This document is downloaded from DR-NTU (https://dr.ntu.edu.sg)
Nanyang Technological University, Singapore.

Understanding and comparing scalable Gaussian process regression for big data

Liu, Haitao; Cai, Jianfei; Ong, Yew-Soon; Wang, Yi

2018

Liu, H., Cai, J., Ong, Y.-S., & Wang, Y. (2019). Understanding and comparing scalable gaussian process regression for big data. Knowledge-Based Systems, 164, 324-335. doi:10.1016/j.knosys.2018.11.002

https://hdl.handle.net/10356/139619

https://doi.org/10.1016/j.knosys.2018.11.002

© 2018 Elsevier B.V. All rights reserved. This paper was published in Knowledge-Based Systems and is made available with permission of Elsevier B.V.

Downloaded on 19 Mar 2024 09:41:25 SGT

arXiv:1811.01159v1 [stat.ML] 3 Nov 2018

Understanding and Comparing Scalable Gaussian Process Regression for Big Data

Haitao Liu[a,*], Jianfei Cai[b], Yew-Soon Ong[b,c], Yi Wang[d]

[a] *Rolls-Royce@NTU Corporate Lab, Nanyang Technological University, Singapore 637460*
[b] *School of Computer Science and Engineering, Nanyang Technological University, Singapore 639798*
[c] *Data Science and Artificial Intelligence Research Center, Nanyang Technological University, Singapore 639798*
[d] *Central Technology Group, Rolls-Royce Singapore, 1 Seletar Aerospace Crescent, Singapore 797565*

Abstract

As a non-parametric Bayesian model which produces informative predictive distribution, Gaussian process (GP) has been widely used in various fields, like regression, classification and optimization. The cubic complexity of standard GP however leads to poor scalability, which poses challenges in the era of big data. Hence, various scalable GPs have been developed in the literature in order to improve the scalability while retaining desirable prediction accuracy. This paper devotes to investigating the methodological characteristics and performance of representative global and local scalable GPs including sparse approximations and local aggregations from four main perspectives: scalability, capability, controllability and robustness. The numerical experiments on two toy examples and five real-world datasets with up to 250K points offer the following findings. In terms of scalability, most of the scalable GPs own a time complexity that is linear to the training size. In terms of capability, the sparse approximations capture the long-term spatial correlations, the local aggregations capture the local patterns but suffer from over-fitting in some scenarios. In terms of controllability, we could improve the performance of sparse approximations by simply increasing the inducing size. But this is not the case for local aggregations. In terms of robustness, local aggregations are robust to various initializations of hyperparameters due to the local attention mechanism. Finally, we highlight that the proper hybrid of global and local scalable GPs may be a promising way to improve both the model capability and scalability for big data.

Keywords: Gaussian process, big data, sparse approximations, local aggregations

1. Introduction

Surrogate-assisted modeling and optimization have been extensively deployed to facilitate modern aeroengine design [1, 2, 3, 4] due to the representational capability of

*Corresponding author
Email addresses: htliu@ntu.edu.sg (Haitao Liu), ASJFCai@ntu.edu.sg (Jianfei Cai), ASYSOng@ntu.edu.sg (Yew-Soon Ong), Yi.Wang4@Rolls-Royce.com (Yi Wang)

complex features. Among current surrogates (also known as machine learning models), as a non-parametric Bayesian model, Gaussian process (GP) [5] (also known as Kriging or emulator), has gained popularity.

Given n training data points $\boldsymbol{X} = \{\boldsymbol{x}_i \in R^d\}_{i=1}^n$ and the relevant observations $\boldsymbol{y} = \{y_i \in R\}_{i=1}^n$, GP intends to infer the latent function $f : R^d \mapsto R$, which is drawn from a Gaussian process, to describe the data pattern. Compared to other popular machine learning models, e.g., random forest [6] and deep neural networks [7], the Bayesian perspective allows GP to provide not only the predictions at unseen points but also the uncertainties about the predictions.[1] The informative predictive distribution enables GP to be deployed in various scenarios, e.g., active learning [9], time-series forecast [10] and Bayesian optimization [11]. However, an inherent disadvantage of GP is that it scales poorly with the training size n, since as a kernel method it handles the data in a one-shot fashion. The standard implementation of GP needs to invert an $n \times n$ covariance matrix, resulting in $\mathcal{O}(n^2)$ complexity in storage and $\mathcal{O}(n^3)$ complexity in time. Hence, the full GP becomes intractable when the training size is greater than $\mathcal{O}(10^4)$ [12].

In the era of big data,[2] numerous data brings vast quantity of information to enhance the analysis, learning and exploration in the machine learning community. Hence, there is a great demand for improving the scalability of standard GP while retaining desirable prediction accuracy.

In current literature, various scalable GPs have been developed for handing big data in different ways. They usually can be classified into two main categories:

- *Global approximations* that summarize the whole training data using a small subset. The simplest way is the subset-of-data (SoD) which trains GP on a random subset of the training data $\mathcal{D} = \{\boldsymbol{X}, \boldsymbol{y}\}$ [13]. The performance of SoD however is limited due to the ignorance of the remaining data. The most commonly used global approximations are the so-called sparse approximations [14, 15, 16, 17], which employ a set of global inducing variables to approximate the joint prior or the interested posterior. By using m inducing points, sparse approximations reduce the time complexity of full GP to $\mathcal{O}(nm^2)$ [15, 16] and more remarkably, $\mathcal{O}(m^3)$ [17]. The advantages of sparse approximations are that they are usually derived in a unifying and elegant Bayesian framework, thus yielding a complete probabilistic model. The limitation however is that the modeling performance is limited by the small set of global inducing points, i.e., it is difficult for them to capture the quick-varying features, especially in high dimensions [18].

- *Local approximations* that employ the idea of divide-and-conquer (D&C) to distribute the whole training process over multiple local GP experts. By partitioning the data $\mathcal{D}$ into M local subsets $\{\mathcal{D}_i = \{\boldsymbol{X}_i, \boldsymbol{y}_i\}\}_{i=1}^M$, each of which is assumed to have the same training size $m_0 = n/M$, local approximations yield the time complexity of $\mathcal{O}(nm_0^2)$. The benefits brought by the D&C idea are that (i) it enables capturing local patterns; and (ii) it scales local approximations up to arbitrary

[1] It has been pointed out that GP is equivalent to a shallow but infinitely wide neural network with Gaussian weights [8].

[2] Big data here mainly refers to the datasets with extremely large sample size n such that they are expensive to be stored and analyzed.

dataset due to the straightforward parallel/distributed structure. The simplest local approximation is to training individual GP experts on multiple local subsets, which however suffers from discontinuous predictions and local over-fitting. Hence, local aggregations have been presented to smooth the predictions by aggregating the predictions from multiple local experts [12, 19, 20, 21, 22]. The limitation of local aggregations however is that they cannot provide a complete probabilistic model, which results in the so-called Kolmogorov inconsistency [23].

This paper intends to comprehensively investigate the characteristics and performance of representative scalable GPs on real-world large-scale datasets. As for the global approximations, we mainly introduce and compare the sparse approximations. Particularly, the sparse approximations are classified into two categories: the *prior approximations* [15, 24, 25] which approximate the prior but perform exact inference, and the *posterior approximations* [16, 17, 26] which retain the exact prior but perform approximate inference. We do not investigate some particular sparse approximations, e.g., the structured kernel interpolation [27] which exploits the inducing set with Kronecker structure, since the inducing size increases exponentially with the dimensionality d, and the training quickly becomes intractable when $d > 5$.

As for the local approximations, we introduce and compare the pure local GPs and the local aggregations using product-of-experts and Bayesian committe machine [12, 19, 20, 21, 22]. We here do not investigate the mixture-of-experts [28, 29] since it is mainly designed for capturing non-stationary features and suffers from intractable inference.

The comparative study introduces and investigates the characteristics of representative scalable GPs from four main perspectives:

- *Scalability* that indicates the time complexity of scalable GPs to handle big data;
- *Capability* that means the representational ability of scalable GPs to handle various tasks;
- *Controllability* that indicates whether we can easily improve the performance of scalable GPs by tuning model parameters, e.g., the inducing size or the number of experts;
- *Robustness* means the sensitivity of scalable GPs to various initializations of hyperparameters.

The remainder of the paper is organized as follows. Section 2 briefly introduces the standard GP. Thereby, Sections 3 and 4 respectively introduce the global and local approximations. Thereafter, Section 5 comprehensively investigates the characteristics and performance of these scalable GPs on several toy examples and five real-world large-scale datasets with up to 250K data points. Finally, Section 6 provides concluding remarks.

2. Gaussian processes regression

As a non-parametric Bayesian model, the Gaussian process places a GP prior over the latent function $f : R^d \mapsto R$ as

$$f(\boldsymbol{x}) = \mathcal{GP}(m(\boldsymbol{x}), k(\boldsymbol{x}, \boldsymbol{x}')), \tag{1}$$

where $m(.)$ is the mean function which is usually taken as zero without loss of generality, and $k(.,.)$ is the kernel function which controls the model smoothness. Popularly, we use the well-known squared exponential (SE) kernel function with automatic relevance determination (ARD) as

$$k(\boldsymbol{x}, \boldsymbol{x}') = \sigma_f^2 \exp\left(-\frac{1}{2}\sum_{i=1}^{d}\frac{(x_i - x_i')^2}{l_i^2}\right), \tag{2}$$

where σ_f^2 is the output signal, and l_i is the input length-scale along the ith dimension. Fig. 1(a) illustrates several samples drawn from the zero-mean GP prior.

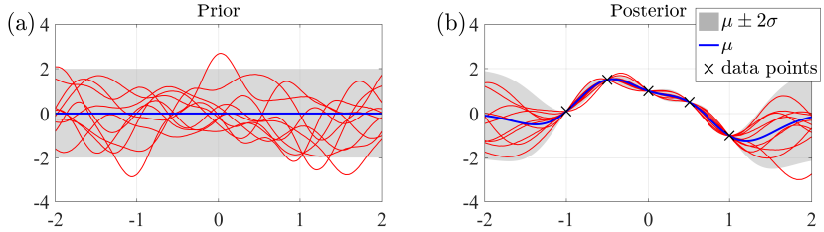


Figure 1: Samples (red curves) drawn from (a) the zero-mean GP prior and (b) the posterior after observing four data points.

