

When Gaussian Process Meets Big Data: A Review of Scalable GPs

Haitao Liu, Yew-Soon Ong, *Fellow, IEEE*, Xiaobo Shen, and Jianfei Cai, *Senior Member, IEEE*

Abstract—The vast quantity of information brought by big data as well as the evolving computer hardware encourages success stories in the machine learning community. In the meanwhile, it poses challenges for the Gaussian process (GP) regression, a well-known non-parametric and interpretable Bayesian model, which suffers from cubic complexity to data size. To improve the scalability while retaining desirable prediction quality, a variety of scalable GPs have been presented. But they have not yet been comprehensively reviewed and analyzed in order to be well understood by both academia and industry. The review of scalable GPs in the GP community is timely and important due to the explosion of data size. To this end, this paper is devoted to the review on state-of-the-art scalable GPs involving two main categories: **global approximations which distillate the entire data and local approximations which divide the data for subspace learning**. Particularly, for global approximations, we mainly focus on sparse approximations comprising prior approximations which modify the prior but perform exact inference, posterior approximations which retain exact prior but perform approximate inference, and structured sparse approximations which exploit specific structures in kernel matrix; for local approximations, we highlight the mixture/product of experts that conducts model averaging from multiple local experts to boost predictions. To present a complete review, recent advances for improving the scalability and capability of scalable GPs are reviewed. Finally, the extensions and open issues regarding the implementation of scalable GPs in various scenarios are reviewed and discussed to inspire novel ideas for future research avenues.

Index Terms—Gaussian process regression, big data, scalability, sparse approximations, local approximations

I. INTRODUCTION

IN the era of big data, the vast quantity of information poses the demand of effective and efficient analysis, interpretation and prediction to explore the benefits lie ahead. Thanks to the big data, the machine learning community tells many success stories [1]–[4] while still leaving many challenges. We focus on Gaussian process (GP) regression [5], also known as Kriging in geostatistics [6], and surrogates or emulators in computer experiments [7]. The GP is a non-parametric statistical model which has been extensively used in various scenarios, e.g., active learning [8], multi-task learning [9], [10], manifold learning [11], and optimization [12].

Big data in the GP community mainly refers to one of the 5V challenges [13]: the *volume* which represents the huge amount of data points to be stored, processed and analyzed, incurring high computational complexity for current

GP paradigms. It is worth noting that this review mainly focuses on scalable GPs for large-scale regression but not on all forms of GPs or other machine learning models.

Given n training points $\mathbf{X} = \{\mathbf{x}_i \in \mathbb{R}^d\}_{i=1}^n$ and their observations $\mathbf{y} = \{y_i = y(\mathbf{x}_i) \in \mathbb{R}\}_{i=1}^n$, GP seeks to infer the latent function $f : \mathbb{R}^d \mapsto \mathbb{R}$ in the function space $\mathcal{GP}(m(\mathbf{x}), k(\mathbf{x}, \mathbf{x}'))$ defined by the mean $m(\cdot)$ and the kernel $k(\cdot, \cdot)$. The most prominent weakness of standard GP is that it suffers from a cubic time complexity $\mathcal{O}(n^3)$ because of **the inversion and determinant of the $n \times n$ kernel matrix $\mathbf{K}_{nn} = k(\mathbf{X}, \mathbf{X})$** . This limits the scalability of GP and makes it unaffordable for large-scale datasets.

Hence, scalable GPs devote to *improving the scalability of full GP while retaining favorable prediction quality* for big data. The extensive literature review summarized in Fig. 1 classifies scalable GPs into two main categories including

(a) *Global approximations* which approximate the kernel matrix \mathbf{K}_{nn} through global distillation. The distillation can be achieved by (i) a subset of the training data with m ($m \ll n$) points (subset-of-data [14]), resulting in a smaller kernel matrix \mathbf{K}_{mm} ; (ii) the remove of uncorrelated entries in \mathbf{K}_{nn} (sparse kernels [15]), resulting in a sparse kernel matrix $\tilde{\mathbf{K}}_{nn}$ with many zero entries; and (iii) the low-rank representation measured between m inducing points and n training points (sparse approximations [1], [16]–[18]), resulting in the **Nyström approximation $\mathbf{K}_{nn} \approx \mathbf{K}_{nm} \mathbf{K}_{mm}^{-1} \mathbf{K}_{mn}$** .

(b) *Local approximations* which follow the *divide-and-conquer* (D&C) idea to focus on the local subsets of training data. Efficiently, local approximations only need to tackle a *local expert* with m_0 ($m_0 \ll n$) data points at each time [19], [20]. Additionally, to produce smooth predictions equipped with valid uncertainty, modeling averaging has been employed through mixture or product of experts [21]–[28].

As depicted in Fig. 2, in terms of scalability, most of the sparse approximations using m inducing points and the local approximations using $m_0 = m$ data points for each expert have the same training complexity as $\mathcal{O}(nm^2)$, and they can be further sped up through parallel/distributed computing [20], [29]–[33]. When organizing the inducing points into Kronecker structure, sparse approximations can further reduce the complexity to $\mathcal{O}(n)$ [18], [34]. In the meantime, by reorganizing the variational lower bound, stochastic optimization is available for sparse approximations with a remarkable complexity of $\mathcal{O}(m^3)$ [1], [35], [36], enabling the regression with million- and even billion-sized data points [36], [37].

It is notable that we welcome GPs with high scalability but require producing favorable predictions, i.e., good model capability. For example, though showing a remarkable complexity

Haitao Liu is with the Rolls-Royce@NTU Corporate Lab, Nanyang Technological University, Singapore, 637460. E-mail: htliu@ntu.edu.sg

Yew-Soon Ong, Xiaobo Shen and Jianfei Cai are with School of Computer Science and Engineering, Nanyang Technological University, Singapore, 639798. E-mail: {asysong, xbshen, asjfc}@ntu.edu.sg.

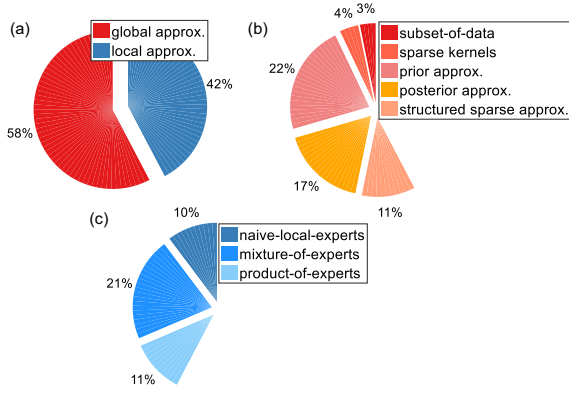


Fig. 1. Percentages of the categories for (a) scalable GPs including (b) global approximations and (c) local approximations in the literature surveyed.

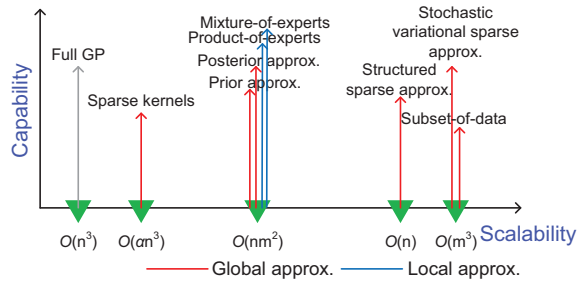


Fig. 2. Comparison of scalable GPs regarding scalability and model capability, where $0 < \alpha < 1$; m is the inducing size for sparse approximations, and the subset size for subset-of-data and local approximations.

of $\mathcal{O}(m^3)$, we cannot expect the subset-of-data to perform well with increasing n . In terms of model capability, global approximations are capable of capturing the global patterns (long-term spatial correlations) but often filter out the local patterns due to the limited global inducing set. In contrast, due to the local nature, local approximations favor capturing local patterns (non-stationary features), enabling them to outperform global approximations for complicated tasks, see the *solar* example in [38]. **The drawback however is that they ignore the global patterns to risk discontinuous predictions and local over-fitting.** Recently, attempts have been made to improve the model capability through, for example, the inter-domain strategy [39], hierarchical structure [40], and hybrid of global & local approximations or neural networks (NNs) [34], [41], [42], showcasing the state-of-the-art performance [34], [43].

The development and success of scalable GPs pose the demand of comprehensive review including the methodological characteristics and comparisons for better understanding. To the best of our knowledge, a detailed survey on various scalable GPs for large-scale regression has not been conducted in the literature before and such a work in the GP community is timely and important due to the explosion of data size.¹

We thus consider a skeletal overview in Fig. 3 to classify, review and analyze state-of-the-art scalable GPs. Specifically,

¹The survey [16] at 14 years ago focuses on the prior approximations, which is just a part of our review in section III-C1. The recent comparison and survey [37], [44] provide however a quick and rough review without detailed analysis.

with a quick introduction of standard GP regression in section II, the two main categories of scalable GPs, global and local approximations, are then comprehensively reviewed in sections III and IV. Moreover, section V reviews the improvements for scalable GPs in terms of scalability and capability. Thereafter, section VI discusses the extensions of scalable GPs in different scenarios to highlight potential research avenues. Finally, section VII offers concluding remarks.

II. GAUSSIAN PROCESS REGRESSION REVISITED

The non-parametric GP regression (GPR) places a GP prior over the latent function as $f(\mathbf{x}) \sim \mathcal{GP}(m(\mathbf{x}), k(\mathbf{x}, \mathbf{x}'))$ [5]. The mean function $m(\mathbf{x})$ is often taken as zero. The kernel function $k(\mathbf{x}, \mathbf{x}')$ controls the smoothness of GP and is often taken as the squared exponential (SE) function equipped with automatic relevance determination (ARD)

$$k_{\text{SE}}(\mathbf{x}, \mathbf{x}') = \sigma_f^2 \exp(-0.5(\mathbf{x} - \mathbf{x}')^\top \mathbf{\Delta}^{-1}(\mathbf{x} - \mathbf{x}')), \quad (1)$$

where $\mathbf{\Delta} = \text{diag}[l_1^2, \dots, l_d^2]$ comprises the length-scales along d dimensions, and σ_f^2 is the signal variance. For other conventional kernels, e.g., the Matérn kernel, please refer to [5].

Given the training data $\mathcal{D} = \{\mathbf{X}, \mathbf{y}\}$ where $y(\mathbf{x}_i) = f(\mathbf{x}_i) + \epsilon$ with the iid noise $\epsilon \sim \mathcal{N}(0, \sigma_\epsilon^2)$, we obtain the model evidence (marginal likelihood) $p(\mathbf{y}|\boldsymbol{\theta}) = \int p(\mathbf{y}|\mathbf{f})p(\mathbf{f})d\mathbf{f} = \mathcal{N}(\mathbf{y}|\mathbf{0}, \mathbf{K}_{nn}^\epsilon)$,² where $\mathbf{K}_{nn}^\epsilon = \mathbf{K}_{nn} + \sigma_\epsilon^2 \mathbf{I}_n$, and $\boldsymbol{\theta}$ comprises the hyperparameters which could be inferred by maximizing

$$\log p(\mathbf{y}) = -\frac{n}{2} \log 2\pi - \frac{1}{2} \log |\mathbf{K}_{nn}^\epsilon| - \frac{1}{2} \mathbf{y}^\top (\mathbf{K}_{nn}^\epsilon)^{-1} \mathbf{y}, \quad (2)$$

which **automatically achieves the bias-variance trade-off.**

Thereafter, the predictive distribution $p(f_*|\mathcal{D}, \mathbf{x}_*) = \mathcal{N}(f_*|\mu(\mathbf{x}_*), \sigma_*^2(\mathbf{x}_*))$ at a test point \mathbf{x}_* has the mean and variance respectively expressed as

$$\mu(\mathbf{x}_*) = \mathbf{k}_{*n} (\mathbf{K}_{nn}^\epsilon)^{-1} \mathbf{y}, \quad (3a)$$

$$\sigma^2(\mathbf{x}_*) = k_{**} - \mathbf{k}_{*n} (\mathbf{K}_{nn}^\epsilon)^{-1} \mathbf{k}_{n*}, \quad (3b)$$

where $\mathbf{k}_{*n} = k(\mathbf{x}_*, \mathbf{X})$ and $k_{**} = k(\mathbf{x}_*, \mathbf{x}_*)$. For y_* , we need to consider the noise such that $p(y_*|\mathcal{D}, \mathbf{x}_*) = \mathcal{N}(y_*|\mu(\mathbf{x}_*), \sigma_*^2(\mathbf{x}_*) + \sigma_\epsilon^2)$.

Alternatively, we can interpret the GP from the *weight-space* view as an extension of the Bayesian linear model as

$$f(\mathbf{x}) = \phi(\mathbf{x})^\top \mathbf{w}, \quad y(\mathbf{x}) = f(\mathbf{x}) + \epsilon, \quad (4)$$

where the Gaussian prior is placed on the weights as $p(\mathbf{w}) = \mathcal{N}(\mathbf{w}|\mathbf{0}, \mathbf{\Sigma})$; $\phi(\mathbf{x}) = [\phi_1(\mathbf{x}), \dots, \phi_v(\mathbf{x})]^\top$ maps the d -dimensional input \mathbf{x} into a v -dimensional feature space. Equivalently, we derive the kernel as $k(\mathbf{x}, \mathbf{x}') = \phi(\mathbf{x})^\top \mathbf{\Sigma} \phi(\mathbf{x}')$. Particularly, the SE kernel (1) can be recovered from an infinite number ($v \rightarrow \infty$) of Gaussian-shaped basis functions $\{\phi_c(\mathbf{x})\}_{c=1}^v$ centered everywhere.

The computational bottleneck of GP inference in (2) is solving the linear system $(\mathbf{K}_{nn}^\epsilon)^{-1} \mathbf{y}$ and the determinant $|\mathbf{K}_{nn}^\epsilon|$. Traditionally, we use the $\mathcal{O}(n^3)$ **Cholesky decomposition** $\mathbf{K}_{nn}^\epsilon = \mathbf{L}\mathbf{L}^\top$ such that $(\mathbf{K}_{nn}^\epsilon)^{-1} \mathbf{y} = \mathbf{L}^\top \setminus (\mathbf{L} \setminus \mathbf{y})$ and $\log |\mathbf{K}_{nn}^\epsilon| = 2 \sum_{i=1}^n \log L_{ii}$. As for predictions in (3),

²For the sake of clarity, the hyperparameters $\boldsymbol{\theta}$ below are omitted from the conditioning of the distribution.

