 2016.