Let us consider a regression task $y(\boldsymbol{x}) = f(\boldsymbol{x}) + \epsilon$ where $\epsilon \sim \mathcal{N}(0, \sigma_\epsilon^2)$ is the *iid* noise, the likelihood writes $p(y|f) = \mathcal{N}(y|f, \sigma_\epsilon^2)$. Typically, the GP can be trained by finding the optimal hyperparameters $\boldsymbol{\theta} = \{\sigma_f^2, l_1, \cdots, l_d, \sigma_\epsilon^2\}$ through maximizing the log marginal likelihood (model evidence) as

$$\log p(\boldsymbol{y}) = -\frac{n}{2}\log(2\pi) - \frac{1}{2}\log|\boldsymbol{K}_{nn} + \sigma_\epsilon^2\boldsymbol{I}| - \frac{1}{2}\boldsymbol{y}^\mathsf{T}(\boldsymbol{K}_{nn} + \sigma_\epsilon^2\boldsymbol{I})^{-1}\boldsymbol{y}, \tag{3}$$

where $\boldsymbol{K}_{nn} = k(\boldsymbol{X}, \boldsymbol{X})$ is the $n \times n$ kernel matrix. Conditioned on the training data and the inferred hyperparameters, we derive the predictive (posterior) distribution $p(f_*|\mathcal{D}, \boldsymbol{x}_*)$ at a test point $\boldsymbol{x}_*$ with the mean and variance respectively given as

$$\mu_* = k_{*n}[\boldsymbol{K}_{nn} + \sigma_\epsilon^2\boldsymbol{I}]^{-1}\boldsymbol{y}, \tag{4a}$$

$$\sigma_*^2 = k_{**} - \boldsymbol{k}_{*n}[\boldsymbol{K}_{nn} + \sigma_\epsilon^2\boldsymbol{I}]^{-1}\boldsymbol{k}_{*n}^\mathsf{T}, \tag{4b}$$

where $k_{**} = k(\boldsymbol{x}_*, \boldsymbol{x}_*)$ and $\boldsymbol{k}_{*n} = k(\boldsymbol{x}_*, \boldsymbol{X})$. Hence, the predictive distribution of y_* has $p(y_*|\mathcal{D}, \boldsymbol{x}_*) = \mathcal{N}(y_*|\mu_*, \sigma_*^2 + \sigma_\epsilon^2)$. Fig. 1(b) depicts several samples drawn from the posterior after observing four data points.

It is found in (3) that the computational bottleneck in the full GP inference is the inversion $(\boldsymbol{K}_{nn} + \sigma_\epsilon^2\boldsymbol{I})^{-1}$ and the determinant $|\boldsymbol{K}_{nn} + \sigma_\epsilon^2\boldsymbol{I}|$, both of which require the time complexity of $\mathcal{O}(n^3)$ through standard Cholesky decomposition.

The cubic complexity poses urgent demand of improving the scalability of standard GP for handling big data. In what follows, we will introduce representative scalable GPs including

- *global approximations*, particularly the sparse approximations, which distillate the training data through m inducing points; and

- *local approximations*, particularly the local aggregations, which follow the idea of D&C to aggregate multiple GP experts for boosting predictions.

3. Global approximations

The simplest global approximation is the subset-of-data (SoD) which trains GP on a random subset of the training data. Given m ($m \ll n$) subset points, the time complexity of SoD is substantially reduced to $\mathcal{O}(m^3)$ in comparison to the full GP. The SoD however may produce over-confident predictions by ignoring the remaining data.

To date, the popular global approximations are sparse approximations [14] motivated by the Nyström approximation [30]. Sparse approximations employ an inducing set $\boldsymbol{X}_m$ which includes m points to summarize the whole training data, thus reducing the time complexity to $\mathcal{O}(nm^2)$.[3] The latent inducing variables $\boldsymbol{f}_m$ akin to $\boldsymbol{f}$ follow the same GP prior $p(\boldsymbol{f}_m) = \mathcal{N}(\boldsymbol{f}_m|\boldsymbol{0}, \boldsymbol{K}_{mm})$.

In what follows, current sparse approximations are classified into two main categories: the *prior approximations* which approximate the prior but perform exact inference, and the *posterior approximations* which retain the exact prior but perform approximate inference.

3.1. Prior approximations

Given the independence assumption $\boldsymbol{f} \perp f_*|\boldsymbol{f}_m$, the joint prior $p(\boldsymbol{f}, f_*)$ can be derived by marginalizing out $\boldsymbol{f}_m$ as

$$p(\boldsymbol{f}, f_*) = \int p(\boldsymbol{f}|\boldsymbol{f}_m)p(f_*|\boldsymbol{f}_m)p(\boldsymbol{f}_m)d\boldsymbol{f}_m. \tag{5}$$

Given the Nyström notation $\boldsymbol{Q}_{ab} = \boldsymbol{K}_{am}\boldsymbol{K}_{mm}^{-1}\boldsymbol{K}_{mb}$, the training and test conditionals in (5) are respectively expressed as

$$p(\boldsymbol{f}|\boldsymbol{f}_m) = \mathcal{N}(\boldsymbol{f}|\boldsymbol{K}_{nm}\boldsymbol{K}_{mm}^{-1}\boldsymbol{f}_m, \boldsymbol{K}_{nn} - \boldsymbol{Q}_{nn}), \tag{6a}$$
$$p(f_*|\boldsymbol{f}_m) = \mathcal{N}(f_*|\boldsymbol{k}_{*m}\boldsymbol{K}_{mm}^{-1}\boldsymbol{f}_m, k_{**} - Q_{**}). \tag{6b}$$

Now we clearly see that $\boldsymbol{f}_m$ is called inducing variables since it induces the dependence between the independent $\boldsymbol{f}_m$ and f_*.

To achieve computational gains, prior approximations further modify the joint prior as

$$p(\boldsymbol{f}, f_*) \approx q(\boldsymbol{f}, f_*) = \int q(\boldsymbol{f}|\boldsymbol{f}_m)q(f_*|\boldsymbol{f}_m)p(\boldsymbol{f}_m)d\boldsymbol{f}_m. \tag{7}$$

Compared to (5), the training and test conditionals are respectively approximated in (7) through replacing the co-variance matrices as

$$q(\boldsymbol{f}|\boldsymbol{f}_m) = \mathcal{N}(\boldsymbol{f}|\boldsymbol{K}_{nm}\boldsymbol{K}_{mm}^{-1}\boldsymbol{f}_m, \tilde{\boldsymbol{Q}}_{nn}), \tag{8a}$$
$$q(f_*|\boldsymbol{f}_m) = \mathcal{N}(f_*|\boldsymbol{k}_{*m}\boldsymbol{K}_{mm}^{-1}\boldsymbol{f}_m, \tilde{Q}_{**}). \tag{8b}$$

[3] The inducing points in $\boldsymbol{X}_m$ are usually regarded as hyperparameters to be inferred.

Thereafter, we marginalize out all the latent variables to approximate the log marginal likelihood $\log p(\boldsymbol{y})$ as

$$\log q(\boldsymbol{y}) = -\frac{n}{2}\log(2\pi) - \frac{1}{2}\log|\tilde{\boldsymbol{Q}}_{nn} + \boldsymbol{Q}_{nn} + \sigma_\epsilon^2\boldsymbol{I}| - \frac{1}{2}\boldsymbol{y}^\mathsf{T}(\tilde{\boldsymbol{Q}}_{nn} + \boldsymbol{Q}_{nn} + \sigma_\epsilon^2\boldsymbol{I})^{-1}\boldsymbol{y}. \quad (9)$$

Through particular selections of $\tilde{\boldsymbol{Q}}_{nn}$, we can efficiently calculate the determinant $|\tilde{\boldsymbol{Q}}_{nn} + \boldsymbol{Q}_{nn} + \sigma_\epsilon^2\boldsymbol{I}|$ and the inversion $(\tilde{\boldsymbol{Q}}_{nn} + \boldsymbol{Q}_{nn} + \sigma_\epsilon^2\boldsymbol{I})^{-1}$ in (9) by only depending on $\boldsymbol{K}_{mm}^{-1}$.

For instance, the subset-of-regressors (SoR) [24, 31, 32] imposes $\tilde{\boldsymbol{Q}}_{nn} = \boldsymbol{0}$ and $\tilde{Q}_{**} = 0$ to the training and test conditionals such that

$$q_{\mathrm{SoR}}(\boldsymbol{f}|\boldsymbol{f}_m) = \mathcal{N}(\boldsymbol{f}|\boldsymbol{K}_{nm}\boldsymbol{K}_{mm}^{-1}\boldsymbol{f}_m, \boldsymbol{0}), \quad (10a)$$

$$q_{\mathrm{SoR}}(f_*|\boldsymbol{f}_m) = \mathcal{N}(f_*|\boldsymbol{k}_{*m}\boldsymbol{K}_{mm}^{-1}\boldsymbol{f}_m, 0). \quad (10b)$$

This is equivalent to applying the Nyström approximation to both training and test data. With this consistent assumption, the SoR is equivalent to a degenerate[4] GP with the rank (at most) m kernel function $k_{\mathrm{SoR}}(\boldsymbol{x}_i, \boldsymbol{x}_j) = k(\boldsymbol{x}_i, \boldsymbol{X}_m)\boldsymbol{K}_{mm}^{-1}k^\mathsf{T}(\boldsymbol{x}_j, \boldsymbol{X}_m)$. However, due to the limited m freedoms in k_{SoR}, the SoR suffers from over-confident prediction variance when leaving the training data. In contrast to SoR, the deterministic training conditional (DTC) [25, 33] imposes $\tilde{\boldsymbol{Q}}_{nn} = \boldsymbol{0}$ but retains the exact test conditional as

$$q_{\mathrm{DTC}}(\boldsymbol{f}|\boldsymbol{f}_m) = \mathcal{N}(\boldsymbol{f}|\boldsymbol{K}_{nm}\boldsymbol{K}_{mm}^{-1}\boldsymbol{f}_m, \boldsymbol{0}), \quad (11a)$$

$$q_{\mathrm{DTC}}(f_*|\boldsymbol{f}_m) = p(f_*|\boldsymbol{f}_m). \quad (11b)$$

Hence, the prediction mean $\mu_{\mathrm{DTC}}(\boldsymbol{x}_*)$ is the same as that of SoR, but the prediction variance $\sigma_{\mathrm{DTC}}^2(\boldsymbol{x}_*)$ is always larger than that of SoR. Notably, due to the inconsistent approximations in (11), the DTC is not an exact GP.

Moreover, the fully independent training conditional (FITC) [15] imposes another independence assumption to remove the dependency among the latent variables $\{f_i\}_{i=1}^n$ such that the training conditional factorizes as

$$q_{\mathrm{FITC}}(\boldsymbol{f}|\boldsymbol{f}_m) = \prod_{i=1}^n p(f_i|\boldsymbol{f}_m) = \mathcal{N}(\boldsymbol{f}|\boldsymbol{K}_{nm}\boldsymbol{K}_{mm}^{-1}\boldsymbol{f}_m, \mathrm{diag}[\boldsymbol{K}_{nn} - \boldsymbol{Q}_{nn}]), \quad (12)$$

whereas the test conditional retains exact. It is found that the variances of $q_{\mathrm{FITC}}(\boldsymbol{f}|\boldsymbol{f}_m)$ are identical to that of $p(\boldsymbol{f}|\boldsymbol{f}_m)$ due to the diagonal correction term $\tilde{\boldsymbol{Q}}_{nn} = \mathrm{diag}[\boldsymbol{K}_{nn} - \boldsymbol{Q}_{nn}]$. Hence, compared to SoR and DTC which completely omit the uncertainty in (10) and (11), FITC partially retains it, resulting in a closer approximation to the prior $p(\boldsymbol{f}, f_*)$.