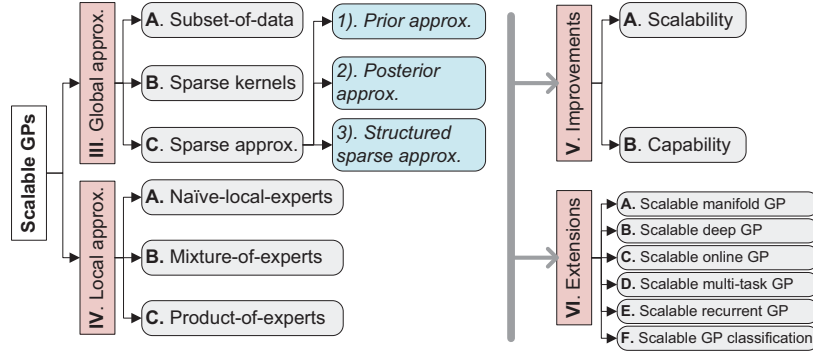


Fig. 3. A skeletal overview of scalable GPs.

the mean costs $\mathcal{O}(n)$ and the variance costs $\mathcal{O}(n^2)$ per test case through pre-computations.

In order to improve the scalability of standard GP for big data, the scalable GPs have been extensively presented and studied in recent years. In what follows, we classify current scalable GPs into global approximations and local approximations, and comprehensively analyze them to showcase their methodological characteristics.

III. GLOBAL APPROXIMATIONS

Global approximations achieve the sparsity of the full kernel matrix \mathbf{K}_{nn} , which is crucial for scalability, through (i) using a subset of the training data (subset-of-data); (ii) removing the entries of \mathbf{K}_{nn} with low correlations (sparse kernels); and (iii) employing a low-rank representation (sparse approximations).

A. Subset-of-data

Subset-of-data (SoD) is the simplest strategy to approximate the full GP by using a subset \mathcal{D}_{sod} of the training data \mathcal{D} . Hence, the SoD retains the standard GP inference at lower time complexity of $\mathcal{O}(m^3)$, since it operates on \mathbf{K}_{mm} which only comprises m ($m \ll n$) data points. A recent theoretical work [45] analyzes the error bounds for the prediction and generalization of SoD through a graphon-based framework, indicating a better speed-accuracy trade-off in comparison to other approximations reviewed below when n is sufficiently large. Though SoD produces reasonable prediction mean for the case with redundant data, it struggles to produce overconfident prediction variance due to the limited subset.

Regarding the selection of \mathcal{D}_{sod} , one could (i) randomly choose m data points from \mathcal{D} , (ii) use clustering techniques, e.g., k -means and KD tree [46], to partition the data into m subsets and choose their centroids as subset points, and (iii) employ active learning criteria, e.g., **differential entropy** [47], **information gain** [48] and **matching pursuit** [49], to sequentially query data points with however higher computing cost.

B. Sparse kernels

Sparse kernels [50] attempt to directly achieve a sparse representation $\tilde{\mathbf{K}}_{nn}$ of \mathbf{K}_{nn} via the particularly designed compactly supported (CS) kernel, which imposes $k(\mathbf{x}_i, \mathbf{x}_j) = 0$ when $|\mathbf{x}_i - \mathbf{x}_j|$ exceeds a certain threshold. Therefore, only the

non-zero elements in $\tilde{\mathbf{K}}_{nn}$ are involved in the calculation. As a result, the training complexity of the GP using CS kernel scales as $\mathcal{O}(\alpha n^3)$ with $0 < \alpha < 1$. **The main challenge in constructing valid CS kernels is to ensure the positive semi-definite (PSD) of $\tilde{\mathbf{K}}_{nn}$, i.e., $\mathbf{v}^\top \tilde{\mathbf{K}}_{nn} \mathbf{v} \geq 0, \forall \mathbf{v} \in \mathbb{R}^n$** [15], [50]–[52]. Besides, the GP using CS kernel is potential for capturing local patterns due to the truncation property.

C. Sparse approximations

Typically, we could conduct eigen-decomposition and choose the first m eigenvalues to approximate the full-rank kernel matrix as $\mathbf{K}_{nn} \approx \mathbf{U}_{nm} \mathbf{\Lambda}_{mm} \mathbf{U}_{nm}^\top$. Thereafter, it is straightforward to calculate the inversion using the **Sherman-Morrison-Woodbury formula**

$$(\mathbf{K}_{nn}^\epsilon)^{-1} \approx \sigma_\epsilon^{-2} \mathbf{I}_n + \sigma_\epsilon^{-2} \mathbf{U}_{nm} (\sigma_\epsilon^2 \mathbf{\Lambda}_{mm}^{-1} + \mathbf{U}_{nm}^\top \mathbf{U}_{nm})^{-1} \mathbf{U}_{nm}^\top,$$

and the determinant using the Sylvester determinant theorem

$$|\mathbf{K}_{nn}^\epsilon| \approx |\mathbf{\Lambda}_{mm}| |\sigma_\epsilon^2 \mathbf{\Lambda}_{mm}^{-1} + \mathbf{U}_{nm}^\top \mathbf{U}_{nm}|,$$

resulting in the complexity of $\mathcal{O}(nm^2)$. However, the eigen-decomposition is of limited interest since itself is an $\mathcal{O}(n^3)$ operation. Hence, we approximate the eigen-functions of \mathbf{K}_{nn} using m data points, leading to the Nyström approximation

$$\mathbf{K}_{nn} \approx \mathbf{Q}_{nn} = \mathbf{K}_{nm} \mathbf{K}_{mm}^{-1} \mathbf{K}_{nm}^\top,$$

which greatly improves large-scale kernel learning [53], and enables naive Nyström GP [54]. **This scalable GP however may produce negative prediction variances** [55], since (i) it is not a complete generative probabilistic model as the Nyström approximation is only imposed on the training data, and (ii) it cannot guarantee the PSD of kernel matrix.

Inspired by the influential Nyström approximation, sparse approximations build a generative probabilistic model, which achieves the sparsity via m inducing points (also referred to as support points, active set or pseudo points) to optimally summarize the dependency of the whole training data. We introduce a set of inducing pairs $(\mathbf{X}_m, \mathbf{f}_m)$. The latent variables \mathbf{f}_m akin to \mathbf{f} follow the same GP prior $p(\mathbf{f}_m) = \mathcal{N}(\mathbf{0}, \mathbf{K}_{mm})$. Besides, \mathbf{f}_m is assumed to be a sufficient statistic for \mathbf{f} , i.e., for any variables \mathbf{z} it holds $p(\mathbf{z}|\mathbf{f}, \mathbf{f}_m) = p(\mathbf{z}|\mathbf{f}_m)$. We could recover the joint prior $p(\mathbf{f}, \mathbf{f}_*)$ by marginalizing out \mathbf{f}_m as $p(\mathbf{f}, \mathbf{f}_*) = \int p(\mathbf{f}, \mathbf{f}_*|\mathbf{f}_m)p(\mathbf{f}_m)d\mathbf{f}_m$.

In what follows, sparse approximations have three main categories:

- *prior approximations* which approximate the prior but perform exact inference;
- *posterior approximations* which retain exact prior but perform approximate inference; and
- *structured sparse approximations* which exploit specific structures in kernel matrix.

1) **Prior approximations:** Prior approximations [16] modify the joint prior, which is the origin of the cubic complexity, using the independence assumption $\mathbf{f} \perp \mathbf{f}_* | \mathbf{f}_m$ such that

$$p(\mathbf{f}, \mathbf{f}_*) = \int p(\mathbf{f} | \mathbf{f}_m) p(\mathbf{f}_* | \mathbf{f}_m) p(\mathbf{f}_m) d\mathbf{f}_m, \quad (5)$$

where the training and test conditionals write, given a Nyström notation $\mathbf{Q}_{ab} = \mathbf{K}_{am} \mathbf{K}_{mm}^{-1} \mathbf{K}_{mb}$,

$$p(\mathbf{f} | \mathbf{f}_m) = \mathcal{N}(\mathbf{f} | \mathbf{K}_{nm} \mathbf{K}_{mm}^{-1} \mathbf{f}_m, \mathbf{K}_{nn} - \mathbf{Q}_{nn}), \quad (6a)$$

$$p(\mathbf{f}_* | \mathbf{f}_m) = \mathcal{N}(\mathbf{f}_* | \mathbf{k}_* \mathbf{K}_{mm}^{-1} \mathbf{f}_m, \mathbf{k}_{**} - \mathbf{Q}_{**}). \quad (6b)$$

We see here \mathbf{f}_m is called inducing variables since the dependencies between \mathbf{f} and \mathbf{f}_* are only induced through \mathbf{f}_m . To obtain computational gains, we modify the training and test conditionals as

$$q(\mathbf{f} | \mathbf{f}_m) = \mathcal{N}(\mathbf{f} | \mathbf{K}_{nm} \mathbf{K}_{mm}^{-1} \mathbf{f}_m, \tilde{\mathbf{Q}}_{nn}), \quad (7a)$$

$$q(\mathbf{f}_* | \mathbf{f}_m) = \mathcal{N}(\mathbf{f}_* | \mathbf{k}_* \mathbf{K}_{mm}^{-1} \mathbf{f}_m, \tilde{\mathbf{Q}}_{**}). \quad (7b)$$

Then, $\log p(\mathbf{y})$ is approximated by $\log q(\mathbf{y})$ as

$$\begin{aligned} \log q(\mathbf{y}) = & -\frac{n}{2} \log 2\pi - \frac{1}{2} \log |\tilde{\mathbf{Q}}_{nn} + \mathbf{Q}_{nn} + \sigma_\epsilon^2 \mathbf{I}_n| \\ & - \frac{1}{2} \mathbf{y}^\top (\tilde{\mathbf{Q}}_{nn} + \mathbf{Q}_{nn} + \sigma_\epsilon^2 \mathbf{I}_n)^{-1} \mathbf{y}. \end{aligned} \quad (8)$$

It is found that specific selections of $\tilde{\mathbf{Q}}_{nn}$ enable calculating $|\tilde{\mathbf{Q}}_{nn} + \mathbf{Q}_{nn} + \sigma_\epsilon^2 \mathbf{I}_n|$ and $(\tilde{\mathbf{Q}}_{nn} + \mathbf{Q}_{nn} + \sigma_\epsilon^2 \mathbf{I}_n)^{-1}$ with a substantially reduced complexity of $\mathcal{O}(nm^2)$.

Particularly, the subset-of-regressors (SoR) [56], also called deterministic inducing conditional (DIC), imposes deterministic training and test conditionals, i.e., $\tilde{\mathbf{Q}}_{nn} = \mathbf{0}$ and $\tilde{\mathbf{Q}}_{**} = \mathbf{0}$, as

$$q_{\text{SoR}}(\mathbf{f} | \mathbf{f}_m) = \mathcal{N}(\mathbf{f} | \mathbf{K}_{nm} \mathbf{K}_{mm}^{-1} \mathbf{f}_m, \mathbf{0}), \quad (9a)$$

$$q_{\text{SoR}}(\mathbf{f}_* | \mathbf{f}_m) = \mathcal{N}(\mathbf{f}_* | \mathbf{k}_* \mathbf{K}_{mm}^{-1} \mathbf{f}_m, \mathbf{0}). \quad (9b)$$

This is equivalent to applying the Nyström approximation to both training and test data, resulting in a **degenerate**³ GP with a rank (at most) m kernel

$$k_{\text{SoR}}(\mathbf{x}_i, \mathbf{x}_j) = k(\mathbf{x}_i, \mathbf{X}_m) \mathbf{K}_{mm}^{-1} k(\mathbf{X}_m, \mathbf{x}_j).$$

Alternatively, we could interpret the SoR from the weight-space view. It is known that the GP using a kernel with an infinite expansion of the input \mathbf{x} in the feature space defined by dense basis functions $\{\phi_c(\mathbf{x})\}_{c=1}^\infty$ is equivalent to a Bayesian linear model in (4) with infinite weights. Hence, the relevance vector machine (RVM) [57] uses only m basis functions $\phi_m(\mathbf{x}) = [\phi_1(\mathbf{x}), \dots, \phi_m(\mathbf{x})]^\top$ for approximation

$$p(\mathbf{f} | \mathbf{w}) = \mathcal{N}(\mathbf{f} | \Phi_{nm} \mathbf{w}, \mathbf{K}_{nn} - \Phi_{nm} \Sigma_{mm} \Phi_{nm}^\top), \quad (10)$$

³It means the kernel $k(\cdot, \cdot)$ has a finite number of non-zero eigenvalues.

where $\Phi_{nm} = [\phi_m(\mathbf{x}_1), \dots, \phi_m(\mathbf{x}_n)]^\top$ and $p(\mathbf{w}) = \mathcal{N}(\mathbf{w} | \mathbf{0}, \Sigma_{mm})$. As a consequence, from the function-space view, the RVM is a GP with the kernel

$$k_{\text{RVM}}(\mathbf{x}_i, \mathbf{x}_j) = \phi^\top(\mathbf{x}_i) \Sigma_{mm} \phi(\mathbf{x}_j),$$

which recovers k_{SoR} when $\Sigma_{mm} = \mathbf{I}_m$ and $\phi_m(\mathbf{x}) = \mathbf{L}^\top k^\top(\mathbf{x}, \mathbf{X}_m)$ where $\mathbf{L} \mathbf{L}^\top = \mathbf{K}_{mm}^{-1}$ [36].⁴ However, as depicted in Fig. 4, the SoR approximation and the RVM-type models [57]–[59] impose too restrictive assumptions to the training and test data such that they produce *overconfident* prediction variances when leaving the training data.⁵

To reverse the uncertainty behavior of SoR, the RVM is healed through augmenting the basis functions at \mathbf{x}_* with however higher computing cost [60]. This augmentation by including \mathbf{f}_* into \mathbf{f}_m was also studied in [16]. Alternatively, the sparse spectrum GP (SSGP) [61] and its variational variants [62]–[64] elegantly address this issue by reconstructing the Bayesian linear model from spectral representation (Fourier features), resulting in the *stationary* kernel

$$k(\mathbf{x}_i, \mathbf{x}_j) = \frac{\sigma_0^2}{m} \phi_m^\top(\mathbf{x}_i) \phi_m(\mathbf{x}_j) = \frac{\sigma_0^2}{m} \sum_{r=1}^m \cos(2\pi \mathbf{s}_r^\top (\mathbf{x}_i - \mathbf{x}_j)),$$

where $\mathbf{s}_r \in \mathbb{R}^d$ represents the spectral frequencies.