The extensions of FITC include for example the fully independent conditional (FIC) [14] which additionally applies the independence assumption to the test conditional $q(\boldsymbol{f}_*|\boldsymbol{f}_m)$ in order to obtain a degenerate GP with the covariance function $k_{\mathrm{FIC}}(\boldsymbol{x}_i, \boldsymbol{x}_j) = k_{\mathrm{SoR}}(\boldsymbol{x}_i, \boldsymbol{x}_j) + \delta_{ij}[k(\boldsymbol{x}_i, \boldsymbol{x}_j) - k_{\mathrm{SoR}}(\boldsymbol{x}_i, \boldsymbol{x}_j)]$, where δ_{ij} is the Kronecker's delta. Besides, the partially independent training conditional (PITC) [14] factorizes $q(\boldsymbol{f}|\boldsymbol{f}_m)$ over M independent subsets (blocks) $\{\mathcal{D}_i\}_{i=1}^M$, thus taking into account the joint distribution of $\boldsymbol{f}_i$ in each

[4] "degenerate" means the GP employs a kernel with a finite rank.

subset. However, it is argued in [34] that though providing a closer approximation to $p(\boldsymbol{f}|\boldsymbol{f}_m)$, the PITC brings little improvements over FITC.

Differently, the partially independent conditional (PIC) [34] retains the conditional independence assumption, i.e., $\boldsymbol{f} \perp f_*|\boldsymbol{f}_m$, for all the blocks except the one containing the test point $\boldsymbol{x}_*$. Suppose that $\boldsymbol{x}_* \in \mathcal{D}_j$, the training conditional is

$$q_{\mathrm{PIC}}(\boldsymbol{f}, f_*|\boldsymbol{f}_m) = p(\boldsymbol{f}_j, f_*|\boldsymbol{f}_m) \prod_{i \neq j}^{M} p(\boldsymbol{f}_i|\boldsymbol{f}_m), \tag{13}$$

which corresponds to an exact GP with $k_{\mathrm{PIC}}(\boldsymbol{x}_i, \boldsymbol{x}_j) = k_{\mathrm{SoR}}(\boldsymbol{x}_i, \boldsymbol{x}_j) + \psi_{ij}[k(\boldsymbol{x}_i, \boldsymbol{x}_j) - k_{\mathrm{SoR}}(\boldsymbol{x}_i, \boldsymbol{x}_j)]$, where $\psi_{ij} = 1$ when $\boldsymbol{x}_i$ and $\boldsymbol{x}_j$ belong to the same block; otherwise $\psi_{ij} = 0$. The PIC is regarded as a hybrid approximation since when we take all the block sizes to one, it recovers FIC; when we take the number of inducing points to zero, it recovers the pure local GP.

3.2. Posterior approximations

In contrast to prior approximations, posterior approximations, e.g., variational free energy (VFE) [16], directly approximate the posterior $p(\boldsymbol{f}, \boldsymbol{f}_m|\boldsymbol{y})$ by a free variational distribution $q(\boldsymbol{f}, \boldsymbol{f}_m|\boldsymbol{y})$. The discrepancy between $q(\boldsymbol{f}, \boldsymbol{f}_m|\boldsymbol{y})$ and $p(\boldsymbol{f}, \boldsymbol{f}_m|\boldsymbol{y})$ can be quantified by the Kullback-Leibler (KL) divergence

$$\begin{aligned} \mathrm{KL}(q(\boldsymbol{f}, \boldsymbol{f}_m|\boldsymbol{y})||p(\boldsymbol{f}, \boldsymbol{f}_m|\boldsymbol{y})) &= \int q(\boldsymbol{f}, \boldsymbol{f}_m|\boldsymbol{y}) \log \frac{q(\boldsymbol{f}, \boldsymbol{f}_m|\boldsymbol{y})}{p(\boldsymbol{f}, \boldsymbol{f}_m, \boldsymbol{y})} d\boldsymbol{f} d\boldsymbol{f}_m + \log p(\boldsymbol{y}) \\ &= -F_q + \log p(\boldsymbol{y}). \end{aligned} \tag{14}$$

It is observed that minimizing the non-negative $\mathrm{KL}(q||p)$ is equivalent to maximizing F_q, since $\log p(\boldsymbol{y})$ is fixed w.r.t. $q(\boldsymbol{f}, \boldsymbol{f}_m|\boldsymbol{y})$. Thus, F_q is called the lower bound of $\log p(\boldsymbol{y})$ or the variational free energy, which permits joint optimization of the variational parameters and hyperparameters.

Since we have $p(\boldsymbol{f}, \boldsymbol{f}_m|\boldsymbol{y}) = p(\boldsymbol{f}|\boldsymbol{f}_m)p(\boldsymbol{f}_m|\boldsymbol{y})$, the variational distribution factorizes similarly as $q(\boldsymbol{f}, \boldsymbol{f}_m|\boldsymbol{y}) = p(\boldsymbol{f}|\boldsymbol{f}_m)q(\boldsymbol{f}_m|\boldsymbol{y})$, where $q(\boldsymbol{f}_m|\boldsymbol{y}) = \mathcal{N}(\boldsymbol{f}_m|\boldsymbol{m}, \boldsymbol{S})$. Fortunately, the optimal variational distribution $q^*(\boldsymbol{f}_m|\boldsymbol{y})$ can be found using the calculus of variations. Taking the derivative of F_q w.r.t. $q(\boldsymbol{f}_m|\boldsymbol{y})$ to zero, we have

$$q^*(\boldsymbol{f}_m|\boldsymbol{y}) = \mathcal{N}(\boldsymbol{f}_m|\sigma_\epsilon^{-2}\boldsymbol{K}_{mm}\boldsymbol{\Sigma}\boldsymbol{K}_{mn}\boldsymbol{y}, \boldsymbol{K}_{mm}\boldsymbol{\Sigma}\boldsymbol{K}_{mm}), \tag{15}$$

where $\boldsymbol{\Sigma} = (\boldsymbol{K}_{mm} + \sigma_\epsilon^{-2}\boldsymbol{K}_{mn}\boldsymbol{K}_{nm})^{-1}$. Inserting $q^*(\boldsymbol{f}_m|\boldsymbol{y})$ back into F_q, we have a "collapsed" bound which is independent of $q(\boldsymbol{f}_m|\boldsymbol{y})$ as

$$\begin{aligned} F_{\mathrm{VFE}} =& -\frac{n}{2}\log(2\pi) - \frac{1}{2}\log|\boldsymbol{Q}_{nn} + \sigma_\epsilon^2\boldsymbol{I}| \\ & - \frac{1}{2}\boldsymbol{y}^\mathsf{T}(\boldsymbol{Q}_{nn} + \sigma_\epsilon^2\boldsymbol{I})^{-1}\boldsymbol{y} - \frac{1}{2\sigma_\epsilon^2}\mathrm{Tr}(\boldsymbol{K}_{nn} - \boldsymbol{Q}_{nn}) \\ =& \log q_{\mathrm{DTC}}(\boldsymbol{y}) - 0.5\sigma_\epsilon^{-2}\mathrm{Tr}(\boldsymbol{K}_{nn} - \boldsymbol{Q}_{nn}). \end{aligned} \tag{16}$$

It is found that compared to the log marginal likelihood of DTC, the VFE bound (16) involves an additional trace term $\mathrm{Tr}(\boldsymbol{K}_{nn} - \boldsymbol{Q}_{nn}) \geq 0$ which represents the total variance

of the training conditional $p(\boldsymbol{f}|\boldsymbol{f}_m)$. Particularly, $\text{Tr}(\boldsymbol{K}_{nn} - \boldsymbol{Q}_{nn}) = 0$ means that $\boldsymbol{f}_m = \boldsymbol{f}$ and we reproduce the full GP. Hence, the trace term helps VFE (i) guard against over-fitting; (ii) choose a good inducing set; and (iii) improve F_{VFE} with increasing m [16, 35, 36, 37].

To further improve the scalability of VFE, unlike the bound (16) that "integrates" out the variational distribution, $q(\boldsymbol{f}_m|\boldsymbol{y})$ now is retained in the bound F_q in order to obtain a full factorization over data points as [17]

$$F_q = \langle \log p(\boldsymbol{y}|\boldsymbol{f}) \rangle_{p(\boldsymbol{f}|\boldsymbol{f}_m)q(\boldsymbol{f}_m|\boldsymbol{y})} - \text{KL}(q(\boldsymbol{f}_m|\boldsymbol{y})||p(\boldsymbol{f}_m)), \tag{17}$$

where $\langle . \rangle_{q(.)}$ represents the expectation over the distribution $q(.)$. The newly organized bound F_q, which is looser than F_{VFE}, has a key property that it can be written as the sum of n terms because of the likelihood $p(\boldsymbol{y}|\boldsymbol{f}) = \prod_{i=1}^{n} p(y_i|f_i)$. Due to (i) the full factorization and (ii) the difficulty of optimizing the variational parameters $\boldsymbol{m}$ and $\boldsymbol{S}$ since they are defined in a non-Euclidean space, one can employ the Stochastic Variational Inference (SVI) [38] to infer the variational parameters in the natural gradient space via efficient stochastic gradient descent (SGD) algorithms [39, 40], resulting in the greatly reduced complexity of $\mathcal{O}(bm^3)$ where b is the mini-batch size in SGD. Therefore, the stochastic variational GP (SVGP) adopts the SGD to train a sparse GP at *any time* with a *small batch* of the training data in each iteration [41].[5]

4. Local approximations

Following the idea of D&C, the simplest local approximation is the pure local GPs which first partition the training data into M subsets for example through clustering techniques, and then train a GP expert on each subset using individual parameters. The D&C idea scales local approximations up to arbitrary dataset due to the straightforward parallel/distributed fashion. Besides, in comparison to the global approximations, the naive local approximation is capable of describing non-stationary features (local patterns) for improving predictions at the cost of however suffering from discontinuous predictions, inaccurate uncertainties and local over-fitting.

To smooth and boost predictions while retaining computation gains from D&C, the local aggregations, which are inspired by the idea of model averaging, were presented to aggregate the predictions from multiple experts. Particularly, given a partition $\{\mathcal{D}_i\}_{i=1}^{M}$ of $\mathcal{D}$, the aggregation methods follow the product rule [19], i.e., assuming independent GP experts, such that the marginal likelihood $p(\boldsymbol{y})$ factorizes as

$$p(\boldsymbol{y}|\boldsymbol{X}, \boldsymbol{\theta}) \approx \prod_{i=1}^{M} p_i(\boldsymbol{y}_i|\boldsymbol{X}_i, \boldsymbol{\theta}_i), \tag{18}$$

where $\boldsymbol{\theta}_i$ is a vector comprising the hyperparameters for the ith expert $\mathcal{M}_i$, and $\boldsymbol{\theta} = \{\boldsymbol{\theta}_1, \cdots, \boldsymbol{\theta}_M\}$. Similar to the sparse approximations, the aggregations reduce the time

[5] Unlike the traditional conjugate gradient descent (CGD) algorithm which takes all the training data to calculate the gradients, the SGD only takes b ($b \ll n$) out of the n data points to calculate the gradients in each iteration, thus enhancing large-scale learning.

complexity of full GP to $\mathcal{O}(nm_0^2)$ when all the experts have an equal training size $m_0 = n/M$, but requiring no additional inducing points or variational parameters.