Another way is to impose more informative assumption to $\tilde{\mathbf{Q}}_{nn}$ and $\tilde{\mathbf{Q}}_{**}$. For instance, the deterministic training conditional (DTC) [65], [66] imposes the deterministic training conditional

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

but retains the exact test conditional. Hence, the prediction mean is the same as that of SoR, but the prediction variance is always larger than that of SoR, and grows to the prior when leaving the inducing points, see Fig. 4. Notably, due to the inconsistent conditionals in (11), the DTC is not an exact GP. Besides, the DTC and SoR often perform not so well due to the restrictive prior assumption $\tilde{\mathbf{Q}}_{nn} = \mathbf{0}$.⁶

Alternatively, the fully independent training conditional (FITC) [67] imposes another fully independence assumption to remove the dependency among $\{f_i\}_{i=1}^n$ such that given $\mathbf{V}_{nn} = \mathbf{K}_{nn} - \mathbf{Q}_{nn}$, the training conditional $q_{\text{FITC}}(\mathbf{f} | \mathbf{f}_m)$

$$:= \prod_{i=1}^n p(f_i | \mathbf{f}_m) = \mathcal{N}(\mathbf{f} | \mathbf{K}_{nm} \mathbf{K}_{mm}^{-1} \mathbf{f}_m, \text{diag}[\mathbf{V}_{nn}]), \quad (12)$$

whereas the test conditional retains exact. It is found that the variances of (12) is identical to that of $p(\mathbf{f} | \mathbf{f}_m)$ due to the correlation $\tilde{\mathbf{Q}}_{nn} = \text{diag}[\mathbf{V}_{nn}]$. Hence, compared to SoR and DTC which throw away the uncertainty in (9) and (11), FITC partially retains it, leading to a closer approximation to the

⁴The choice of $\phi_m(\mathbf{x})$ should produce a PSD kernel matrix such that $\mathbf{K}_{nn} - \Phi_{nm} \Sigma_{mm} \Phi_{nm}^\top \succeq \mathbf{0}$. Alternatively, we can take the simple form $\phi_m(\mathbf{x}) = \Lambda_{mm} \mathbf{k}_m^\top(\mathbf{x})$ where Λ_{mm} is a diagonal matrix and $\mathbf{k}_m(\mathbf{x}) = k(\mathbf{x}, \mathbf{X}_m)$ [57]; or we take $\phi_m(\mathbf{x}) = \Lambda^{1/2} \mathbf{U}^\top \mathbf{k}_m^\top(\mathbf{x})$ with Λ and \mathbf{U} respectively being the eigenvalue matrix and eigenvector matrix of \mathbf{K}_{mm}^{-1} [58], leading to a scaled Nyström approximation.

⁵The degenerate kernels k_{SoR} and k_{RVM} only have m degrees of freedom, and suffer from the odd property of depending on inputs.

⁶This could be addressed by the variational variant of DTC reviewed in section III-C2.

prior $p(\mathbf{f}, \mathbf{f}_*)$. Moreover, the fully independence assumption can be extended to $q(\mathbf{f}_*|\mathbf{f}_m)$ to derive the fully independent conditional (FIC)⁷ [16], which **stands as a non-degenerate GP** with the kernel

$$k_{\text{FIC}}(\mathbf{x}_i, \mathbf{x}_j) = k_{\text{SoR}}(\mathbf{x}_i, \mathbf{x}_j) + \delta_{ij}[k(\mathbf{x}_i, \mathbf{x}_j) - k_{\text{SoR}}(\mathbf{x}_i, \mathbf{x}_j)],$$

where δ_{ij} is the Kronecker's delta. Note that k_{FIC} has a constant **prior variance but is not stationary**. Alternatively, the approximation in (12) can be derived from minimizing the Kullback-Leibler (KL) divergence $\text{KL}(p(\mathbf{f}, \mathbf{f}_m)||q(\mathbf{f}_m) \prod_{i=1}^n q(\mathbf{f}_i|\mathbf{f}_m))$ [68], which quantifies the similarity between the exact and approximated joint prior.

Particularly, the FITC produces prediction mean and variance at \mathbf{x}_* as $\mu(\mathbf{x}_*) = \mathbf{k}_{*m} \Psi \mathbf{K}_{mn} \Lambda^{-1} \mathbf{y}$ and $\sigma^2(\mathbf{x}_*) = k_{**} - \mathbf{Q}_{**} + \mathbf{k}_{*m} \Psi \mathbf{K}_{m*}$ where $\Psi^{-1} = \mathbf{K}_{mm} + \mathbf{K}_{mn} \Xi^{-1} \mathbf{K}_{nm}$ and $\Xi = \text{diag}[\mathbf{V}_{nn}] + \sigma_\epsilon^2 \mathbf{I}_n$. It is found that the diagonal correlation $\text{diag}[\mathbf{V}_{nn}]$ represents the posterior variances of \mathbf{f} given \mathbf{f}_m . Hence, these varying variances, which are zeros exactly at \mathbf{X}_m , enable FITC to capture the noise heteroscedasticity, see Fig. 4, at the cost of (i) producing an invalidate estimation (nearly zero) of the noise variance σ_ϵ^2 , and (ii) sacrificing the accuracy of prediction mean [69].

To improve FITC, the partially independent training conditional (PITC) [16] has the training conditional $q_{\text{PITC}}(\mathbf{f}|\mathbf{f}_m)$

$$:= \prod_{i=1}^M p(\mathbf{f}_i|\mathbf{f}_m) = \mathcal{N}(\mathbf{f}|\mathbf{K}_{nm} \mathbf{K}_{mm}^{-1} \mathbf{f}_m, \text{blkdiag}[\mathbf{V}_{nn}]). \quad (13)$$

This equates to partitioning the training data \mathcal{D} into M independent subsets (blocks) $\{\mathcal{D}_i\}_{i=1}^M$, and taking into account the joint distribution of \mathbf{f}_i in each subset. But it is argued that though being a closer approximation to $p(\mathbf{f}|\mathbf{f}_m)$, **the blocking $q_{\text{PITC}}(\mathbf{f}|\mathbf{f}_m)$ brings little improvements over FITC** [41]. This issue can be addressed by the extended partially independent conditional (PIC) [41] discussed in section V-B.

So far, we have reviewed state-of-the-art prior approximations including SoR, DTC, FI(T)C and PITC. Regarding their implementations, the choose of inducing points is crucial. Alternatively, similar to SoD, we could use clustering techniques to select a finite set of space-filling inducing points from \mathcal{D} , or we employ some querying criteria [49], [56], [66], [70], [71] to sequentially choose informative inducing points. More flexibly, inducing points are regarded as parameters to be optimized together with other hyperparameters [67], which additionally introduces $m \times d$ parameters and **turns the inference into a high-dimensional optimization task**. Besides, **with increasing m , the benefits brought by the optimization over the simple selection from training data vanish**. Interestingly, a recent work [72] shows the first attempt to simultaneously determine the number and locations of inducing points in the Bayesian framework by placing a prior on \mathbf{X}_m .

Finally, the heteroscedasticity of FITC raises another finding that this approximation attempts to achieve a desirable predictive accuracy at low computing cost, rather than faithfully recovering the standard GP with increasing m . Indeed, the

prior approximations recover the full GP when $\mathbf{X}_m = \mathbf{X}$. But this configuration is not the global optimum when maximizing $\log q(\mathbf{y})$, which makes them philosophically troubling. Besides, learning inducing points via the optimization of (8) may produce poor predictions [17]. These issues will be addressed by the posterior approximations reviewed below.

2) Posterior approximations: Different from prior approximations, posterior approximations [1], [17] retain exact prior but perform approximate inference. **The most well-known posterior approximation is the elegant variational free energy (VFE)** [17] proposed by Titsias in 2009 by using variational inference (VI) [73]. Instead of modifying the prior $p(\mathbf{f}, \mathbf{f}_*)$, VFE directly approximates the posterior $p(\mathbf{f}, \mathbf{f}_m|\mathbf{y})$, the learning of which is a central task in statistical models, by introducing a variational distribution $q(\mathbf{f}, \mathbf{f}_m|\mathbf{y})$. Then, we have their KL divergence $\text{KL}(q(\mathbf{f}, \mathbf{f}_m|\mathbf{y})||p(\mathbf{f}, \mathbf{f}_m|\mathbf{y}))$

$$:= \log p(\mathbf{y}) - \left\langle \log \frac{p(\mathbf{y}, \mathbf{f}, \mathbf{f}_m)}{q(\mathbf{f}, \mathbf{f}_m|\mathbf{y})} \right\rangle_{q(\mathbf{f}, \mathbf{f}_m|\mathbf{y})} = \log p(\mathbf{y}) - F_q, \quad (14)$$

where $\langle \cdot \rangle_{q(\cdot)}$ represents the expectation over the distribution $q(\cdot)$.⁸ It is found that **minimizing the rigorously defined $\text{KL}(q||p) \geq 0$ is equivalent to maximizing F_q** , since $\log p(\mathbf{y})$ is constant for $q(\mathbf{f}, \mathbf{f}_m|\mathbf{y})$. Thus, **F_q is called evidence lower bound (ELBO) or variational free energy**, which permits us to jointly optimize the variational parameters⁹ and hyperparameters. It is observed that maximizing F_q w.r.t. the hyperparameters directly improves F_q ; while maximizing F_q w.r.t. the variational parameters implicitly drives the approximation to match both the posterior $p(\mathbf{f}, \mathbf{f}_m|\mathbf{y})$ and the evidence $p(\mathbf{y})$.

To derive a tighter bound, the calculus of variations finds the optimal variational distribution $q^*(\mathbf{f}_m|\mathbf{y})$ to remove the dependency of F_q on $q(\mathbf{f}_m|\mathbf{y})$ by taking the relevant derivative to zero, leading to the “collapsed” bound

$$F_{\text{VFE}} = \log q_{\text{DTC}}(\mathbf{y}) - \frac{1}{2\sigma_\epsilon^2} \text{tr}[\mathbf{V}_{nn}] \geq F_q. \quad (15)$$

Note that F_{VFE} differs with $\log q_{\text{DTC}}$ only by a trace term, which however substantially improves the inference quality. In order to maximize F_{VFE} , we should decrease the trace $\text{tr}[\mathbf{V}_{nn}] \geq 0$, which represents the total variance of predicting the latent variables \mathbf{f} given \mathbf{f}_m . Particularly, $\text{tr}[\mathbf{V}_{nn}] = 0$ means $\mathbf{f}_m = \mathbf{f}$ and we recover the full GP. Hence, the trace term (i) is a regularizer that guards against over-fitting; (ii) seeks to deliver a good inducing set; and (iii) always improves F_q with increasing m , see the theoretical analysis in [69], [74]. The third property implies that given enough resources the VFE will recover the full GP, see Fig. 4. In contrast, without this trace term, the DTC often risks over-fitting [75].

Regarding the improvements of VFE, it was extended to continuous and discrete inputs through an efficient QR factorization-based optimization over both inducing points and hyperparameters [76]. The estimation of inducing points

⁸Matthews et al. [74] further extended the procedure to infinite index sets using the KL divergence between stochastic processes such that the posterior is approximated over the entire process \mathbf{f} .

⁹Notably, the inducing positions are regarded as the variational parameters in $q(\mathbf{f}_m|\mathbf{y})$ rather than the model parameters.

⁷The predictive distributions of FITC and FIC only differ when predicting multiple test points simultaneously.

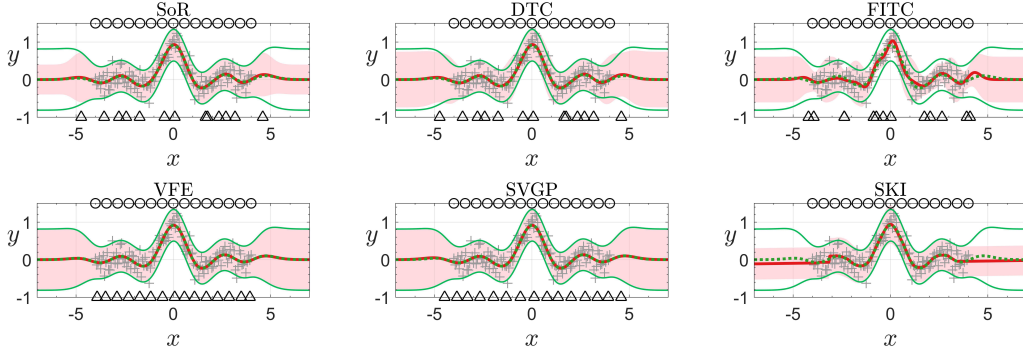


Fig. 4. Illustration of sparse approximations on a 1D toy example with $y(x) = \text{sinc}(x) + \epsilon$ where $\epsilon \sim \mathcal{N}(0, 0.04)$. In the panel, the + symbols represent 120 training points; the top circles represent the initial locations of inducing points, whereas the bottom triangles represent the optimized locations of inducing points; the dot green curves represent the prediction mean of full GP; the green curves represent 95% confidence interval of the full GP predictions; the red curves represent the prediction mean of sparse approximations; the shaded regions represent 95% confidence interval of the predictions of sparse approximations. For SKI, it does not optimize over the positions of inducing points. It is found that among the three prior approximations, (i) the SoR suffers from over-confident prediction variance when leaving the training data; (ii) the FITC captures heteroscedasticity in variance; and (iii) all of them are not guaranteed to converge to the full GP, indicated by the overlapped inducing points. **Differently, the VFE and its stochastic variant SVGP approximate the full GP well due to the posterior approximation.** Finally, though greatly reducing the time complexity by structured inducing set, the SKI may produce discontinuous predictions.

has also been improved in an augmented feature space [77], which is similar to the inter-domain strategy [39]. The authors argued that the similarity of inducing points measured in the Euclidean space is inconsistent to that measured by the GP kernel function. Hence, they assigned a mixture prior on \mathbf{X} in the latent feature space, and derived a regularized bound for choosing good inducing points in the kernel space. Besides, Matthews et al. [74], [78] bridged the gap between the variational inducing-points framework and the more general KL divergence between stochastic processes. Using this new interpretation, Bui et al. [79] approximated the general, infinite joint prior $p(\mathbf{f}_m, \mathbf{f}_{\neq m}, \mathbf{y}) = p(\mathbf{f}_m, \mathbf{f}_{\neq m} | \mathbf{y})p(\mathbf{y})$ which comprises two inferential objects of interest: posterior distribution and model evidence. Minimizing their KL divergence thus encourages *direct* approximation to both posterior and evidence. Hence, the FITC and VFE are interpreted jointly as

$$\log q_{\text{PEP}}(\mathbf{y}) = \log q(\mathbf{y}) - \frac{1-\alpha}{2\alpha} \text{tr} \left[\log \left(\mathbf{I}_n + \frac{\alpha}{\sigma_\epsilon^2} \mathbf{V}_{nn} \right) \right], \quad (16)$$

where $\log q(\mathbf{y})$ takes the form (8) with $\tilde{\mathbf{Q}}_{nn} = \alpha \text{diag}[\mathbf{V}_{nn}]$. By varying $\alpha \in (0, 1]$, we recover FITC when $\alpha = 1$ and VFE when $\alpha \rightarrow 0$. Besides, a hybrid approximation using a moderate α , e.g., $\alpha = 0.5$, often produces better predictions.

To further improve the scalability of VFE, Hensman et al. [1] retained the variational distribution $q(\mathbf{f}_m | \mathbf{y}) = \mathcal{N}(\mathbf{f}_m | \mathbf{m}, \mathbf{S})$ in F_q to obtain a relaxed bound

$$F_q = \langle \log p(\mathbf{y} | \mathbf{f}) \rangle_{p(\mathbf{f} | \mathbf{f}_m)q(\mathbf{f}_m | \mathbf{y})} - \text{KL}(q(\mathbf{f}_m | \mathbf{y}) || p(\mathbf{f}_m)). \quad (17)$$

The first term in the right-hand side of F_q is the sum of n terms due to the *iid* observation noises, i.e., $p(\mathbf{y} | \mathbf{f}) = \prod_{i=1}^n p(y_i | f_i)$. Hence, the stochastic gradient descent (SGD) [80], which encourages large-scale learning, could be employed to obtain an unbiased estimation of F_q using a mini-batch $\{\mathbf{X}_b, \mathbf{y}_b\}$ as

$$F_q \approx \frac{n}{|\mathbf{y}_b|} \sum_{y_i \in \mathbf{y}_b} \int q(\mathbf{f}_m | \mathbf{y}) p(f_i | \mathbf{f}_m) \log p(y_i | f_i) df_i d\mathbf{f}_m - \text{KL}(q(\mathbf{f}_m | \mathbf{y}) || p(\mathbf{f}_m)). \quad (18)$$

Due to the difficulty of optimizing variational parameters \mathbf{m} and \mathbf{S} in the Euclidean space, one can employ the Stochastic Variational Inference (SVI) [81] using natural gradients,¹⁰ resulting in a remarkable complexity of $\mathcal{O}(m^3)$ when $|\mathbf{y}_b| = 1$, and more interestingly, the *online* or *anytime* learning fashion.

Therefore, a crucial property of the **stochastic variational GP (SVGP)** is that it trains a sparse GP at *any time* with a small subset of the training data in each iteration [35]. Another interesting property is that taking a unit step in the natural gradient direction equals to performing an update in the Variational Bayes Expectation Maximization (VB-EM) framework [83]. Though showing high scalability and desirable approximation, the SVGP has some drawbacks: (i) the bound F_q is less tight than F_{VFE} because $q(\mathbf{f}_m | \mathbf{y})$ is not optimally eliminated; (ii) it optimizes over $q(\mathbf{f}_m | \mathbf{y})$ with a huge number of variational parameters, thus requiring much time to complete one epoch of training; and (iii) the introduction of SVI brings the empirical requirement of carefully turning the parameters of SGD.

Inspired by the idea of Hensman, Peng et al. [36] derived the similar factorized variational bound for GPs by taking the weight-space augmentation in (10). The weight-space view (i) allows using flexible basis functions to incorporate various low-rank structures; and (ii) provides a composite non-convex bound enabling the speedup using an asynchronous proximal gradient-based algorithm [84]. By deploying the variational model in a distributed machine learning platform PARAM-ETERSERVER [85], the authors have first scaled GP up to *billions* of data points. Similarly, Cheng and Boots [86] also derived a stochastic variational framework from the weight-space view with the difference being that the mean and variance of $p(\mathbf{f} | \mathbf{w})$ respectively use the *decoupled* basis function sets ϕ_a and ϕ_b , leading to more flexible inference. Besides, a recent interesting **work [35] presents a novel unifying, anytime variational framework akin to Hensman's for accommodating existing sparse approximations, e.g., SoR, DTC, FIT(C) and PIT(C), such that they can be trained via the efficient SGD**

¹⁰The superiority of natural gradients over ordinal gradients for regression has been verified in [82].

which achieves asymptotic convergence to the predictive distribution of the chosen sparse model. The key of this work is to conduct a *reverse* variational inference wherein “reverse” means we can find a prior $p(\mathbf{f}_m) = \mathcal{N}(\mathbf{f}_m|\nu, \mathbf{\Lambda})$ (not the conventional GP prior) such that the variational distribution $q^*(\mathbf{f}_m|\mathbf{y}) = p(\mathbf{f}_m|\mathbf{y})$ for FI(T)C and PI(T)C is the maximum of the variational lower bound.¹¹ Finally, the scalability of Hensman’s model can be further reduced to nearly $\mathcal{O}(m)$ by introducing Kronecker structures for inducing points and the variance of $q(\mathbf{f}_m|\mathbf{y})$ [87], [88].

Titsias and Hensman’s models have been further improved by using, e.g., (i) Bayesian treatment of hyperparameters [89]–[91] rather than traditional point estimation which risks overfitting when the number of hyperparameters is small; and (ii) non-Gaussian likelihoods [90], [92], [93].

3) **Structured sparse approximations**: A direct speedup to solve $(\mathbf{K}_{nn}^\epsilon)^{-1}\mathbf{y}$ in standard GP can be achieved through **fast matrix-vector multiplication (MVM)** [94], [95], which iteratively solves the linear system using **conjugate gradients (CG)** with s ($s \ll n$) iterations,¹² resulting in a time complexity of $\mathcal{O}(sn^2)$. It was argued by [14] that the original MVM has some open questions, e.g., the determination of s , the lack of meaningful speedups, and the badly conditioned kernel matrix. Alternatively, the **pre-conditioned CG** (PCG) [96] employs a pre-conditioning matrix through for example the Nyström approximation to improve the conditioning of kernel matrix and accelerate the CG convergence.

More interestingly, when the kernel matrix \mathbf{K}_{nn} itself has some algebraic structure, the MVM provides massive scalability. For example, the **Kronecker methods** [97], [98] exploit the multi-variate grid inputs $\mathbf{x} \in \Omega_1 \times \dots \times \Omega_d$ and the tensor product kernel with the form $k(\mathbf{x}_i, \mathbf{x}_j) = \prod_{t=1}^d k(\mathbf{x}_i^t, \mathbf{x}_j^t)$.¹³ Then, the kernel matrix decomposes to a Kronecker product $\mathbf{K}_{nn} = \mathbf{K}_1 \otimes \dots \otimes \mathbf{K}_d$, which eases the eigen-decomposition with a greatly reduced time complexity of $\mathcal{O}(d\bar{n}^{d+1})$ where $\bar{n} = \sqrt[d]{n}$ for $d > 1$.¹⁴ Another one is the **Toeplitz methods** [99]—complementary to the Kronecker methods—that exploit the kernel matrix **built from regularly spaced one dimensional points**, resulting in the time complexity of $\mathcal{O}(d\bar{n}^d \log \bar{n})$. The severe limitation of the Kronecker and Toeplitz methods is that they require **grid inputs**, preventing them from being applied to the general arbitrary data points.¹⁵

To handle arbitrary data while retaining the efficient Kronecker structure, the structured kernel interpolation (SKI) [18] imposes the grid constraint on the inducing points. Hence, the matrix \mathbf{K}_{mm} admits the Kronecker structure for $d > 1$ and the Toeplitz structure for $d = 1$, whereas the cross kernel

matrix \mathbf{K}_{nm} is approximated for example by a local linear interpolation using adjacent grid inducing points as

$$k(\mathbf{x}_i, \mathbf{u}_j) \approx w_i k(\mathbf{u}_a, \mathbf{u}_j) + (1 - w_i) k(\mathbf{u}_b, \mathbf{u}_j), \quad (19)$$

where \mathbf{u}_a and \mathbf{u}_b are two inducing points most closely bound \mathbf{x}_i , and w_i is the interpolation weight. Inserting the approximation (19) back into \mathbf{Q}_{nn} , we have

$$\mathbf{Q}_{nn} \approx \mathbf{W}_{nm} \mathbf{K}_{mm}^{-1} \mathbf{W}_{nm}^\top, \quad (20)$$

where the weight matrix \mathbf{W} is extremely sparse since it only has two non-zero entries per row for local linear interpolation, leading to an impressive time complexity of $\mathcal{O}(n + d\bar{m}^{d+1})$ with $\bar{m} = \sqrt[d]{m}$ for solving $(\mathbf{K}_{nn}^\epsilon)^{-1}\mathbf{y}$. Also, the sparse \mathbf{W} incurs the prediction mean with *constant-time* complexity $\mathcal{O}(1)$ and the prediction variance with complexity $\mathcal{O}(m)$ after pre-computing. Furthermore, Pleiss et al. [101] derived a *constant-time* prediction variance using Lanczos approximation, which admits s iterations of MVM for calculation.

The original SKI has two main drawbacks. First, the number m of grid inducing points grows *exponentially* with dimensionality d , making it impractical for $d > 5$. To address this issue, one could use dimensionality reduction or manifold learning to map the inducing points into a p -dimensional ($p \ll d$) latent space [102]; or more interestingly, one can use the hierarchical structure of neural networks to extract the latent low-dimensional feature space [34], [103]. Furthermore, continual efforts [88], [104], [105] have been made to directly reduce the time complexity to be *linear* with d by exploiting the row-partitioned Khatri-Rao structure of \mathbf{K}_{nm} , or imposing tensor train decomposition and Kronecker product to the mean and variance of $q(\mathbf{f}_m|\mathbf{y})$ in Hensman’s variational framework. The linear complexity with d permits the use of numerous inducing points, e.g., $m = 10^d$.

Second, the SKI may produce discontinuous predictions due to the local weight interpolation, and provide overconfident prediction variance when leaving the training data due to the restrictive SoR framework, see Fig. 4. To smooth the predictions, Evans and Nair [104] exploited the row-partitioned Khatri-Rao structure of \mathbf{K}_{nm} rather than using local weight interpolation. To have sensible uncertainty, a diagonal correlation akin to that of FITC has been considered [104], [106].

Finally, note that the permit of many inducing points is expected to improve the model capability. But due to the *grid* constraint, the structured sparse approximations (i) use fixed inducing points, (ii) resort to dimensionality reduction for tackling high-dimensional tasks, and (iii) place the vast majority of inducing points on the domain boundary with increasing d , which in turn may degenerate the model capability.

IV. LOCAL APPROXIMATIONS

Inspired by D&C, local approximations use localized experts to improve the scalability of GP. Besides, compared to global approximations, the **local nature enables capturing non-stationary features**. In what follows, we comprehensively review the *naive-local-experts* which directly employs the pure local experts for prediction, and the *mixture-of-experts* and *product-of-experts* which inherit the advantages of naive-local-experts but boost the predictions through model averaging.

¹¹For VFE and its stochastic variant SVGP, normally, we pre-define the prior $p(\mathbf{f}_m) = \mathcal{N}(\mathbf{f}_m|\mathbf{0}, \mathbf{K}_{mm})$, and then find an optimal $q^*(\mathbf{f}_m|\mathbf{y})$ to maximize the variational lower bound.

¹²The solution to $\mathbf{A}\mathbf{x} = \mathbf{b}$ is the unique minimum of the quadratic function $0.5\mathbf{x}^\top \mathbf{A}\mathbf{x} - \mathbf{x}^\top \mathbf{b}$.

¹³Popular kernels for example the SE kernel (1) fulfill the product structure.

¹⁴ \mathbf{K}_i ($1 \leq i \leq d$) is an $\bar{n} \times \bar{n}$ matrix when the number of points along each dimension is the same. Besides, for the detailed introduction of GP with multiplicative kernels, please refer to section 2.2 of [98].

¹⁵This limitation was relaxed in the *partial grid* and variable noise scenario by introducing virtual observations and inputs [98], [100].

A. Naive-local-experts

It is known that a pair of points far away from each other has a low correlation. Hence, localized experts trained on subsets of \mathcal{D} is expected to produce sensible predictions with low computational complexity. Particularly, the simple naive-local-experts (NLE) [107], [108] lets the local expert \mathcal{M}_i completely responsible for the subregion Ω_i defined by \mathbf{X}_i . Mathematically, we predict at $\mathbf{x}_* \in \Omega_i$ as $p(y_*|\mathcal{D}, \mathbf{x}_*) \approx p_i(y_*|\mathcal{D}_i, \mathbf{x}_*)$.

According to the partition of \mathcal{D} , we classify NLE into two main categories: (i) **inductive NLE**, which first partitions the input space and trains all the experts, and then chooses an appropriate one for predicting at \mathbf{x}_* ; and (ii) **transductive NLE**, which particularly chooses a neighborhood subset \mathcal{D}_* around \mathbf{x}_* , and trains the relevant expert \mathcal{M}_* for predicting at \mathbf{x}_* .

Inductive NLE employs a *static* partition of the whole data using clustering techniques, e.g., Voronoi tessellations [107] and trees [109], [110], and trains independent local GP experts, resulting in $\mathcal{O}(nm_0^2)$ where $m_0 = n/M$ is the training size for each expert. The partition and the experts are usually learned separately; or they can be learned jointly with Bayesian treatment [111]. In contrast, transductive NLE, e.g., the nearest-neighbors (NeNe) [112] which could induce a valid stochastic process [108], [113], employs a *dynamic* partition to choose m_0 neighbor points around \mathbf{x}_* , resulting in $\mathcal{O}(n_t m_0^3)$ complexity that relies on the test size n_t . **A key issue in transductive NLE is the definition of the neighborhood set \mathcal{D}_* around \mathbf{x}_* .** The simplest way is using geometric closeness criteria¹⁶ for selection, which however are not optimal without considering the spatial correlation [114]. Hence, some GP-based active learning methods have been employed to sequentially update the neighborhood set [20], [32], [115], [116].

Whilst enjoying the capability of capturing non-stationary features due to the localized structure, the NLE (i) produces discontinuous predictions on the boundaries of subregions and (ii) suffers from poor generalization capability since it misses the long-term spatial correlations, as depicted in Fig. 5. To address the discontinuity issue, the patched GPs [117], [118] impose continuity conditions such that two adjacent local GPs are patched to share the nearly identical predictions on the boundary. But the patched GPs suffer from inconsistent and even negative prediction variance, and are only available in low dimensional space [72], [118]. More popularly, the model averaging strategy, which is accomplished by the mixture/product of local GP experts elaborated below, well smooths the predictions from multiple experts. **To address the generalization issue**, it is possible to (i) share the hyperparameters across experts, like [26]; or (ii) combine local approximations with global approximations, which will be reviewed in section V-B.

B. Mixture-of-experts

The mixture-of-experts (MoE) devotes to combining the *local* and *diverse* experts owning individual hyperparameters for improving the overall accuracy and reliability [21], [22].¹⁷

¹⁶The selected points should be close to \mathbf{x}_* ; meanwhile, they should distribute uniformly to avoid conveying redundant information.

¹⁷This topic has been studied in a broad regime, called *ensemble learning* [119], for aggregating various learning models to boost predictions.

As shown in Fig. 6, MoE generally expresses the combination as a Gaussian mixture model (GMM) [120]

$$p(y|\mathbf{x}) = \sum_{i=1}^M g_i(\mathbf{x}) p_i(y|\mathbf{x}), \quad (21)$$

where $g_i(\mathbf{x})$ is the gating function, which usually takes a parametric form like the softmax or probit function [120], [121], and can be thought as the probability $p(z = i) = \pi_i$ that the expert indicator z is i , i.e., \mathbf{x} is assigned to expert \mathcal{M}_i ; $p_i(y|\mathbf{x})$ comes from \mathcal{M}_i and is responsible for component i of the mixture. In (21), the gates $\{g_i\}_{i=1}^M$ manage the mixture through *probabilistic* partition of the input space for defining the subregions where the individual experts responsible for. The experts can be a variety of machine learning models, e.g., linear model and support vector machines [122], [123].