After training the experts $\{\mathcal{M}_i\}_{i=1}^M$ on the relevant subsets $\{\mathcal{D}_i\}_{i=1}^M$, we combine their predictions together at the test point $\boldsymbol{x}_*$ by employing some aggregation criteria. For instance, under the independence assumption, the product-of-experts (PoE) [19, 21, 42, 43, 44, 45] conducts the aggregation as

$$p_{\mathcal{A}}(y_*|\mathcal{D}, \boldsymbol{x}_*) = \prod_{i=1}^{M} p_i^{\beta_i}(y_*|\mathcal{D}_i, \boldsymbol{x}_*), \tag{19}$$

where β_i is a weight quantifying the contribution of $\mathcal{M}_i$ at $\boldsymbol{x}_*$. The product of Gaussian distributions in (19) results in another Gaussian distribution with the mean and variance analytically expressed as

$$\mu_{\mathcal{A}}(\boldsymbol{x}_*) = \sigma_{\mathcal{A}}^2(\boldsymbol{x}_*) \sum_{i=1}^{M} \beta_i \sigma_i^{-2}(\boldsymbol{x}_*)\mu_i(\boldsymbol{x}_*), \tag{20a}$$

$$\sigma_{\mathcal{A}}^{-2}(\boldsymbol{x}_*) = \sum_{i=1}^{M} \beta_i \sigma_i^{-2}(\boldsymbol{x}_*), \tag{20b}$$

where $\mu_i(\boldsymbol{x}_*)$ and $\sigma_i^2(\boldsymbol{x}_*)$ are respectively the prediction mean and variance of $\mathcal{M}_i$ at $\boldsymbol{x}_*$. By taking a constant weight $\beta_i = 1$, the original PoE quantifies the contribution of each expert by the prediction precision [19]. However, the naive sum of experts' prediction precisions in (20b) will make the aggregated variance vanish quickly with increasing M, i.e., producing seriously over-confident prediction variance [22, 46]. Hence, the generalized PoE (GPoE) [21] introduces a varying weight $\beta_i = 0.5(\log \sigma_{**}^2 - \log \sigma_i^2(\boldsymbol{x}_*))$, which is defined as the difference in the differential entropy between the prior and the posterior, to weaken the votes of those poor experts with large uncertainty. This flexible weight, however, produces an explosive prediction variance when $\boldsymbol{x}_*$ is far away from $\boldsymbol{X}$ [22]. To address this issue, we could either impose a constraint $\sum_{i=1}^M \beta_i = 1$ [21] or simply employ $\beta_i = 1/M$ [12].

To improve the performance of PoEs, the Bayesian committee machine (BCM) [12, 20, 22, 47] additionally takes the prior $p(y_*|\boldsymbol{x}_*)$ into account and imposes a conditional independence assumption $p(\boldsymbol{y}|f_*, \boldsymbol{X}) = \prod_{i=1}^M p(\boldsymbol{y}_i|f_*, \boldsymbol{X}_i)$. Consequently, the aggregated predictive distribution derived from the Bayes rule is

$$p_{\mathcal{A}}(y_*|\mathcal{D}, \boldsymbol{x}_*) = \frac{\prod_{i=1}^M p_i^{\beta_i}(y_*|\mathcal{D}_i, \boldsymbol{x}_*)}{p^{\sum_{i=1}^M \beta_i - 1}(y_*|\boldsymbol{x}_*)}, \tag{21}$$

with the prediction mean and variance analytically given as

$$\mu_{\mathcal{A}}(\boldsymbol{x}_*) = \sigma_{\mathcal{A}}^2(\boldsymbol{x}_*) \sum_{i=1}^{M} \beta_i \sigma_i^{-2}(\boldsymbol{x}_*)\mu_i(\boldsymbol{x}_*), \tag{22a}$$

$$\sigma_{\mathcal{A}}^{-2}(\boldsymbol{x}_*) = \sum_{i=1}^{M} \beta_i \sigma_i^{-2}(\boldsymbol{x}_*) + \left(1 - \sum_{i=1}^{M} \beta_i\right) \sigma_{**}^{-2}. \tag{22b}$$

Table 1: Comparison of the space and time complexity of representative global and local scalable GPs, where m is the inducing size for sparse approximations and m_0 is the training size of each expert for local aggregations.

Model	Storage	Training	Test
SoR, DTC, FITC, VFE	$\mathcal{O}(nm)$	$\mathcal{O}(nm^2)$	$\mathcal{O}(m^2)$
SVGP	$\mathcal{O}(bm^2)$	$\mathcal{O}(bm^3)$	$\mathcal{O}(m^2)$
(G)PoE, (R)BCM	$\mathcal{O}(nm_0)$	$\mathcal{O}(nm_0^2)$	$\mathcal{O}(nm_0)$

Compared to (20b), the prior correlation in (22b) helps BCMs recover the GP prior when leaving the training data. The original BCM [20] takes $\beta_i = 1$, and the newly developed robust BCM (RBCM) [12] employs the varying β_i like GPoE in order to produce robust predictions within $\boldsymbol{X}$. The BCMs however are found to suffer from weak experts when leaving $\boldsymbol{X}$ [12, 22].

Regarding the model capability, the PoEs in (19) allow the GP experts to own individual parameters in order to capture non-stationary features, whereas the BCMs in (21) cannot due to the shared prior $p(y_*|\boldsymbol{x}_*)$ across experts. Besides, for the BCMs using experts with shared hyperparameters, the direct aggregation of $\{p_i(y_*|\mathcal{D}_i, \boldsymbol{x}_*)\}_{i=1}^M$ induces seriously over-confident prediction variance with increasing n [22]. To alleviate this issue, we could use the newly proposed generalized RBCM framework [22]; or simply, we aggregate $\{p_i(f_*|\mathcal{D}_i, \boldsymbol{x}_*)\}_{i=1}^M$ instead of $\{p_i(y_*|\mathcal{D}_i, \boldsymbol{x}_*)\}_{i=1}^M$, and finally add the estimated noise variance [12].

Finally, Table 1 summarizes the training and test time complexity of representative global and local scalable GPs. It is found that if $m = m_0$, all the scalable GPs except SVGP own the same training complexity. Besides, the local aggregations have a higher test complexity since they need the predictions of all the experts at $\boldsymbol{x}_*$. Finally, note that the computations in the scalable GPs listed above can be sped up through distributed/parallel computing, see [12, 48, 49].

5. Numerical experiments

This section first employs several toy examples to illustrate the methodological characteristics of global and local scalable GPs, and then applies some of them to five real-world datasets with up to 250K data points. The goal of this comparative study is to enhance the understanding of representative scalable GPs and investigate their usability by addressing the issues as: (i) what are the features of these scalable GPs and in what scenarios they are useful? (ii) are the scalable GPs controllable to model parameters? and (iii) are the scalable GPs robust to various initializations of hyperparameters?

In the comparative study below, we implement the scalable GPs based on the GPML toolbox[6], the GPstuff toolbox[7] and the GPy toolbox[8]. Before model training, the data pre-processing is performed by normalizing $\boldsymbol{y}$ and each column of $\boldsymbol{X}$ to $\mathcal{N}(0, 1)$. In modeling, we employ the SE kernel in (2), and initialize the length-scales $l_1, \cdots, l_d$ as

[6] http://www.gaussianprocess.org/gpml/code/matlab/doc/
[7] https://github.com/gpstuff-dev/gpstuff
[8] https://github.com/SheffieldML/GPy

0.5, the signal variance σ_f^2 as 1.0, and the noise variance σ_ϵ^2 as 0.1. All the scalable GPs except SVGP employ the CGD for inference with the maximum number of iterations as 100. The particular SVGP employs the Adadelta SGD algorithm [39] for inference with the step rate as 0.1, the momentum as 0.9, and the maximum number of iterations as 1000. All the codes are executed on a personal computer with four 3.70 GHz cores and 16 GB RAM.

Finally, given n_* test points $\{\boldsymbol{X}_*, \boldsymbol{y}_*\}$, we assess the prediction accuracy of scalable GPs using the standardized mean square error (SMSE) defined as

$$\text{SMSE} = \frac{\sum_{j=1}^{n_*} (y_{*j} - \mu_{*j})^2}{n_* \times \text{var}(\boldsymbol{y})}. \tag{23}$$

The SMSE criterion quantifies the discrepancy between the predictions and the exact function values; particularly, it equals to one when the model always predicts the mean of $\boldsymbol{y}$. Besides, to quantify the quality of predictive distribution that considers both prediction mean and variance, we employ the mean standardized log loss (MSLL) defined as

$$\text{MSLL} = \frac{1}{n_*} \sum_{j=1}^{n_*} \left[\log \mathcal{N}(y_{*j}|\overline{\boldsymbol{y}}, \text{var}(\boldsymbol{y})) - \log p(y_{*j}|\mathcal{D}, \boldsymbol{x}_{*j})\right], \tag{24}$$

where $\log p(y_{*j}|\mathcal{D}, \boldsymbol{x}_{*j}) = -0.5\left[\log(2\pi\sigma_{*j}^2) + (y_{*j} - \mu_{*j})^2/\sigma_{*j}^2\right]$. The MSLL will be negative for high quality models, and particularly, it will be zero when the model always predicts the mean and variance of $\boldsymbol{y}$.

5.1. Toy examples

5.1.1. Characteristics of sparse approximations

This section attempts to study various sparse approximations via a toy example expressed as

$$y(x) = \text{sinc}(x) + \epsilon, \quad x \in [-4, 4], \tag{25}$$

where $\epsilon = \mathcal{N}(0, 0.04)$. We randomly draw 120 training points from this generative function; besides, we generate 300 test points in $[-7, 7]$. The sparse approximations include SoR, DTC, FITC, PIC, VFE and SVGP. As for model configurations, we choose 15 initial inducing points equally spaced in $[-4, 4]$; particularly, we partition the training data into $M = 10$ disjoint subsets for the PIC approximation; we use the batch size of $b = 30$ for SVGP. Fig. 2 depicts the predictions of the six sparse approximations, respectively, on the toy example. For the purpose of comparison, the results of full GP are involved in the figure.

It is first observed that the two posterior approximations, VFE and its variant SVGP, provide the best approximation to the full GP, since they directly approximate the posterior with no modification to the joint prior. If we gradually increase the inducing size m, they will finally converge to the full GP. To verify this, Fig. 3(a) depicts the convergence curves of VFE with different m values. The horizontal axis represents the number of optimization iterations, and the vertical axis represents the negative log marginal likelihood (NLML). The black dash line indicates the converged NLML value of full GP. It is observed that when using a small inducing set ($m = 5$), the VFE converges with a larger NLML value than that of full GP; but with the increase of m, the NLML of VFE quickly

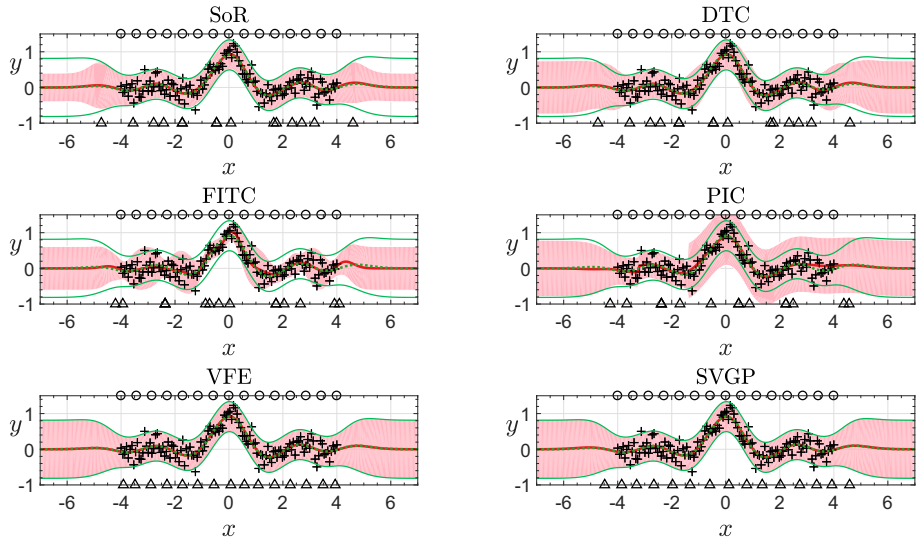


Figure 2: A toy example to illustrate the characteristics of various sparse approximations. The crosses represent the training points. The green dot curve represents the prediction mean of full GP. The two green curves represent 95% confidence interval of the full GP prediction. The red curve represents the prediction mean of a sparse approximation. The shadow region represents 95% confidence interval of the sparse prediction. The top circles and bottom triangles represent the positions of initial and optimized inducing points, respectively.

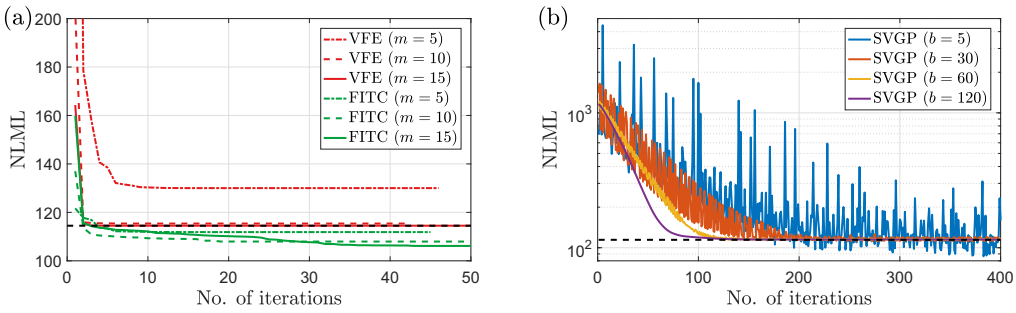


Figure 3: The convergence curves of (a) VFE and FITC using CGD with different inducing sizes, and (b) SVGP using SGD with different batch sizes on the toy example.

converges to that of full GP. Note that because of the simplicity of this toy example, the VFE with $m = 15$ has provided a very close approximation to the full GP.

The differences between VFE and SVGP are that (i) SVGP employs a less tight bound (17) defined in an augmented probabilistic space;[9] and (ii) SVGP employs the SGD for optimization. Fig. 3(b) shows the convergence curves of SVGP using different batch sizes for SGD. It is observed that compared to the deterministic CGD, the SGD

[9]Due to the explicit variational distribution $q(\boldsymbol{f}_m|\boldsymbol{y})$ in (17), the SVGP should consider $m+m(m+1)/2$ additional variational parameters.

(i) produces many fluctuations with a small batch size; and (ii) converges more slowly even by using $b = 120$, because of the relaxed bound F_q (17) and the huge parameter space. But the superiority of SGD for big data is that (i) it greatly reduces the computational complexity by running in a batch mode; and (ii) it can achieve better solutions for complicated functions by easily escaping from local optima and saddle points. For example, the SVGP with $b = 5$ in Fig. 3(b) finds some smaller NLML values than that of full GP.

As for the four prior approximations, it is observed in Fig. 2 that the SoR produces severely over-confident prediction variance when leaving $\boldsymbol{X}$. This is because the SoR imposes too restrictive assumptions to the training and test data in (10). Though equipped with the same prediction expressions to that of VFE, the DTC produces poorer predictions due to the restrictive marginal likelihood. Different from SoR and DTC, the FITC is capable of capturing the heteroscedastic noise. Let us look inside the prediction variance of FITC

$$\sigma^2(\boldsymbol{x}_*) = k_{**} - Q_{**} + \boldsymbol{k}_{*m}\boldsymbol{\Sigma}_{mm}\boldsymbol{k}_{m*},$$

where $\boldsymbol{\Sigma}_{mm} = (\boldsymbol{K}_{mm} + \boldsymbol{K}_{mn}\boldsymbol{\Lambda}^{-1}\boldsymbol{K}_{nm})^{-1}$ and $\boldsymbol{\Lambda} = \text{diag}[\boldsymbol{K}_{nn} - \boldsymbol{Q}_{nn}] + \sigma_\epsilon^2\boldsymbol{I}_n$. It is found that the diagonal term $\text{diag}[\boldsymbol{K}_{nn} - \boldsymbol{Q}_{nn}]$, which represents the input-dependent variances of the training conditional $p(\boldsymbol{f}|\boldsymbol{f}_m)$, enables FITC to capture the heteroscedasticity of noise at the cost of (i) producing an invalid estimation (nearly zero) of the noise variance σ_ϵ^2, (ii) worsening the accuracy of prediction mean, and (iii) producing overlapped inducing points [36]. Finally, the hybrid PIC approximation, which is capable of capturing local patterns, is found to produce discontinuous predictions and conservative variances.

As discussed before, the posterior approximations VFE and SVGP can be regarded as the approximation to full GP. But this is not the case for the four prior approximations. Prior approximations seek to achieve good prediction accuracy at a low computational cost, rather than faithfully converging to the full GP with increasing m. This can be reflected by two observations: (i) some of the optimized inducing points in Fig. 2 overlap with each other, especially for FITC;[10] and (ii) rather than converging to the NLML of full GP with the increase of m, the FITC tends to produce a different, smaller NLML value in Fig. 3(a).

5.1.2. Characteristics of local approximations

Different from the sparse approximations which capture long-term spatial correlations based on the global inducing set, the local approximations focus on subspace learning via local experts, thus capturing local patterns (non-stationary features). As an illustration, we apply the pure local GPs to the toy example (25) in Fig. 4. Particularly, we partition the 120 training points into $M = 10$ local subsets and train individual GP experts. It is found that the naive local approximation captures local patterns at the cost of (i) producing discontinuity on the boundaries of sub-regions, and (ii) risking over-fitting in some local regions.

The first issue could be addressed by the model averaging strategies like PoE and BCM. As shown in Fig. 4, the PoE and GPoE yield continuous predictions. But the original product rule makes PoE produce over-confident prediction variance and deteriorated prediction mean, which are not preferred in practice. Instead, the GPoE employs

[10] The reason for FITC to produce overlapped inducing points has been theoretically analyzed in [36].

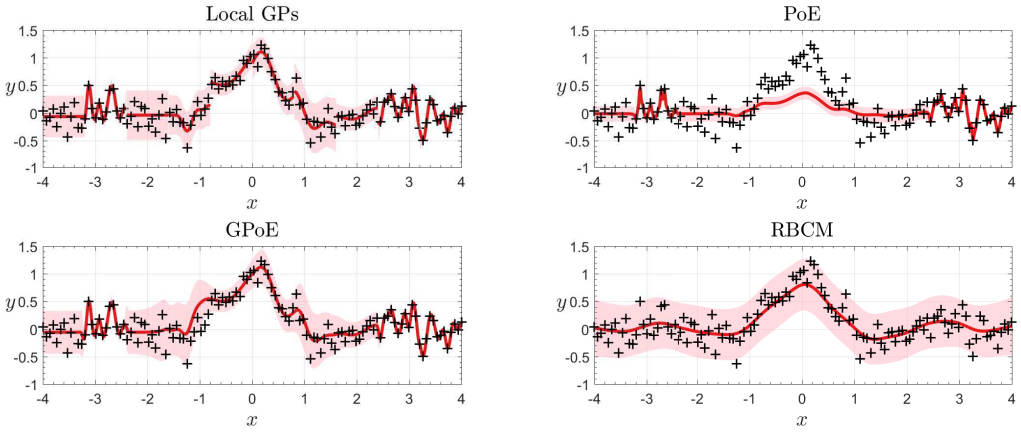


Figure 4: A toy example to illustrate the characteristics of various local approximations.

a varying weight β_i to weaken the votes of poor experts, resulting in sensible prediction mean and variance.

The second issue however is directly inherited by PoE and GPoE, since they have no mechanism to avoid local over-fitting. To alleviate the over-fitting issue, we could increase the training size for each expert (i.e., small M value) in order to take into account the long-term spatial correlation at the cost of degrading the capability of capturing local patterns and increasing the complexity. Besides, the hybrid approximations [34, 50, 51] may be a promising solution to this issue, since they inherit the advantages of both global and local approximations.[11] Finally, another alternative way to guard against local over-fitting is to sharing hyperparameters across experts, like the RBCM in Fig. 4. It is observed that the RBCM has conservative prediction variances. Note that though sharing the hyperparameters, the local structure itself could help RBCM capture local patterns, which will be shown in next section.

5.1.3. Global vs. local approximations

Here we take the time-series *solar* dataset [52] which contains 391 data points with *quick-varying* features for comparing global and local approximations. The study attempts to investigate whether global and local approximations could handle complicated tasks using limited computational resources.

Among the four local approximations, we take the RBCM for example and use the k-means technique to partition the training data into $M = 10$ disjoint subsets, resulting in approximately 39 data points for each expert. Among the six global approximations, we employ the VFE and use $m = 40$ inducing points. Particularly, in order to model the quick-varying features in this dataset, we initialize the length-scales in the SE kernel (2) with a small value of 0.0442 after data normalization.[12]

[11] The current hybrid approximations, e.g., the PIC in Fig. 2, however still suffer from the discontinuity issue.

[12] In the original input space $[1600, 2000]$ of the *solar* dataset, the initial length-scale is equal to 5, which is a very small value.

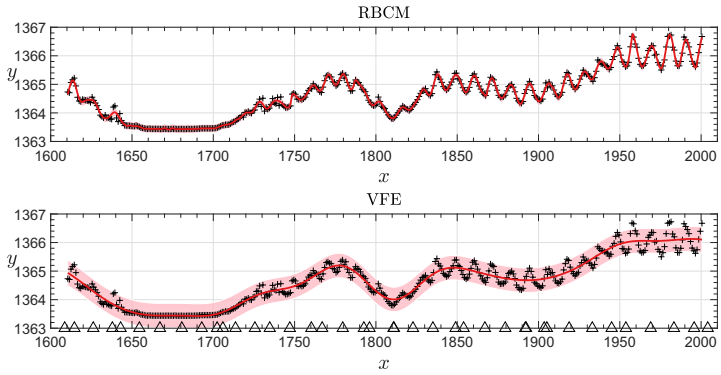


Figure 5: A *solar* example involving quick-varying features to illustrate the characteristics of RBCM and VFE.

Fig. 5 depicts the modeling results of RBCM and VFE on the *solar* dataset. It turns out that though sharing the hyperparameters across experts, the RBCM captures the quick-varying features successfully. This is because the *localized structure* helps RBCM equipped with *local attention* to take into account the local patterns when estimating the shared hyperparameters. On the contrary, even by using such small initial length-scales, the VFE fails to capture the quick-varying features due to the small set of *global* inducing points. The performance of VFE indeed could be improved by increasing the number of inducing points, which however becomes unattractive in terms of computational complexity.