The training of MoE usually assumes that the data is *iid* such that we maximize the factorized log likelihood $\sum_{t=1}^n \log p(y_t|\mathbf{x}_t)$ to learn the gating functions and the experts simultaneously by the gradient-based optimizers [124] and more popularly, the EM algorithm [120], [122], [125], [126]. The joint learning permits (i) probabilistic (soft) partition of the input space via both the data and the experts themselves, and (ii) diverse experts specified for different but *overlapped* subregions. Finally, the predictive distribution at \mathbf{x}_* is

$$p(y_*|\mathcal{D}, \mathbf{x}_*) = \sum_{i=1}^M g_i(\mathbf{x}_*|\mathcal{D}) p_i(y_*|\mathcal{D}, \mathbf{x}_*), \quad (22)$$

where $g_i(\mathbf{x}_*|\mathcal{D})$ can be regarded as the posterior probability $p(z_* = i|\mathcal{D})$, called responsibility.

To advance the MoE, (i) the single-layer model in Fig. 6 is extended to a tree-structured hierarchical architecture [125]; (ii) the Bayesian approach is employed instead of the maximum likelihood to get rid of over-fitting and noise-level underestimate [127]; (iii) the t -distribution is considered to handle the outliers [128]; and finally (iv) instead of following the conditional mixture (21), the input distribution $p(\mathbf{x})$ is considered to form the joint likelihood $p(y, \mathbf{x})$ for better assignment of experts [122].

Next, we review the mixture of GP experts for big data. It is observed that (i) the original MoE is designed for capturing multi-modal (non-stationary) features, and the individual *global* experts are responsible for all the data points, leading to high complexity; (ii) the *iid* data assumption does not hold for GP experts which model the data dependencies through joint distribution; and (iii) the parametric gating function g_i is not favored in the Bayesian non-parametric framework. In 2001, Tresp [129] first introduced the mixture of GP experts, which employs $3M$ GP experts to respectively capture the mean, the noise variance, and the gate parameters with nearly $\mathcal{O}(3Mn^3)$ complexity, which is unattractive for big data.

The mixture of GP experts for big data should address two issues: (i) how to reduce the computational complexity (model complexity), and (ii) how to determine the number of experts (model selection).

To address the model complexity issue, there are three core threads. The first is the localization of experts. For instance, the

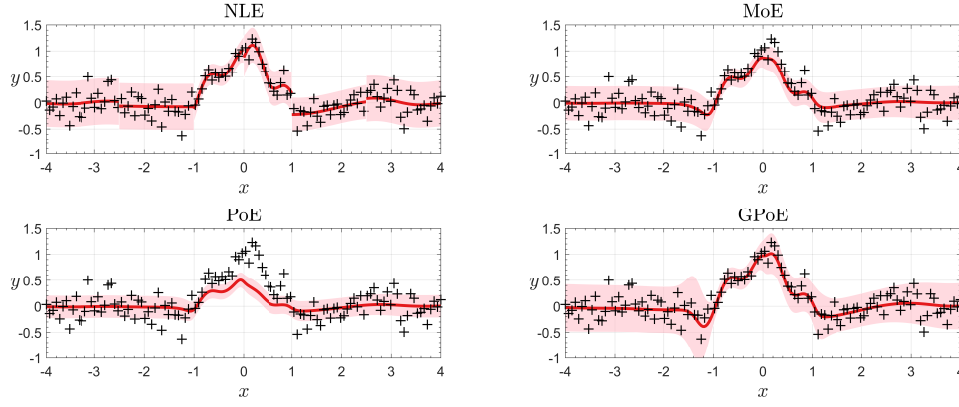


Fig. 5. Illustration of local approximations using six individual experts on the toy example. Note that for MoE, we did not jointly learn the experts and gating functions. For simplicity, we use the individual experts and the differential entropy as β_i in the softmax gating function. It is found that the NLE suffers from discontinuity and poor generalization. The PoE produces poor prediction mean and overconfident prediction variance due to the inability of suppressing poor experts. To alleviate this issue, we could either use gating functions, like the MoE, to provide desirable predictions; or use input-dependent weights, like GPoE, to boost the predictions.

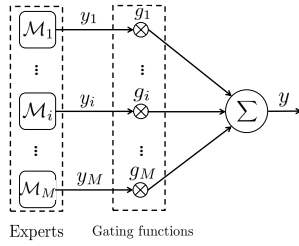


Fig. 6. Illustration of mixture-of-experts.

infinite mixture of GP experts (iMGPE) [23] uses a localized likelihood to get rid of the *iid* assumption as

$$p(\mathbf{y}|\mathbf{X}) = \sum_{\mathbf{z} \in \mathcal{Z}} p(\mathbf{z}|\mathbf{X}) \prod_i p(\mathbf{y}_i|\mathbf{z}, \mathbf{X}_i). \quad (23)$$

Given an instance of the expert indicators $\mathbf{z} = [z_1, \dots, z_n]^T$, the likelihood factorizes over local experts, resulting in $\mathcal{O}(nm_0^2)$ when each expert has the same training size m_0 . Similar to [122], the iMGPE model was further improved by employing the joint distribution $p(\mathbf{y}, \mathbf{X})$ rather than the conditional $p(\mathbf{y}|\mathbf{X})$ as [130]

$$p(\mathbf{y}, \mathbf{X}) = \sum_{\mathbf{z} \in \mathcal{Z}} p(\mathbf{z}) \prod_i p(\mathbf{y}_i|\mathbf{z}, \mathbf{X}_i)p(\mathbf{X}_i|\mathbf{z}). \quad (24)$$

The fully generative model is capable of handling partially specified data and providing inverse functional mappings. But the inference over (23) and (24) should resort to the expensive Markov Chain Monte Carlo (MCMC) sampling. Alternatively, the localization can be achieved by the *hard-cut* EM algorithm using a truncation representation, wherein the E-step assigns the data to experts through maximum a posteriori (MAP) of the expert indicators \mathbf{z} or a threshold value [42], [131]–[133]. Thereafter, the M-step only operates on small subsets.

The second thread is combining global experts with the sparse approximations reviewed in section III-C under the variational EM framework. The dependency among outputs is broken to make variational inference feasible by (i) interpreting GP as the finite Bayesian linear model in (10) [24], [134],

or (ii) using the FITC experts that factorize over \mathbf{f} given the inducing set \mathbf{f}_m [42], [133]. With m inducing points for each expert, the complexity is $\mathcal{O}(nm^2M)$, which can be further reduced to $\mathcal{O}(nm^2)$ with the hard-cut EM [42], [133].

Note that the first two threads assign the data dynamically according to the data property and the experts' performance. Hence, they are denoted as **mixture of implicitly localized experts (MILE)** [22]. The implicit partition determines the optimal allocation of experts, thus enabling capturing the interaction among experts. This advantage encourages the application on data association [135], [136]. The main drawback of MILE however is that in the competitive learning process, some experts may be eliminated due to the zero-coefficient problem caused by unreasonable initial parameters [137].

To relieve the problem of MILE, the third thread is to pre-partition the input space by clustering techniques and assign points to the experts before model training [138], [139]. The **mixture of explicitly localized experts (MELE)** [22] (i) reduces the model complexity as well, and (ii) explicitly determines the architecture of MoE and poses distinct local experts. In the meantime, the drawback of MELE is that the clustering-based partition misses the information from data labels and experts such that it cannot capture the interaction among experts.

Finally, to address the model selection issue, the Akaike information criterion [140] and the synchronously balancing criterion [141] have been employed to choose over a set of candidate M values. More elegantly, the input-dependent Dirichlet process (DP) [23], the Polya urn distribution [130] or the more general Pitman-Yor process [142] is introduced over the expert indicators \mathbf{z} to act as gating functions, which in turn automatically infer the number of experts from data. The complex prior and the infinite M however raise the difficulty in inference [24]. Therefore, for simplicity, a stick-breaking representation of DP is usually used [24], [133].

C. Product-of-experts

Different from the MoE which employs a weighted sum of several probability distributions (experts) via an “*or*” opera-

tion, the product-of-experts (PoE) [25] multiplies these probability distributions, which sidesteps the weight assignment in MoE and is similar to an “and” operation, as

$$p(y|x) = \frac{1}{Z} \prod_{i=1}^M p_i(y|x), \quad (25)$$

where Z is a normalizer, which however makes the inference intractable when maximizing the likelihood $\sum_{i=1}^n \log p(y_i|x_i)$ [143].

Fortunately, the GP experts sidestep this issue since $p_i(y|x)$ in (25) is a Gaussian distribution. Hence, the product of multiple Gaussians is still a Gaussian distribution, resulting in a factorized marginal likelihood of PoE over GP experts

$$p(y|X) = \prod_{i=1}^M p(y_i|X_i), \quad (26)$$

where $p_i(y_i|X_i) \sim \mathcal{N}(y_i|0, K_i + \sigma_{\epsilon,i}^2 I_{n_i})$ with $K_i = k(X_i, X_i) \in R^{n_i \times n_i}$ and n_i being the training size of expert \mathcal{M}_i . This factorization degenerates the full kernel matrix K_{nn} into a diagonal block matrix $\text{diag}[K_1, \dots, K_M]$, leading to $K_{nn}^{-1} \approx \text{diag}[K_1^{-1}, \dots, K_M^{-1}]$. Hence, the complexity is substantially reduced to $\mathcal{O}(nm_0^2)$ given $n_i = m_0$.

It is observed that the PoE likelihood (26) is a special case of the MoE likelihood (23): the MoE likelihood averages the PoE likelihood over possible configurations of the expert indicators z . Consequently, the joint learning of gating functions and experts makes MoE achieve optimal allocation of experts such that it may outperform PoE [144]. Generally, due to the weighted sum form (21), the MoE will never be sharper than the sharpest expert; on the contrary, due to the product form (25), the PoE can be sharper than any of the experts. This can be confirmed in Fig. 5: the PoE produces poor prediction mean and overconfident prediction variance by aggregating the predictions from six independent experts, due to the inability of suppressing poor experts; on the contrary, the MoE provides desirable predictions through gating functions.

Hence, in order to improve PoE, we retain the effective training process but modify the predicting process. Instead of following the simple product rule to aggregate the experts' predictions, various aggregation criteria have been proposed to weaken the votes of poor experts.¹⁸ Particularly, the aggregations are expected to have several properties [144]: (i) the aggregated prediction is sensible in terms of probability, and (ii) the aggregated prediction is robust to weak experts.

Given the GP experts $\{\mathcal{M}_i\}_{i=1}^M$ with predictive distributions $\{p(y_*|\mathcal{D}_i, \mathbf{x}_*) = \mathcal{N}(\mu_i(\mathbf{x}_*), \sigma_i^2(\mathbf{x}_*))\}_{i=1}^M$ at \mathbf{x}_* , the PoEs [25], [144]–[146] aggregate the experts' predictions through a modified product rule as

$$p(y_*|\mathcal{D}, \mathbf{x}_*) = \prod_{i=1}^M p_i^{\beta_{*i}}(y_*|\mathcal{D}_i, \mathbf{x}_*), \quad (27)$$

where β_{*i} is a weight quantifying the contribution of $p_i(y_*|\mathcal{D}_i, \mathbf{x}_*)$ at \mathbf{x}_* . Using (27), we can derive the aggregated prediction mean and variance with closed-form expressions.

The original product-rule aggregation [25] employs a constant weight $\beta_{*i} = 1$, resulting in the aggregated precision $\sigma^{-2}(\mathbf{x}_*) = \sum_{i=1}^M \sigma_i^{-2}(\mathbf{x}_*)$ which will explode rapidly with increasing M . To alleviate the overconfident uncertainty, the generalized PoE (GPoE) [144] introduces a varying weight β_{*i} , which is defined as the difference in the differential entropy between the expert's prior and posterior, to increase or decrease the importance of experts based on their prediction uncertainty. But with this flexible weight, the GPoE produces explosive prediction variance when leaving the training data [28]. To address this issue, we can impose a constraint $\sum_{i=1}^M \beta_{*i} = 1$, see the favorable predictions in Fig. 5; or we can employ a simple weight $\beta_{*i} = 1/M$ such that the GPoE recovers the GP prior when leaving \mathbf{X} , at the cost of however producing underconfident prediction variance [26].¹⁹

Alternatively, Bayesian committee machine (BCM) [26], [147]–[149] aggregates the experts' predictions from another point of view by imposing a conditional independence assumption $p(y|y_*) \approx \prod_{i=1}^M p(y_i|y_*)$, which in turn explicitly introduces a common prior $p(y_*|\theta)$ for experts.²⁰ Thereafter, by using the Bayes rule we have

$$p(y_*|\mathcal{D}, \mathbf{x}_*, \theta) = \frac{\prod_{i=1}^M p_i^{\beta_{*i}}(y_*|\mathcal{D}_i, \mathbf{x}_*, \theta)}{p^{\sum_{i=1}^M \beta_{*i}-1}(y_*|\theta)}. \quad (28)$$

The prior correlation term helps BCM recover the GP prior when leaving \mathbf{X} , and the varying β_{*i} akin to that of GPoE helps produce robust BCM (RBCM) predictions within \mathbf{X} [26]. The BCM however will produce unreliable prediction mean when leaving \mathbf{X} , which has been observed and analyzed in [26], [28]. Notably, unlike PoE, the common prior in BCM requires that all the experts should share the hyperparameters: that is why we explicitly write (28) conditioned on θ .

It has been pointed out that the conventional PoEs and BCMs are inconsistent [28], [150]. That is, their aggregated predictions cannot recover that of full GP when $n \rightarrow \infty$. To raise consistent aggregation, the nested pointwise aggregation of experts (NPAE) [27] removes the independence assumption by assuming that y_i has not yet been observed such that $\mu_i(\mathbf{x}_*)$ is a random variable. The NPAE provides theoretically consistent predictions at the cost of requiring much higher time complexity due to the inversion of a new $M \times M$ kernel matrix at each test point. To be efficient while retaining consistent predictions, instead of using the fixed GP prior to correct the aggregation like (R)BCM, the generalized RBCM (GRBCM) [28] (i) introduces a global communication expert \mathcal{M}_c to perform correction, i.e., acting as a base expert, and (ii) considers the covariance between global and local experts to improve predictions, leading to the aggregation

$$p(y_*|\mathcal{D}, \mathbf{x}_*) = \frac{\prod_{i=2}^M p_{+i}^{\beta_{*i}}(y_*|\mathcal{D}_{+i}, \mathbf{x}_*),}{p_c^{\sum_{i=2}^M \beta_{*i}-1}(y_*|\mathcal{D}_c, \mathbf{x}_*)}, \quad (29)$$

where $p_c(y_*|\mathcal{D}_c, \mathbf{x}_*)$ is the predictive distribution of \mathcal{M}_c , and $p_{+i}(y_*|\mathcal{D}_{+i}, \mathbf{x}_*)$ is the predictive distribution of an enhanced expert \mathcal{M}_{+i} trained on the augmented dataset $\mathcal{D}_{+i} =$

¹⁹With $\beta_{*i} = 1/M$, the GPoE's prediction mean is the same as that of PoE, but the prediction variance blows up as M times that of PoE.