Besides, the local attention mechanism may help improve the robustness of RBCM to various settings. To verify this, we study an extreme case wherein the length-scales in the SE kernel are initialized as 2.0 which in the original space is 226. By simply increasing the number of experts to $M = 40$, the RBCM again is enabled to capture the quick-varying feature at, interestingly, lower computational cost. But it is notable that the increase of M does not always correspond to good predictions, since practical datasets often do not follow the *iid* noise assumption. That means too much localized experts for RBCM may degrade the generalization capability (indicated by severely over-confident prediction variance), which will be observed in next section.

5.2. Real-world datasets

This section seeks to assess the representative global and local scalable GPs on five real-world datasets with different characteristics, e.g., regular/clustering inputs, noise/noiseless observations, and homoscedastic/heteroscedastic noise, see Table 2. The *airfoil* dataset [53] comprises different sizes of NACA 0012 airfoils at various wind tunnel speeds and angles of attack, and the output is the scaled sound pressure level. The *protein* dataset [53] describes the physicochemical properties of the protein tertiary structure. The *sarcos* dataset [5] describes the inverse kinematics of a robot arm. The *chem* dataset [54] concerns the physical simulations relating to electron energies in molecules. Finally, the *sdss* dataset [55] comes from the Sloan Digital Sky Surveys 12th Data Release. Note that each configuration of n and n_* in Table 2 has ten random instances in order to comprehensively evaluate the performance of scalable GPs.

Table 2: Characteristics of five real-world datasets.

dataset	d	n	n_*	remark
airfoil	5	1,200	303	regular inputs
protein	9	35,000	10,730	clustering inputs
sarcos	21	40,000	8,933	nearly noiseless
chem	15	60,000	11,969	heteroscedastic, tiny noise
sdss	10	250,000	50,000	heteroscedastic noise

Table 3: Model parameters of scalable GPs for the five real-world datasets. m is the inducing size for VFE, SVGP and FITC, b is the batch size for SVGP, and M is the number of experts for GPoE and RBCM.

Parameter	*airfoil*	*protein*	*sarcos*	*chem*	*sdss*
m	60	400	400	300	250
b	50	2,500	2,500	3,000	9,000
M	20	35	50	80	500

Among the six global scalable GPs in Fig. 3, we select the VFE and SVGP due to the high approximation quality, and the FITC due to the capability of capturing heteroscedastic noise; the SoR, DTC and PIC are not included due to their poor or discontinuous predictions. Among the four local scalable GPs in Fig. 4, we choose the RBCM trained on experts with shared hyperparameters and the GPoE trained on experts with individual hyperparameters; the pure local GPs and PoE are not considered due to their less competitive performance.

5.2.1. Comparison of global and local scalable GPs

We first study the scalability and capability of global and local scalable GPs. Table 3 offers the model parameters of scalable GPs, including the inducing size m, the batch size b and the number M of experts, for the five real-world datasets. During the comparison study, the inducing points are initialized by the centroids of clusters partitioned by the k-means technique; the disjoint local subsets are partitioned by the k-means technique as well. We choose the model parameters in Table 3 such that these scalable GPs have comparable running time. The data pre-processing and optimization configurations are consistent to that in Section 5.1.

Fig. 6 depicts the modeling results of five scalable GPs over ten runs on the five datasets in terms of SMSE and MSLL. The horizontal axis represents the sum of training and predicting time. Note that due to the small training size, we include the results of full GP on the *airfoil* dataset for comparison.

As for the three global scalable GPs, the VFE and SVGP produce similar results since they are derived in the same posterior approximation framework. However, the SVGP is more potential in terms of efficiency for large-scale learning since it allows using the SGD optimization, see the results on the *sdss* dataset. Different from VFE and SVGP which follow a constant noise assumption, the FITC is capable of describing possible heteroscedastic noise variances, which are indicated by the smaller MSLL values in the figure. But this superiority of FITC comes at the cost of worsening the accuracy of prediction mean, resulting in larger SMSE values.

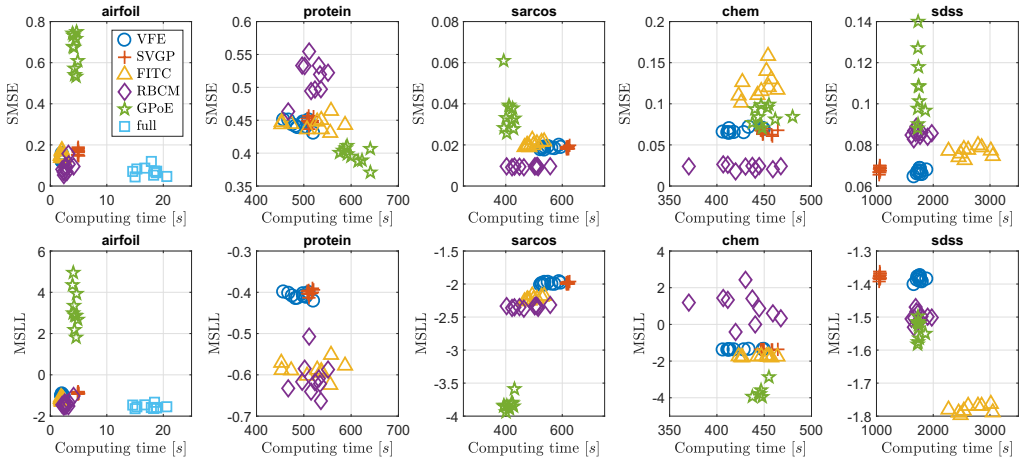


Figure 6: Comparative results of VFE, SVGP, FITC, RBCM and GPoE on the five real-world datasets. The results of full GP are provided for the *airfoil* dataset with $n = 1200$. Note that the GPoE has no MSLL symbols on the *protein* dataset since it produces invalid MSLL values in all the ten runs. Similarly, we only show five success runs of GPoE in terms of MSLL on the *chem* dataset.

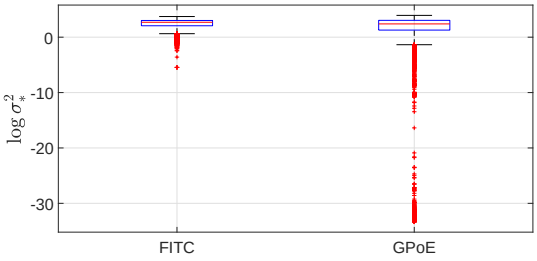


Figure 7: The boxplots of $\log \sigma_*^2$ estimated by FITC and GPoE in a run on the *protein* dataset.

As for the two local scalable GPs, due to the individual experts, the GPoE captures better predictive distributions on three out of the five datasets, and produces more accurate predictions on the *protein* dataset. However, it is observed that the GPoE yields invalid prediction variances on the *protein* and *chem* datasets due to the existence of local over-fitting. For instance, Fig. 7 depicts the $\log \sigma_*^2$ values estimated respectively by FITC and GPoE, since both of which can capture the heteroscedasticity in noise variance, on the *protein* dataset. It is observed that in comparison to FITC, the GPoE produces many extremely small prediction variances due to local over-fitting. These extreme prediction variances of GPoE, as low as nearly e^{-35}, in turn bring invalid MSLL with the value up to 6.59×10^{10}. Besides, it is observed that compared to the RBCM, the individual treatment of GP experts in GPoE often induces ill-conditioned kernel matrix, especially in the scenario with many experts. This prohibits the model inference. The local over-fitting and the ill-conditioned phenomenon of GPoE could be alleviated by using a small M, i.e., large experts that can take into account the the spatial correlations. But this would degrade the model capability and improve the complexity of GPoE. Hence, in practice

we prefer RBCM, unless there exist some tricks to guard against local over-fitting and ill-conditioned kernel matrix in GPoE.

Finally, the comparison between global and local scalable GPs shows that the local attention mechanism enables RBCM to produce smaller MSLL values than that of VFE and SVGP in four out of the five datasets, and smaller SMSE values in three out of the five datasets.

5.2.2. Impact of model parameters

It is found that the model parameters, e.g., the inducing size m and the number M of experts, affect the performance of scalable GPs. More inducing points bring better distillation of training data for sparse approximations, while more experts enhance the localization by taking into account more local patterns for local aggregations. Hence, this section seeks to run global and local scalable GPs using various model parameters in order to investigate their controllability. i.e., explicitly controlling the model performance by tuning m or M.

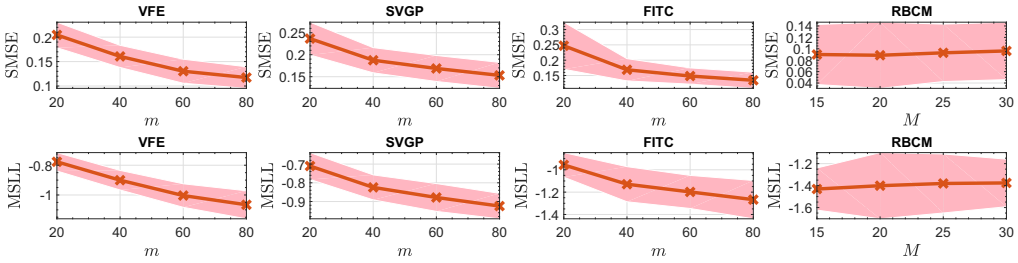


Figure 8: Impact of varying inducing size m or number M of experts on the performance of scalable GPs on the *airfoil* dataset. The shaded region represents two times the standard deviation over ten runs.

Fig. 8 depicts the modeling results of scalable GPs using different m or M values on the *airfoil* dataset. The results of scalable GPs on the remaining four datasets are provided in the Appendix. Note that the GPoE is not included in the comparison due to the unstable performance induced by local over-fitting and ill-conditioned kernel matrix.

The results in Fig. 8 and the Appendix indicate that VFE, SVGP and FITC provide better predictions with the increase of inducing size m. It is because more inducing points bring closer approximation to the full GP. This on the other hand indicates their good controllability: we could improve the predictions by simply increasing m.

However, it is interesting to observe that the RBCM provides similar predictions with different M values on not only the *airfoil* dataset but also the *protein* and *sarcos* datasets in the Appendix. This is caused by the local attention mechanism which considers local features, thus enabling RBCM to provide better and more robust estimation of the hyperparameters in comparison to VFE and FITC. More precisely, Fig. 9 depicts the estimated noise variance σ_ϵ^2 of VFE and RBCM using different model parameters on the *airfoil* dataset. As the ground truth, the noise variance estimated by the full GP on this dataset has an average value of $\sigma_\epsilon^2 = 0.0218$. It is observed that by using different M values, the RBCM is always capable of providing a small σ_ϵ^2 close to that of full GP. On the contrary, the VFE provides a too conservative σ_ϵ^2 when using a small inducing size. The estimated σ_ϵ^2 becomes closer to that of full GP with increasing m at the cost of higher computing complexity.

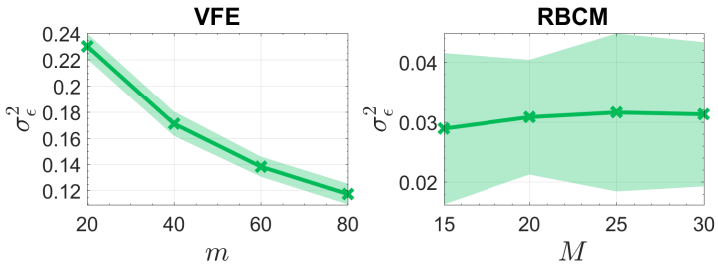


Figure 9: The estimated noise variance σ_ϵ^2 of VFE and RBCM using different model parameters on the *airfoil* dataset. The shaded region represents two times the standard deviation over ten runs.

In comparison to the conservative VFE, SVGP and FITC, the local attention helps RBCM quickly obtain a good estimation of σ_ϵ^2, which in turn encourages better predictions on some datasets. However, this is often not the case for complicated datasets with heteroscedastic noise, like the *chem* and *sdss* datasets. In these datasets, some subregions with small noise variances favor a small estimation of σ_ϵ^2, which however is not beneficial for other subregions with large noise variances. This inbalance becomes more serious when M is large, since now we are forced to handle more localized regions. For example, as shown in Fig. 14 in the Appendix, the RBCM provides poorer predictions with the increase of M.

The inconsistent performance of RBCM w.r.t. the number of experts on the five datasets indicates a poor controllability of RBCM, since it is unclear how to improve its performance by tuning M.

5.2.3. Impact of the initialization of hyperparameters

It is known that we train the GP by maximizing the marginal likelihood $p(\boldsymbol{y})$, which however is usually a non-convex optimization problem. Hence, this section investigates the robustness of scalable GPs, including VFE, FITC and RBCM, to the initialization of hyperparameters, including the SE kernel parameters $\{l_1, \cdots, l_d, \sigma_f^2\}$ and the noise variance σ_ϵ^2, on the *airfoil* dataset. Particularly, we randomly initialize the kernel parameters and the noise variance as $l_i \sim \text{random}(0, 1)$, $\sigma_f^2 \sim \text{random}(0, 1)$, and $\sigma_\epsilon^2 \sim \text{random}(0, 0.5)$ in order to produce 100 instances.

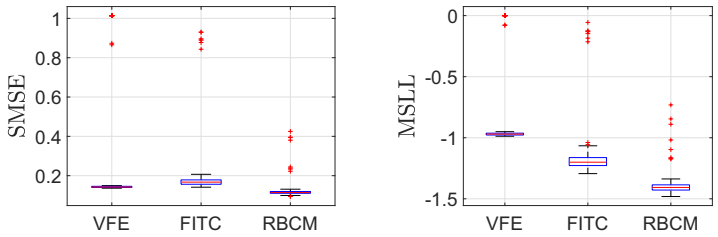


Figure 10: Impact of the initialization of hyperparameters on the performance of VFE, FITC and RBCM on the *airfoil* dataset.

Fig. 10 depicts the boxplots of the three scalable GPs with respect to 100 initializations of hyperparameters on the *airfoil* dataset. Some of the outliers in the boxplots

represent the failure runs since they provide poor SMSE (close to one) and MSLL (close to zero). Here, we define a failure run wherein the SMSE is larger than 0.8 and the MSLL value is larger than -0.3. It is observed that the VFE fails in 19 out of the 100 runs, the FITC fails in 7 runs, and the RBCM has no failure run. The results indicate that the RBCM is robust to various initializations of hyperparameters, because (i) as explained before, it uses the local attention to help estimate the hyperparameters well; and (ii) compared to VFE and FITC which have a large parameter space by considering the additional inducing parameters, the RBCM has a narrow parameter space due to the sharing of hyperparameters across experts.

6. Conclusions

This paper studies representative scalable GPs including the sparse approximations and the local aggregations on two toy examples and five large-scale real-world datasets. We summarize below their characteristics in terms of scalability, capability, robustness and controllability.

For sparse approximations including the prior and posterior approximations, we have the following findings from the numerical experiments:

- In terms of scalability, all the sparse approximations except SVGP have the same time complexity of $\mathcal{O}(nm^2)$. The SVGP reorganizes the variational lower bound such that it factorizes over data points, thus reducing the complexity to $\mathcal{O}(bm^3)$ via SGD;

- In terms of capability, most of the prior approximations provide poorer predictions than the posterior counterparts. Particularly, the FITC captures heteroscedastic noise at the cost of worsening the accuracy of prediction mean. The posterior approximations including VFE and SVGP are preferred since they are faithful approximations of full GP.

- In terms of robustness, the sparse approximations are sensitive to the initialization of hyperparameters, because of the augmented parameter space by considering inducing and variational parameters.

- In terms of controllability, it is observed that VFE, SVGP and FITC generally offer better predictions with increasing m.

For local aggregations including GPoE and RBCM, we have the following findings from the numerical experiments:

- In terms of scalability, the training complexity of local aggregations is the same as most sparse approximations when we have the training size $m_0 = m$ for each expert. But the test complexity is a bit higher since we need the predictions from M experts.

- In terms of capability, the GPoE that allows individual hyperparameters for experts is capable of capturing non-stationary features. However, with increasing M, the local over-fitting and the possible ill-conditioned kernel matrix would significantly degrade GPoE's generalization capability, rendering it impractical. Contrarily, the

RBCM (i) shares hyperparameters across experts to guard against over-fitting and (ii) estimates the hyperparameters well due to the local attention mechanism.

- In terms of robustness, the RBCM is robust to the initialization of hyperparameters on the *airfoil* dataset due to the local attention mechanism.
- In terms of controllability, it is unclear how to tune the number M of experts for RBCM.

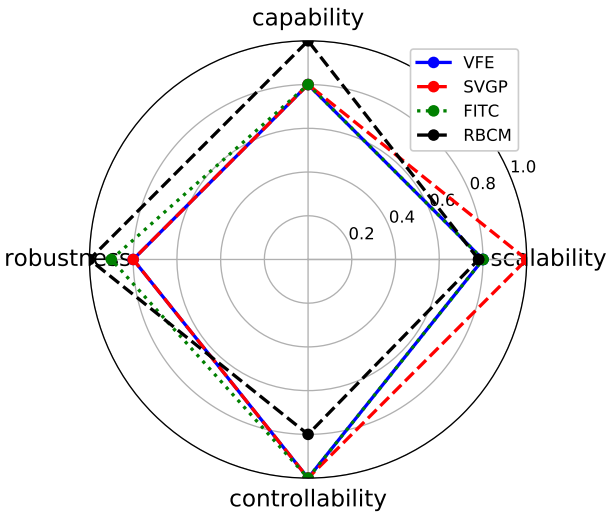


Figure 11: The radar plot to illustrate the representative global and local scalable GPs in terms of scalability, capability, robustness and controllability.

According to the above conclusions, Fig. 11 summarizes the characteristics of four successful global/local scalable GPs including VFE, SVGP, FITC and RBCM in terms of scalability, capability, robustness and controllability. To further improve the model capability while retaining the scalability, alternatively, we may combine sparse and local approximations together such that the hybrid could (i) guard against local over-fitting, and (ii) capture non-stationary features. Another promising avenue is the combination of scalable GPs and the well-known feature extractor, deep neural networks, to further boost the representational capability and scalability for big data [56, 57].

Acknowledgments

This work was conducted within the Rolls-Royce@NTU Corporate Lab with support from the National Research Foundation (NRF) Singapore under the Corp Lab@University Scheme. It is also partially supported by the Data Science and Artificial Intelligence Research Center (DSAIR) and the School of Computer Science and Engineering at Nanyang Technological University.

A. Appendix

Fig.s 12-15 below depict the impact of varying inducing size m or number M of experts on the performance of scalable GPs on the *protein*, *sarcos*, *chem* and *sdss* datasets, respectively.

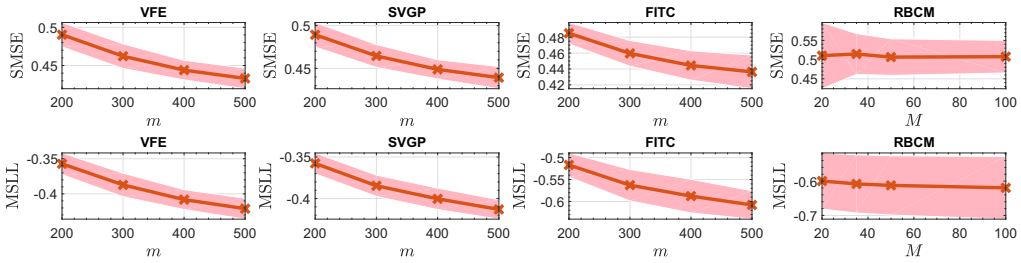


Figure 12: Impact of varying inducing size m or number M of experts on the performance of scalable GPs on the *protein* dataset.

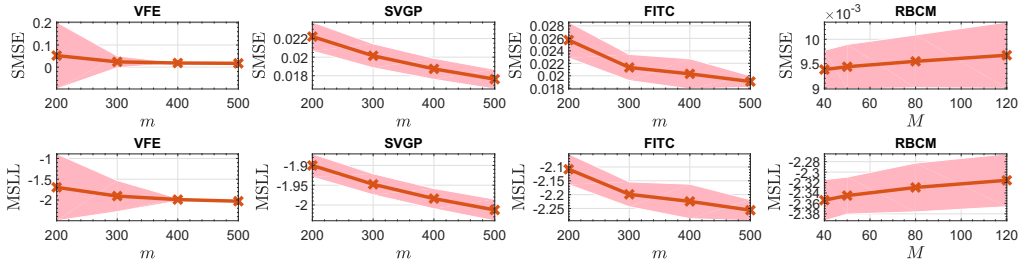


Figure 13: Impact of varying inducing size m or number M of experts on the performance of scalable GPs on the *sarcos* dataset.

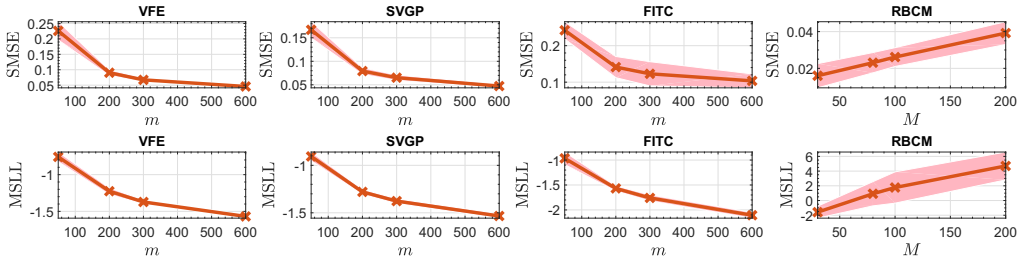


Figure 14: Impact of varying inducing size m or number M of experts on the performance of scalable GPs on the *chem* dataset.

References

References

[1] F. Duchaine, T. Morel, L. Gicquel, Computational-fluid-dynamics-based Kriging optimization tool for aeronautical combustion chambers, AIAA Journal 47 (3) (2009) 631–645.