²⁰The BCM can be interpreted as a sparse GP which treats the test inputs as inducing points [16].

¹⁸Note that the aggregation strategy is *post-processing* or *transductive* since it is independent from model training but depends on the test point location.

$\{\mathcal{D}_i, \mathcal{D}_c\}$. Different from (R)BCM, GRBCM employs the informative $p_c(y_*|\mathcal{D}_c, \mathbf{x}_*)$ rather than the simple prior to support consistent predictions when $n \rightarrow \infty$.

Note that the implementations of these transductive aggregations usually share the hyperparameters across experts [26], i.e., $\theta_i = \theta$, because (i) it achieves automatic regularization for guarding against local over-fitting, and eases the inference due to fewer hyperparameters; (ii) it allows to temporarily ignore the noise term of GP in aggregation, i.e., using $p_i(f_*|\mathcal{D}_i, \mathbf{x}_*)$ instead of $p_i(y_*|\mathcal{D}_i, \mathbf{x}_*)$ like [26], to relieve the inconsistency of typical aggregations; and finally (iii) the BCMs cannot support experts using individual hyperparameters as discussed before. But the shared hyperparameters limit the capability of capturing non-stationary features, which is the superiority of local approximations.²¹ Besides, another main drawback of aggregations is the *Kolmogorov inconsistency* [151] induced by the separation of training and predicting such that it is not a unifying probabilistic framework. That is, when we extend the predictive distributions at multiple test points, e.g., \mathbf{x}_* and \mathbf{x}'_* , we have $p(y_*|\mathcal{D}) \neq \int p(y_*, y'_*|\mathcal{D})dy'_*$.

V. IMPROVEMENTS OVER SCALABLE GPs

A. Scalability

The reviewed global approximations, especially the sparse approximations, have generally reduced the standard cubic complexity to $\mathcal{O}(nm^2)$ through m inducing points. Moreover, their complexity can be further reduced through SVI [1] (with $\mathcal{O}(m^3)$) and the exploitation of structured data [18] (with $\mathcal{O}(n+d \log m^{1+1/d})$). Sparse approximations however are still computationally impractical in the scenarios requiring real-time predictions, for example, environmental sensing and monitoring [152]. Alternatively, we can implement sparse approximations using advanced computing infrastructure, e.g., Graphics Processing Units (GPU) and distributed clusters/processors, to further speed up the computation.

Actually, the exact GP using GPU and distributed clusters has been investigated [153]–[157] in the regime of distributed learning [158]. The direct strategy implements parallel and fast linear algebra algorithms, e.g., the HODLR algorithm [155] and the MVM algorithm, with modern distributed memory and multi-core/multi-GPU hardware.²²

In the meantime, the GPU accelerated sparse GPs have been explored. Since most of the terms in (15) can be factorized over data points, the inference can be parallelized and accelerated by GPU [29], [30]. Moreover, by further using the relaxed variational lower bound (17) or grid inducing points in (20), the TensorFlow-based *GPflow* library [31] and the PyTorch-based *GPpyTorch* library [159] have been developed to exploit the usage of GPU hardware.

Besides, the parallel sparse GPs, e.g., the parallel PIT(C) and Incomplete Cholesky Factorization (ICF), have been developed using the message passing interface framework to distribute computations over multiple machines [160], [161].

²¹When sharing hyperparameters, the local structure itself may have good estimations of hyperparameter to capture some kind of local patterns [38].

²²Wang et al. [157] successfully trained a MVM-based exact GP over a million data points in three days through eight GPUs.

Ideally, the parallelization can achieve a speed-up factor close to the number of machines in comparison to the centralized counterparts. Recently, a unifying framework which distributes conventional sparse GPs, including DTC, FI(T)C, PI(T)C and low-rank-cum-markov approximation (LMA) [162], have been built via varying correlated noise structure [43]. Impressively, Peng et al. [36] first implemented the sparse GPs in a distributed computing platform using up to one *billion* training points, and trained the model successfully within two hours.

The local approximations generally have the same complexity to the global approximations if the training size m_0 for each expert is equal to the inducing size m . The local opinion however naturally encourages the parallel/distributed implementations to further reduce computational complexity, see for example [20], [32], [33].

B. Capability

Originated from the low-rank Nyström approximation, the global sparse approximations have been found to work well for approximating slow-varying features with high spatial correlations. This is because in this case, the spectral expansion of the kernel matrix \mathbf{K}_{nn} is dominated by a few large eigenvectors. On the contrary, when the latent function f has quick-varying (non-stationary) features, e.g., the complicated time series tasks [163], the limited global inducing set struggles to exploit the local patterns. The D&C inspired local approximations however are capable of capturing local patterns but suffer from the inability of describing global patterns. Hence, in order to enhance the representational capability of scalable GPs, the hybrid approximations are a straightforward thread by combining global and local approximations in tandem.

Alternatively, the hybrid approximations can be accomplished through an additive process [41], [164], [165]. For instance, after partitioning the input space into subregions, the partially independent conditional (PIC) [41] and its stochastic and distributed variants [35], [43], [91] extend PITC by retaining the conditional independence of training and test, i.e., $\mathbf{f} \perp \mathbf{f}_*|\mathbf{f}_m$, for all the subregions except the one containing the test point \mathbf{x}_* , thus enabling the integration of local and global approximations in a transductive pattern. Mathematically, suppose that $\mathbf{x}_* \in \Omega_j$ we have

$$q(\mathbf{f}, \mathbf{f}_*|\mathbf{f}_m) = p(\mathbf{f}_j, \mathbf{f}_*|\mathbf{f}_m) \prod_{i \neq j}^M p(\mathbf{f}_i|\mathbf{f}_m). \quad (30)$$

This model corresponds to an exact GP with an additive kernel

$$k_{\text{PIC}}(\mathbf{x}_i, \mathbf{x}_j) = k_{\text{SoR}}(\mathbf{x}_i, \mathbf{x}_j) + \psi_{ij}[k(\mathbf{x}_i, \mathbf{x}_j) - k_{\text{SoR}}(\mathbf{x}_i, \mathbf{x}_j)],$$

where $\psi_{ij} = 1$ when \mathbf{x}_i and \mathbf{x}_j belong to the same block; otherwise $\psi_{ij} = 0$. Note that the hybrid PIC recovers FIC by taking all the subregion sizes to one; it is left with the purely local GPs by taking the inducing size to zero. The additive kernel similar to k_{PIC} has also been employed in [164]–[166] by combining the CS kernel [15], [52] and the sparse approximation. Furthermore, as an extension of PIC, the tree-structured GP [167] ignores most of the inter-subregion dependencies of inducing points, but concentrates on the dependency of adjacent subregions lying on a chained

tree structure. The almost purely localized model reduces the time complexity to be linear to n and allows using many inducing points.

Hybrid approximations can also be conducted through a coarse-to-fine process, resulting in a hierarchical structure with multiple layers yielding multi-resolution [40], [168]–[170]. For example, Lee et al. [40] extended the work of [167] into a hierarchically-partitioned GP approximation. This model has multiple layers, with the root layer being localized GPs. Particularly, each layer owns the individual kernel, the configuration of which is determined by the density of inducing points; the adjacent layers share a cross-covariance function which is convolved from two relevant kernels, like [171]. Similarly, Park et al. [168] presented a two-layer model, wherein a GP is placed over the centroids of the subsets as $g(\mathbf{c}) \sim \mathcal{GP}(0, k_g(\mathbf{c}, \mathbf{c}'))$ to construct a rough global approximation in the top layer; then in each subregion of the root layer, a local GP is trained by using the global-level GP as the mean prior $f_i(\mathbf{x}) \sim \mathcal{GP}(g(\mathbf{x}), k_i(\mathbf{x}, \mathbf{x}'))$. This model has also been improved into multi-layer structure [169].

Inevitably, the combination of local approximations may induce discontinuous predictions and inaccurate uncertainties on the boundaries of subregions. For example, the tree-structured GPs [40], [167] completely adopt a localized predictive distribution, which suffers from severe discontinuity. The predictions could be smoothed by placing inducing points on the boundaries of subregions [40], which however is hard to implement. The PIC predictive distribution is composed of both global and local terms [41], which partially alleviates the discontinuity. To completely address the discontinuity, Nguyen et al. [42], [133] combined sparse approximations with model averaging strategies, e.g., MoE.

Finally, despite the hybrid approximations, the representational capability of sparse approximations can be enhanced through a more powerful probabilistic framework. For instance, the inter-domain GP [39], which employs an idea similar to the convolution process in multi-output GP [10], [171] and high-dimensional GP [172], uses a linear integral transform $g(\mathbf{z}) = \int w(\mathbf{x}, \mathbf{z})f(\mathbf{x})d\mathbf{x}$ to map the inducing points into another domain of possibly different dimensions.²³ The inducing variables in the new domain can own a new kernel and induce richer dependencies in the old domain. The inter-domain idea has also been applied to the posterior approximations [74], [79], [173]. Besides, from the weight-space view in (10), it is encouraged to employ different configurations for the basis functions to capture slow- and quick-varying features using different scales [174], [175]. This kind of weight-space non-stationary GP indeed can be derived from the inter-domain view, see [39].

Alternatively, unlike the standard GP using a homoscedastic noise $\epsilon \sim \mathcal{N}(0, \sigma_\epsilon^2)$, the FITC has been extended by a varying noise as $p(\epsilon) = \mathcal{N}(\epsilon | \mathbf{0}, \text{diag}[\mathbf{h}])$ where $\mathbf{h} = [\sigma_\epsilon^2(\mathbf{x}_1), \dots, \sigma_\epsilon^2(\mathbf{x}_n)]^T$ [176]. Moreover, Hoang et al. [43] employed a B -th order Markov property on the correlated noise process $p(\epsilon) = \mathcal{N}(\epsilon | \mathbf{0}, \mathbf{K}_\epsilon)$ in a distributed variational

framework. The unifying framework accommodates existing sparse approximations, e.g., DTC and PIC, by varying the Markov order and noise structure.²⁴ Yu et al. [91] further extended this work through Bayesian treatment of hyperparameters to guard against over-fitting. More elegantly, Almosallam et al. [177] derived a scalable heteroscedastic Bayesian model from the weight-space view by adopting an additional log GP, which is analogous to [178], [179], to account for noise variance as $\sigma_\epsilon^2(\mathbf{x}_i) = \exp(\phi(\mathbf{x}_i)\mathbf{w} + b)$. Differently, Liu et al. [180] derived the stochastic and distributed variants of [178] for scalable heteroscedastic regression. They found that the distributed variant using experts with hybrid parameters improves both scalability and capability, while the stochastic variant using global inducing set may sacrifice the prediction mean for describing the heteroscedastic noise.

VI. EXTENSIONS AND OPEN ISSUES

A. Scalable manifold GP

In scalable GP literature, we usually focus on the scenario wherein the training size n is large (e.g., $n \geq 10^4$) whereas the number d of inputs is modest (e.g., up to hundreds of dimensions). However, in practice we may need to handle the task with comparable n and d or even $d \gg n$, leading to the demand of high-dimensional scalable GP. In practice, we often impose low-dimensional constraints to restrict the high-dimensional problems, i.e., the inputs often lie in a p -dimensional ($p < d$) manifold embedded in the original d -dimensional space. This is because high-dimensional statistical inference is solvable only when the input size d is compatible with the statistical power based on the training size n [181].

Hence, various manifold GPs [11], [176], which are expressed as

$$y = f(\Upsilon \mathbf{x}) + \epsilon, \quad (31)$$

where $\Upsilon \in R^{p \times d}$ is a mapping matrix, have been developed to tackle high-dimensional big data through linear/nonlinear dimensionality reduction [176], [182], [183] or neural network-like input transformation [184]. As a result, the kernel operates the data in a lower dimensional space as $k(\Upsilon \mathbf{x}, \Upsilon \mathbf{x}')$. Note that the mapping matrix and the scalable regression are learned jointly in the Bayesian framework for producing favorable results. Particularly, the true dimensionality of the manifold can be estimated using Bayesian mixture models [185], which however induce a heavy computational budget.

A recent exciting theoretical finding [181] turns out that the learning of the intrinsic manifold can be bypassed, since the GP learned in the original high-dimensional space can achieve the optimal rate when f is not highly smooth. This motivates the use of Bayesian model averaging based on random compression over various configurations in order to reduce computational demands [186].

Continual theoretical and empirical efforts are required for designing specific components, e.g., the convolutional kernel [172], for scalable manifold GPs, because of the urgent demands in various fields, e.g., computer vision (CV).

²³ $g(\mathbf{z})$ is still a GP by a linear transform of $f(\mathbf{x})$. Besides, with $w(\mathbf{x}, \mathbf{z}) = \delta(\mathbf{x} - \mathbf{z})$ where δ is a Dirac delta, the inter-domain GP recovers FITC.

²⁴It achieves the state-of-the-art results on the airline dataset with up to two million data points.

B. Scalable deep GP

Motivated by the enormous success of deep learning in various fields, the scalable deep GPs [34], [187] have been investigated in recent years.²⁵ A simple representative is combining the structural NNs and the flexible non-parametric GP together, wherein NNs map the original input space to the feature space for extracting non-stationary/recurrent features, and the last-layer sparse GP conducts standard regression over the latent space [34], [103], [184], [190]. The parameters of NNs and GP are jointly learned by maximizing the marginal likelihood in order to guard against over-fitting. The NNs+GP structure produces sensible uncertainties, and is **found to be robust to adversarial examples in CV tasks** [191]. Particularly, Cremanns and Roos [192] employed the same hybrid structure, but used the NNs to learn input-dependent hyperparameters for the additive kernels. Then, the NeNe algorithm is employed to ease the GP inference. Besides, Iwata and Ghahramani [193] used the outputs of NNs as prior mean for SVGP [1].

More elegantly, inspired by deep learning, the deep GP (DGP) [187] and its variants [194]–[198], which employ the hierarchical and functional composite

$$y(\mathbf{x}) = f_i(f_{i-1}(\cdots f_1(\mathbf{x}))) + \epsilon \quad (32)$$

to stack multiple layers of latent variable model (LVM) [11] for extracting features. **The DGP showcases great flexibility in (un)supervised scenarios**, resulting in however a non-standard GP. The recently developed convolutional kernel [172] opens up the way of DGP for CV tasks [199]. Note **that the inference in DGP is intractable and expensive**, thus efficient training requires a sophisticated approximate inference via inducing points [196], [200], which in turn may limit the capability. Easier inference without loss of prediction accuracy has always been a big challenge for DGP to completely show its potential beyond regression.