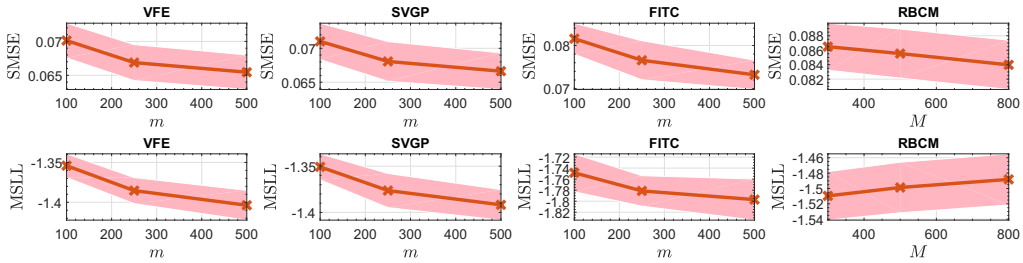


Figure 15: Impact of varying inducing size m or number M of experts on the performance of scalable GPs on the *sdss* dataset.

[2] X. Liu, Q. Zhu, H. Lu, Modeling multiresponse surfaces for airfoil design with multiple-output-Gaussian-process regression, Journal of Aircraft 51 (3) (2014) 740–747.

[3] A. Amrit, L. Leifsson, S. Koziel, Y. A. Tesfahunegn, Efficient multi-objective aerodynamic optimization by design space dimension reduction and co-Kriging, in: 17th AIAA/ISSMO Multidisciplinary Analysis and Optimization Conference, AIAA, 2016, pp. AIAA 2016–3515.

[4] N. Wagle, E. W. Frew, Forward adaptive transfer of Gaussian process regression, Journal of Aerospace Information Systems 14 (4) (2017) 214–231.

[5] C. E. Rasmussen, C. K. Williams, Gaussian processes for machine learning, MIT Press, 2006.

[6] A. Liaw, M. Wiener, et al., Classification and regression by randomForest, R news 2 (3) (2002) 18–22.

[7] Y. LeCun, Y. Bengio, G. Hinton, Deep learning, Nature 521 (May) (2015) 436–444.

[8] R. M. Neal, Bayesian learning for neural networks, Vol. 118, Springer Science & Business Media, 2012.

[9] H. Liu, Y.-S. Ong, J. Cai, A survey of adaptive sampling for global metamodeling in support of simulation-based complex engineering design, Structural and Multidisciplinary Optimization 57 (1) (2018) 393–416.

[10] D. Foreman-Mackey, E. Agol, S. Ambikasaran, R. Angus, Fast and scalable Gaussian process modeling with applications to astronomical time series, The Astronomical Journal 154 (6) (2017) 220.

[11] J. Snoek, O. Rippel, K. Swersky, R. Kiros, N. Satish, N. Sundaram, M. Patwary, M. Prabhat, R. Adams, Scalable Bayesian optimization using deep neural networks, in: International Conference on Machine Learning, 2015, pp. 2171–2180.

[12] M. P. Deisenroth, J. W. Ng, Distributed Gaussian processes, in: International Conference on Machine Learning, JMLR. org, 2015, pp. 1481–1490.

[13] K. Chalupka, C. K. Williams, I. Murray, A framework for evaluating approximation methods for Gaussian process regression, Journal of Machine Learning Research 14 (Feb) (2013) 333–350.

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

[15] E. Snelson, Z. Ghahramani, Sparse Gaussian processes using pseudo-inputs, in: Advances in Neural Information Processing Systems, 2006, pp. 1257–1264.

[16] M. K. Titsias, Variational learning of inducing variables in sparse Gaussian processes, in: Artificial Intelligence and Statistics, 2009, pp. 567–574.

[17] J. Hensman, N. Fusi, N. D. Lawrence, Gaussian processes for big data, in: Uncertainty in Artificial Intelligence, AUAI Press, 2013, pp. 282–290.

[18] T. D. Bui, R. E. Turner, Tree-structured Gaussian process approximations, in: Advances in Neural Information Processing Systems, 2014, pp. 2213–2221.

[19] G. E. Hinton, Training products of experts by minimizing contrastive divergence, Neural Computation 14 (8) (2002) 1771–1800.

[20] V. Tresp, A Bayesian committee machine, Neural Computation 12 (11) (2000) 2719–2741.

[21] Y. Cao, D. J. Fleet, Generalized product of experts for automatic and principled fusion of Gaussian process predictions, arXiv preprint arXiv:1410.7827.

[22] H. Liu, J. Cai, Y. Ong, Y. Wang, Generalized robust Bayesian committee machine for large-scale Gaussian process regression, in: International Conference on Machine Learning, JMLR. org, 2018, pp. 1–10.

[23] Y.-L. K. Samo, S. J. Roberts, String and membrane Gaussian processes, Journal of Machine Learning Research 17 (1) (2016) 4485–4571.
[24] A. J. Smola, P. L. Bartlett, Sparse greedy Gaussian process regression, in: Advances in Neural Information Processing Systems, 2001, pp. 619–625.
[25] M. Seeger, C. Williams, N. Lawrence, Fast forward selection to speed up sparse Gaussian process regression, in: Artificial Intelligence and Statistics, PMLR, 2003, pp. EPFL–CONF–161318.
[26] A. Dezfouli, E. V. Bonilla, Scalable inference for Gaussian process models with black-box likelihoods, in: Advances in Neural Information Processing Systems, 2015, pp. 1414–1422.
[27] A. Wilson, H. Nickisch, Kernel interpolation for scalable structured Gaussian processes (KISS-GP), in: International Conference on Machine Learning, 2015, pp. 1775–1784.
[28] C. E. Rasmussen, Z. Ghahramani, Infinite mixtures of Gaussian process experts, in: Advances in Neural Information Processing Systems, 2002, pp. 881–888.
[29] S. E. Yuksel, J. N. Wilson, P. D. Gader, Twenty years of mixture of experts, IEEE Transactions on Neural Networks and Learning Systems 23 (8) (2012) 1177–1193.
[30] C. K. Williams, M. Seeger, Using the Nyström method to speed up kernel machines, in: Advances in Neural Information Processing Systems, 2001, pp. 682–688.
[31] B. W. Silverman, Some aspects of the spline smoothing approach to non-parametric regression curve fitting, Journal of the Royal Statistical Society. Series B (Methodological) 47 (1) (1985) 1–52.
[32] G. Wahba, X. Lin, F. Gao, D. Xiang, R. Klein, B. Klein, The bias-variance tradeoff and the randomized GACV, in: Advances in Neural Information Processing Systems, 1999, pp. 620–626.
[33] L. Csató, M. Opper, Sparse on-line Gaussian processes, Neural Computation 14 (3) (2002) 641–668.
[34] E. Snelson, Z. Ghahramani, Local and global sparse Gaussian process approximations, in: Artificial Intelligence and Statistics, PMLR, 2007, pp. 524–531.
[35] M. K. Titsias, Variational model selection for sparse Gaussian process regression, Tech. rep., University of Manchester (2009).
[36] M. Bauer, M. van der Wilk, C. E. Rasmussen, Understanding probabilistic sparse Gaussian process approximations, in: Advances in Neural Information Processing Systems, 2016, pp. 1533–1541.
[37] A. G. d. G. Matthews, J. Hensman, R. Turner, Z. Ghahramani, On sparse variational methods and the Kullback-Leibler divergence between stochastic processes, Journal of Machine Learning Research 51 (2016) 231–239.
[38] M. D. Hoffman, D. M. Blei, C. Wang, J. Paisley, Stochastic variational inference, Journal of Machine Learning Research 14 (1) (2013) 1303–1347.
[39] M. D. Zeiler, ADADELTA: An adaptive learning rate method, arXiv preprint arXiv:1212.5701.
[40] D. P. Kingma, J. Ba, Adam: A method for stochastic optimization, arXiv preprint arXiv:1412.6980.
[41] T. N. Hoang, Q. M. Hoang, B. K. H. Low, A unifying framework of anytime sparse Gaussian process regression models with stochastic variational inference for big data, in: International Conference on Machine Learning, 2015, pp. 569–578.
[42] T. Chen, J. Ren, Bagging for Gaussian process regression, Neurocomputing 72 (7-9) (2009) 1605–1610.
[43] Y. Okadome, Y. Nakamura, Y. Shikauchi, S. Ishii, H. Ishiguro, Fast approximation method for Gaussian process regression using hash function for non-uniformly distributed data, in: International Conference on Artificial Neural Networks, Springer, 2013, pp. 17–25.
[44] B. van Stein, H. Wang, W. Kowalczyk, T. Bäck, M. Emmerich, Optimally weighted cluster Kriging for big data regression, in: International Symposium on Intelligent Data Analysis, Springer, 2015, pp. 310–321.
[45] B. van Stein, H. Wang, W. Kowalczyk, M. Emmerich, T. Bäck, Cluster-based Kriging approximation algorithms for complexity reduction, arXiv preprint arXiv:1702.01313.
[46] B. Szabo, H. van Zanten, An asymptotic analysis of distributed nonparametric methods, arXiv preprint arXiv:1711.03149.
[47] S. Mair, U. Brefeld, Distributed robust Gaussian process regression, Knowledge and Information Systems 55 (2) (2018) 415–435.
[48] Y. Gal, M. van der Wilk, C. E. Rasmussen, Distributed variational inference in sparse Gaussian process regression and latent variable models, in: Advances in Neural Information Processing Systems, Curran Associates, Inc., 2014, pp. 3257–3265.
[49] Z. Dai, A. Damianou, J. Hensman, N. Lawrence, Gaussian process models with parallelization and GPU acceleration, arXiv preprint arXiv:1410.4984.
[50] J. Vanhatalo, V. Pietiläinen, A. Vehtari, Approximate inference for disease mapping with sparse Gaussian processes, Statistics in Medicine 29 (15) (2010) 1580–1607.
[51] B.-J. Lee, J. Lee, K.-E. Kim, Hierarchically-partitioned Gaussian process approximation, in: Arti-

ficial Intelligence and Statistics, 2017, pp. 822–831.
[52] J. Hensman, N. Durrande, A. Solin, Variational fourier features for Gaussian processes, arXiv preprint arXiv:1611.06740.
[53] D. Dheeru, E. Karra Taniskidou, UCI machine learning repository (2017).
URL http://archive.ics.uci.edu/ml
[54] M. Malshe, L. Raff, M. Rockley, M. Hagan, P. M. Agrawal, R. Komanduri, Theoretical investigation of the dissociation dynamics of vibrationally excited vinyl bromide on an ab initio potential-energy surface obtained using modified novelty sampling and feedforward neural networks. II. Numerical application of the method, The Journal of Chemical Physics 127 (13) (2007) 134105.
[55] I. A. Almosallam, M. J. Jarvis, S. J. Roberts, GPz: non-stationary sparse Gaussian processes for heteroscedastic uncertainty estimation in photometric redshifts, Monthly Notices of the Royal Astronomical Society 462 (1) (2016) 726–739.
[56] W.-b. Huang, D. Zhao, F. Sun, H. Liu, E. Y. Chang, Scalable Gaussian process regression using deep neural networks., in: International Joint Conference on Artificial Intelligence, 2015, pp. 3576–3582.
[57] A. G. Wilson, Z. Hu, R. Salakhutdinov, E. P. Xing, Deep kernel learning, in: Artificial Intelligence and Statistics, 2016, pp. 370–378.