C. Scalable multi-task GP

Due to the multi-task problems that have arose in various fields, e.g., environmental sensor networks and structure design, multi-task GP (MTGP) [9], [10], also **known as multi-output GP**, seeks to learn the latent T correlated tasks $\mathbf{f} = [f_1, \cdots, f_T]^T : R^d \mapsto R^T$ simultaneously as

$$\mathbf{f}(\mathbf{x}) \sim \mathcal{GP}(\mathbf{0}, \mathbf{k}_{\text{MTGP}}(\mathbf{x}, \mathbf{x}')), \quad \mathbf{y}(\mathbf{x}) = \mathbf{f}(\mathbf{x}) + \boldsymbol{\epsilon}, \quad (33)$$

where $\boldsymbol{\epsilon} = [\epsilon_1, \cdots, \epsilon_T]^T$ is the individual noises. The crucial in MTGP is the construction of a valid multi-task kernel $\mathbf{k}_{\text{MTGP}}(\mathbf{x}, \mathbf{x}') \in R^{T \times T}$, which can be built through for example linear model of coregionalization [201] and convolution process [171]. Compared to individual modeling of tasks which loses valuable information, the joint learning of tasks enables boosting predictions by exploiting the task correlations and leveraging information across tasks.

Given that each of the T tasks has n training points, MTGP collects the data from all the tasks and fuses them in an entire kernel matrix, **leading to a much higher complexity $\mathcal{O}(T^3 n^3)$** .

²⁵GP has been pointed out as a shallow but infinitely wide neural network with Gaussian weights [188], [189].

Hence, since the inference in most MTGPs follows the standard process, the above reviewed sparse approximations and local approximations have been applied to MTGPs [171], [202]–[204] to improve the scalability.

To date, scalable MTGPs are mainly studied in the scenario where the tasks have well defined labels and share the input space with modest dimensions. Many efforts are required for extending current MTGPs to handle the 4V challenges in the regime of multi-task (multi-output) learning [205].

D. Scalable online GP

Typical it is assumed that the entire data \mathcal{D} is available a priori to conduct the *off-line* training. We however should consider the scenario where the data arrives sequentially, i.e., *online* or *streaming* data, in small unknown batches. For the complicated online regression, the model [65] should (i) have real-time adaptation to the streaming data; and (ii) handle large-scale case since the new data is continuously arriving.

Sparse GPs are extensible for online learning since they employ a small inducing set to summarize the whole training data [206]–[208]. As a result, the arrived new data interacts only with the inducing points to enhance fast online learning. This is reasonable since the updates of $\mu(\mathbf{x}_*)$ and $\sigma^2(\mathbf{x}_*)$ of FITC and PITC only rely on the inducing set and new data [209], [210]. Moreover, the stochastic variants naturally showcase the online structure [211], since the bound in (17) supports mini-batch learning by stochastic optimization.

But there are two issues for scalable online GPs. First, some of them [207], [209], [210] fix the hyperparameters to obtain constant complexity per update. It is argued that empirically the optimization improves the model significantly in the first few iterations [211]. Hence, with the advanced computing power and the demand of accurate predictions, it could update hyperparameters online over a small number of iterations.

Second, the scalable online GPs implicitly assume that the new data and old data are drawn from the same input distribution. This however is not the case in tasks with complex trajectories, e.g., an evolving time-series [212]. To address the *evolving* online learning, Nguyen et al. [213] presented a simple and intuitive idea using local approximations. This method maintains multiple local GPs, and either uses the new data to update the specific GP when they fall into the relevant local region, or uses the new data to train a new local GP when they are far away from the old data, resulting in however information loss from available training data. As the extension of [65], [214], Bui et al. [212] deployed an elegant probabilistic framework to update the posterior distributions and hyperparameters in an online fashion, where the interaction happens between the old and new inducing points. The primary theoretical bounds for this Bayesian online inference model were also provided in [215].

E. Scalable recurrent GP

There exist various tasks, e.g., speech recognition, system identification, energy forecasting and robotics, wherein the datasets are sequential and the ordering matters. Here, we focus on recurrent GP [216], [217] to handle sequential data.

The popular recurrent GP is GP-based nonlinear autoregressive models with exogenous inputs (GP-NARX) [216], which is generally expressed as

$$\mathbf{x}_t = [y_{t-1}, \dots, y_{t-L_y}, u_{t-1}, \dots, u_{t-L_u}], \quad y_t = f(\mathbf{x}_t) + \epsilon_{y_t}, \quad (34)$$

where u_t is the external input, y_t is the output observation at time t , L_y and L_u are the lagged parameters that respectively indicate the numbers of delayed outputs and inputs to form the regression vector \mathbf{x}_t , f is the emission function, and ϵ_{y_t} accounts for observation noise. Note that the observations $y_{t-1}, \dots, y_{t-L_y}$ here are considered to be deterministic. The transformed input \mathbf{x}_t comprising previous observations and external inputs enables using standard scalable GPs, e.g., sparse approximations, to train the GP-NARX. Due to the simplicity and applicability, GP-NARX has been well studied and extended to achieve robust predictions against outliers [218], local modeling by incorporating prior information [219], and higher-order frequency response functions [220]. A main drawback of GP-NARX however is that it cannot account for the observation noise in \mathbf{x}_t , leading to the errors-in-variables problem. To address this issue, we could (i) conduct data-preprocessing to remove the noise from data [221]; (ii) adopt GPs considering input noise [200]; and (iii) employ the more powerful state space models (SSM) [217] introduced below.

The GP-SSM employs a more general recurrent structure as

$$\mathbf{x}_t = g(\mathbf{x}_{t-1}, u_{t-1}) + \epsilon_{x_t}, \quad y_t = f(\mathbf{x}_t) + \epsilon_{y_t}, \quad (35)$$

where \mathbf{x}_t is the state of the system which acts as an internal memory, g is the transition function, f is the emission function, ϵ_{x_t} is the transition noise, and finally ϵ_{y_t} is the emission noise. Note that GP-NARX is a simplified GP-SSM model with observable state. The GP-SSM takes into account the transition noise and brings the flexibility in requiring no lagged parameters. **But this model suffers from intractable inference since we need to marginalize out all the latent variables, thus requiring approximate inference** [217], [222], [223].

Finally, we again see the trend in combining recurrent GPs with neural networks. For instance, the deep recurrent GP [224] attempts to mimic the well-known recurrent neural networks (RNNs), with each layer modeled by a GP. Similar to DGP [187], the inference in deep recurrent GP is intractable and requires sophisticated approximations, like [225]. Hence, to keep the model as simple as possible while retaining the recurrent capability, the long short-term memory (LSTM) model [226] is combined with scalable GPs, resulting in analytical inference and desirable results [190].

F. Scalable GP classification

Different from the regression tasks mainly reviewed by this article with continuous real observations, the classification has discrete class labels. To this end, the binary GP classification (GPC) model [227] with $y \in \{0, 1\}$ is usually formulated as

$$f \sim \mathcal{GP}(0, k), \quad p(y|f) = \text{Bernoulli}(\pi(f)), \quad (36)$$

where $\pi(\cdot) \in [0, 1]$ is an inverse link function²⁶ that squashes f into the class probability space. Differently, the multi-class GPC (MGPC) [228] with $y \in \{1, \dots, C\}$ is

$$f^c \sim \mathcal{GP}(0, k_c), \quad p(y|\mathbf{f}) = \text{Categorical}(\pi(\mathbf{f})), \quad (37)$$

where $\{f^c\}_{c=1}^C$ are independent latent functions²⁷ for C classes, and $\mathbf{f} = [f^1, \dots, f^C]^T : \mathbb{R}^d \mapsto \mathbb{R}^C$.²⁸ Due to the non-Gaussian likelihood, exact inference for GPC however is intractable, thus requiring approximate inference, the key of which approximates the non-Gaussian posterior $p(f|y) \propto p(y|f)p(f)$ with a Gaussian $q(f|y)$ [227].

Motivated by the success of scalable GPR, we could directly treat GPC as a regression task [231]; or solve it as GPR by a transformation that interprets class labels as outputs of a Dirichlet distribution [232]. This sidesteps the non-Gaussian likelihood. A more principled way however is adopting GPCs in (36) and (37), and combining approximate inference, e.g., laplace approximation, EP and VI, with the sparse strategies in section III-C to derive scalable GPCs [90], [233]–[237].²⁹

The main challenges of scalable GPC, especially MGPC, are: (i) the intractable inference and posterior, and (ii) the high training complexity for a large C . For the first issue, the stochastic GPC derives the model evidence expressed as the integration over an one-dimensional Gaussian distribution, which can be adequately calculated using Gaussian-Hermite quadrature [234], [238]. Furthermore, the GPC equipped with the FITC assumption owns a completely analytical model evidence [235], [236]. Particularly, when taking the logit/softmax inverse link function, the Pölya-Gamma data augmentation [239] offers analytical inference and posterior for GPC [237], [240]. For the second issue, since the complexity of MGPC is linear to the number C of classes, alternatively, we may formulate the model evidence as a sum over classes like [230], thus allowing efficient stochastic training.

VII. CONCLUSIONS

Although the GP itself has a long history, the non-parametric flexibility and the high interpretability make it popular yet posing many challenges in the era of big data. In this paper, we have attempted to summarize the state of scalable GPs in order to (i) well understand the state-of-the-art, and (ii) attain insights into new problems and discoveries. The extensive review seeks to uncover the applicability of scalable GPs to real-world large-scale tasks, which in turn present new challenges, models and theory in the GP community.

APPENDIX A LIBRARIES AND DATASETS

Table I summarizes the primary libraries that implement representative scale GPs and are well-known for both academia

²⁶The conventional inverse link functions include the step function, the (multinomial) probit/logit function, and the softmax function.

²⁷In contrast, inspired by the idea of MTGP, the correlations among latent functions have been exploited by [229].

²⁸Similar to GPR, potential classification error can be considered through $f = \hat{f} + \epsilon$. By varying over different noise distributions, e.g., Gaussian and logistic, we however could recover conventional inverse link functions, e.g., probit and logit functions, through the integration of ϵ [230].

²⁹Different from sparse GPR, the inducing points optimized in sparse GPC are usually pushed towards decision boundary [234].

TABLE I
LIST OF PRIMARY LIBRARIES SUPPORTING REPRESENTATIVE SCALABLE GPs.

Package	Language	Global	Local	Others	GPU supported
GPML	Matlab	FITC [67], VFE [17], SPEP [79], SKI [18]	-	-	-
GPY	Python	VFE [17], SPEP [79], SKI [18], SVGP [1]	-	-	✓
GPstuff	Matlab&R	SoR [56], DTC [65], FITC [67], VFE [17], SVGP [1], CS [15]	-	PIC [41], CS+FIC [165]	-
GPflow	Python	VFE [17], SVGP [1]	-	NNs+SVGP [191]	✓
pyMC3	Python	DTC [65], FITC [67], VFE [17]	-	-	-
GPyTorch	Python	SKI [18]	-	DKL (NNs+SKI) [34]	✓
pyGPs	Python	FITC [67]	-	-	-
AugGP	Julia	VFE [17], SVGP [1]	-	-	-
laGP	R	-	NeNe [112]	-	✓
GPLP	Matlab	-	NeNe [112], PoE [145], DDM [241]	PIC [41]	-

TABLE II
BIG REGRESSION DATASETS ($n \geq 10^4$) IN THE LITERATURE.

Dataset	No. of inputs	No. of observations
terrain [40]	2	40.00K
aimpeak [161]	5	41.85K
sarcos [5]	21	48.93K
natural sound [18]	1	59.31K
chem [14]	15	63.07K
kuka [242]	28	197.92K
crime [243]	2	233.09K
sdss [177]	10	300.00K
precipitation [106]	3	628.47K
mujoco [86]	23	936.35K
airline [1]	8	5.93M
bimbo [37]	147	9.05M
fortune [37]	112	10.35M
NYC taxi [36]	9	1.21B

and industry.³⁰ It is observed that Python and hardware acceleration are becoming popular for the GP community. Note that some specific scalable GP packages implementing the advanced models reviewed in sections V and VI are not listed here, which can be found in the relevant researchers' webpage.

Besides, except the well-known UCI³¹ regression datasets with $n \in [15, 4.18 \times 10^6]$ and $d \in [3, 4.8 \times 10^5]$, and the LIBSVM³² regression datasets with $n \in [152, 2.06 \times 10^4]$ and $d \in [6, 4.27 \times 10^6]$, Table II summarizes the regression datasets ($n \geq 10^4$) occurred in scalable GP literature. It is found that researchers have assessed scalable GPs with up to about one billion training points [36].

ACKNOWLEDGEMENTS

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.

REFERENCES

- [1] J. Hensman, N. Fusi, and N. D. Lawrence, "Gaussian processes for big data," in *Uncertainty in Artificial Intelligence*, 2013, pp. 282–290.
- [2] S. Del Río, V. López, J. M. Benítez, and F. Herrera, "On the use of MapReduce for imbalanced big data using Random Forest," *Information Sciences*, vol. 285, pp. 112–137, 2014.
- [3] Y. LeCun, Y. Bengio, and G. Hinton, "Deep learning," *Nature*, vol. 521, no. 7553, p. 436, 2015.
- [4] D. Silver, J. Schrittwieser, K. Simonyan, I. Antonoglou, A. Huang, A. Guez, T. Hubert, L. Baker, M. Lai, A. Bolton *et al.*, "Mastering the game of Go without human knowledge," *Nature*, vol. 550, no. 7676, p. 354, 2017.
- [5] C. E. Rasmussen and C. K. Williams, *Gaussian processes for machine learning*. MIT Press, 2006.
- [6] G. Matheron, "Principles of geostatistics," *Economic Geology*, vol. 58, no. 8, pp. 1246–1266, 1963.
- [7] J. Sacks, W. J. Welch, T. J. Mitchell, and H. P. Wynn, "Design and analysis of computer experiments," *Statistical Science*, vol. 4, no. 4, pp. 409–423, 1989.
- [8] H. Liu, Y.-S. Ong, and J. Cai, "A survey of adaptive sampling for global metamodeling in support of simulation-based complex engineering design," *Structural and Multidisciplinary Optimization*, vol. 57, no. 1, pp. 393–416, 2018.
- [9] M. A. Alvarez, L. Rosasco, N. D. Lawrence *et al.*, "Kernels for vector-valued functions: A review," *Foundations and Trends® in Machine Learning*, vol. 4, no. 3, pp. 195–266, 2012.
- [10] H. Liu, J. Cai, and Y.-S. Ong, "Remarks on multi-output Gaussian process regression," *Knowledge-Based Systems*, vol. 144, no. March, pp. 102–121, 2018.
- [11] N. Lawrence, "Probabilistic non-linear principal component analysis with Gaussian process latent variable models," *Journal of Machine Learning Research*, vol. 6, no. Nov, pp. 1783–1816, 2005.
- [12] B. Shahriari, K. Swersky, Z. Wang, R. P. Adams, and N. De Freitas, "Taking the human out of the loop: A review of Bayesian optimization," *Proceedings of the IEEE*, vol. 104, no. 1, pp. 148–175, 2016.
- [13] S. Yin and O. Kaynak, "Big data for modern industry: challenges and trends," *Proceedings of the IEEE*, vol. 103, no. 2, pp. 143–146, 2015.
- [14] K. Chalupka, C. K. Williams, and I. Murray, "A framework for evaluating approximation methods for Gaussian process regression," *Journal of Machine Learning Research*, vol. 14, pp. 333–350, 2013.
- [15] T. Gneiting, "Compactly supported correlation functions," *Journal of Multivariate Analysis*, vol. 83, no. 2, pp. 493–508, 2002.
- [16] J. Quiñero-Candela and C. E. Rasmussen, "A unifying view of sparse approximate Gaussian process regression," *Journal of Machine Learning Research*, vol. 6, no. Dec, pp. 1939–1959, 2005.
- [17] M. K. Titsias, "Variational learning of inducing variables in sparse Gaussian processes," in *Artificial Intelligence and Statistics*, 2009, pp. 567–574.
- [18] A. Wilson and H. Nickisch, "Kernel interpolation for scalable structured Gaussian processes (KISS-GP)," in *International Conference on Machine Learning*, 2015, pp. 1775–1784.
- [19] R. B. Gramacy and H. K. H. Lee, "Bayesian treed Gaussian process models with an application to computer modeling," *Journal of the American Statistical Association*, vol. 103, no. 483, pp. 1119–1130, 2008.

³⁰The primary libraries include: (1) GPML (<http://www.gaussianprocess.org/gpml/>); (2) GPY (<https://github.com/SheffieldML/GPy>); (3) GPstuff (<https://github.com/gpstuff-dev/gpstuff>); (4) GPflow (<https://github.com/GPflow/GPflow>); (5) pyMC3 (<https://github.com/pymc-devs/pymc3>); (6) GPy-Torch (<https://github.com/cornellius-gp/gpytorch>); (7) pyGPs (<https://github.com/PMBio/pygp>); (8) AugGP (<https://github.com/theogf/AugmentedGaussianProcesses.jl>); (9) laGP (http://bobby.gramacy.com/r_packages/laGP/); (10) GPLP (<http://www.jmlr.org/mloss/>).

³¹<https://archive.ics.uci.edu/ml/index.php>

³²<https://www.csie.ntu.edu.tw/~cjlin/libsvm/>

- [20] R. B. Gramacy *et al.*, “laGP: Large-scale spatial modeling via local approximate Gaussian processes in R,” *Journal of Statistical Software*, vol. 72, no. 1, pp. 1–46, 2016.
- [21] S. E. Yuktel, J. N. Wilson, and P. D. Gader, “Twenty years of mixture of experts,” *IEEE Transactions on Neural Networks and Learning Systems*, vol. 23, no. 8, pp. 1177–1193, 2012.
- [22] S. Masoudnia and R. Ebrahimpour, “Mixture of experts: A literature survey,” *Artificial Intelligence Review*, vol. 42, no. 2, pp. 275–293, 2014.
- [23] C. E. Rasmussen and Z. Ghahramani, “Infinite mixtures of Gaussian process experts,” in *Advances in Neural Information Processing Systems*, 2002, pp. 881–888.
- [24] S. Sun and X. Xu, “Variational inference for infinite mixtures of Gaussian processes with applications to traffic flow prediction,” *IEEE Transactions on Intelligent Transportation Systems*, vol. 12, no. 2, pp. 466–475, 2011.
- [25] G. E. Hinton, “Training products of experts by minimizing contrastive divergence,” *Neural Computation*, vol. 14, no. 8, pp. 1771–1800, 2002.
- [26] M. P. Deisenroth and J. W. Ng, “Distributed Gaussian processes,” in *International Conference on Machine Learning*. JMLR. org, 2015, pp. 1481–1490.
- [27] D. Rulli re, N. Durrande, F. Bachoc, and C. Chevalier, “Nested Kriging predictions for datasets with a large number of observations,” *Statistics and Computing*, vol. 28, no. 4, pp. 849–867, 2018.
- [28] H. Liu, J. Cai, Y. Wang, and Y.-S. Ong, “Generalized robust Bayesian committee machine for large-scale Gaussian process regression,” in *International Conference on Machine Learning*, 2018, pp. 1–10.
- [29] Y. Gal, M. van der Wilk, and 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.
- [30] Z. Dai, A. Damianou, J. Hensman, and N. Lawrence, “Gaussian process models with parallelization and GPU acceleration,” *arXiv preprint arXiv:1410.4984*, 2014.
- [31] A. G. d. G. Matthews, M. van der Wilk, T. Nickson, K. Fujii, A. Boukouvalas, P. Le n-Villagr , Z. Ghahramani, and J. Hensman, “GPflow: A Gaussian process library using TensorFlow,” *Journal of Machine Learning Research*, vol. 18, no. 40, pp. 1–6, 2017.
- [32] R. B. Gramacy and B. Haaland, “Speeding up neighborhood search in local Gaussian process prediction,” *Technometrics*, vol. 58, no. 3, pp. 294–303, 2016.
- [33] R. B. Gramacy, J. Niemi, and R. M. Weiss, “Massively parallel approximate Gaussian process regression,” *SIAM/ASA Journal on Uncertainty Quantification*, vol. 2, no. 1, pp. 564–584, 2014.
- [34] A. G. Wilson, Z. Hu, R. Salakhutdinov, and E. P. Xing, “Deep kernel learning,” in *Artificial Intelligence and Statistics*, 2016, pp. 370–378.
- [35] T. N. Hoang, Q. M. Hoang, and 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.
- [36] H. Peng, S. Zhe, X. Zhang, and Y. Qi, “Asynchronous distributed variational Gaussian process for regression,” in *International Conference on Machine Learning*, 2017, pp. 2788–2797.
- [37] R. Rivera and E. Burnaev, “Forecasting of commercial sales with large scale Gaussian processes,” *arXiv preprint arXiv:1709.05548*, 2017.
- [38] H. Liu, J. Cai, Y.-S. Ong, and Y. Wang, “Understanding and comparing scalable Gaussian process regression for big data,” *Knowledge-Based Systems*, vol. 164, pp. 324–335, 2019.
- [39] M. L zaro-Gredilla and A. Figueiras-Vidal, “Inter-domain Gaussian processes for sparse inference using inducing features,” in *Advances in Neural Information Processing Systems*, 2009, pp. 1087–1095.
- [40] B.-J. Lee, J. Lee, and K.-E. Kim, “Hierarchically-partitioned Gaussian process approximation,” in *Artificial Intelligence and Statistics*, 2017, pp. 822–831.
- [41] E. Snelson and Z. Ghahramani, “Local and global sparse Gaussian process approximations,” in *Artificial Intelligence and Statistics*. PMLR, 2007, pp. 524–531.
- [42] T. Nguyen and E. Bonilla, “Fast allocation of Gaussian process experts,” in *International Conference on Machine Learning*, 2014, pp. 145–153.
- [43] T. N. Hoang, Q. M. Hoang, and B. K. H. Low, “A distributed variational inference framework for unifying parallel sparse Gaussian process regression models,” in *International Conference on Machine Learning*, 2016, pp. 382–391.
- [44] G. Camps-Valls, J. Verrelst, J. Munoz-Mari, V. Laparra, F. Mateo-Jimenez, and J. Gomez-Dans, “A survey on Gaussian processes for earth-observation data analysis: A comprehensive investigation,” *IEEE Geoscience and Remote Sensing Magazine*, vol. 4, no. 2, pp. 58–78, 2016.
- [45] K. Hayashi, M. Imaizumi, and Y. Yoshida, “On random subsampling of Gaussian process regression: A graphon-based analysis,” *arXiv preprint arXiv:1901.09541*, 2019.
- [46] F. P. Preparata and M. I. Shamos, *Computational geometry: An introduction*. Springer Science & Business Media, 2012.
- [47] R. Herbrich, N. D. Lawrence, and M. Seeger, “Fast sparse Gaussian process methods: The informative vector machine,” in *Advances in Neural Information Processing Systems*, 2003, pp. 625–632.
- [48] M. Seeger, “Bayesian Gaussian process models: PAC-Bayesian generalisation error bounds and sparse approximations,” Ph.D. dissertation, University of Edinburgh, 2003.
- [49] S. Keerthi and W. Chu, “A matching pursuit approach to sparse Gaussian process regression,” in *Advances in Neural Information Processing Systems*, 2006, pp. 643–650.
- [50] A. Melkumyan and F. Ramos, “A sparse covariance function for exact Gaussian process inference in large datasets,” in *International Joint Conferences on Artificial Intelligence*, vol. 9, 2009, pp. 1936–1942.
- [51] M. Buhmann, “A new class of radial basis functions with compact support,” *Mathematics of Computation*, vol. 70, no. 233, pp. 307–318, 2001.
- [52] H. Wendland, *Scattered data approximation*. Cambridge university press, 2004.
- [53] A. Gittens and M. W. Mahoney, “Revisiting the nystr m method for improved large-scale machine learning,” *Journal of Machine Learning Research*, vol. 17, no. 1, pp. 3977–4041, 2016.
- [54] C. K. Williams and M. Seeger, “Using the Nystr m method to speed up kernel machines,” in *Advances in Neural Information Processing Systems*, 2001, pp. 682–688.
- [55] C. Williams, C. Rasmussen, A. Scwaighofer, and V. Tresp, “Observations on the nystr m method for Gaussian process prediction,” University of Edinburgh, Tech. Rep., 2002.
- [56] A. J. Smola and P. L. Bartlett, “Sparse greedy Gaussian process regression,” in *Advances in Neural Information Processing Systems*, 2001, pp. 619–625.
- [57] M. E. Tipping, “Sparse Bayesian learning and the relevance vector machine,” *Journal of Machine Learning Research*, vol. 1, no. Jun, pp. 211–244, 2001.
- [58] H. Peng and Y. Qi, “EigenGP: Gaussian process models with adaptive eigenfunctions,” in *International Joint Conferences on Artificial Intelligence*, 2015, pp. 3763–3769.
- [59] N. Cressie and G. Johannesson, “Fixed rank Kriging for very large spatial data sets,” *Journal of the Royal Statistical Society: Series B (Statistical Methodology)*, vol. 70, no. 1, pp. 209–226, 2008.
- [60] C. E. Rasmussen and J. Qui n ro-Candela, “Healing the relevance vector machine through augmentation,” in *International Conference on Machine Learning*, 2005, pp. 689–696.
- [61] M. L zaro-Gredilla and J. Qui n ro-Candela, “Sparse spectrum Gaussian process regression,” *Journal of Machine Learning Research*, vol. 11, pp. 1865–1881, 2010.
- [62] Y. Gal and R. Turner, “Improving the Gaussian process sparse spectrum approximation by representing uncertainty in frequency inputs,” in *International Conference on Machine Learning*, 2015, pp. 655–664.
- [63] L. S. Tan, V. M. Ong, D. J. Nott, and A. Jasra, “Variational inference for sparse spectrum Gaussian process regression,” *Statistics and Computing*, vol. 26, no. 6, pp. 1243–1261, 2016.
- [64] Q. M. Hoang, T. N. Hoang, and K. H. Low, “A generalized stochastic variational bayesian hyperparameter learning framework for sparse spectrum Gaussian process regression,” in *AAAI Conference on Artificial Intelligence*, 2017, pp. 2007–2014.
- [65] L. Cs t  and M. Opper, “Sparse on-line Gaussian processes,” *Neural Computation*, vol. 14, no. 3, pp. 641–668, 2002.
- [66] M. Seeger, C. Williams, and N. Lawrence, “Fast forward selection to speed up sparse Gaussian process regression,” in *Artificial Intelligence and Statistics*. PMLR, 2003, pp. EPFL–CONF–161 318.
- [67] E. Snelson and Z. Ghahramani, “Sparse Gaussian processes using pseudo-inputs,” in *Advances in Neural Information Processing Systems*, 2006, pp. 1257–1264.
- [68] E. L. Snelson, “Flexible and efficient Gaussian process models for machine learning,” Ph.D. dissertation, University College London, 2008.
- [69] M. Bauer, M. van der Wilk, and C. E. Rasmussen, “Understanding probabilistic sparse Gaussian process approximations,” in *Advances in Neural Information Processing Systems*, 2016, pp. 1533–1541.

- [70] L. Csató and M. Opper, "Sparse representation for Gaussian process models," in *Advances in Neural Information Processing Systems*, 2001, pp. 444–450.
- [71] J. Schreiter, D. Nguyen-Tuong, and M. Toussaint, "Efficient sparsification for Gaussian process regression," *Neurocomputing*, vol. 192, pp. 29–37, 2016.
- [72] A. Pourhabib, F. Liang, and Y. Ding, "Bayesian site selection for fast Gaussian process regression," *IEEE Transactions*, vol. 46, no. 5, pp. 543–555, 2014.
- [73] D. M. Blei, A. Kucukelbir, and J. D. McAuliffe, "Variational inference: A review for statisticians," *Journal of the American Statistical Association*, vol. 112, no. 518, pp. 859–877, 2017.
- [74] A. G. d. G. Matthews, J. Hensman, R. Turner, and Z. Ghahramani, "On sparse variational methods and the Kullback-Leibler divergence between stochastic processes," *Journal of Machine Learning Research*, vol. 51, pp. 231–239, 2016.
- [75] M. K. Titsias, "Variational model selection for sparse Gaussian process regression," University of Manchester, Tech. Rep., 2009.
- [76] Y. Cao, M. A. Brubaker, D. J. Fleet, and A. Hertzmann, "Efficient optimization for sparse Gaussian process regression," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 37, no. 12, pp. 2415–2427, 2015.
- [77] S. Zhe, "Regularized variational sparse Gaussian processes," in *NIPS Workshop on Approximate Inference*, 2017, pp. 1–5.
- [78] A. G. de Garis Matthews, "Scalable Gaussian process inference using variational methods," Ph.D. dissertation, University of Cambridge, 2016.
- [79] T. D. Bui, J. Yan, and R. E. Turner, "A unifying framework for Gaussian process pseudo-point approximations using power expectation propagation," *Journal of Machine Learning Research*, vol. 18, no. 1, pp. 3649–3720, 2017.
- [80] D. P. Kingma and J. Ba, "Adam: A method for stochastic optimization," *arXiv preprint arXiv:1412.6980*, 2014.
- [81] M. D. Hoffman, D. M. Blei, C. Wang, and J. Paisley, "Stochastic variational inference," *Journal of Machine Learning Research*, vol. 14, no. 1, pp. 1303–1347, 2013.
- [82] H. Salimbeni, S. Eleftheriadis, and J. Hensman, "Natural gradients in practice: Non-conjugate variational inference in Gaussian process models," in *Artificial Intelligence and Statistics*, 2018, pp. 689–697.
- [83] J. Hensman, M. Rattray, and N. D. Lawrence, "Fast variational inference in the conjugate exponential family," in *Advances in Neural Information Processing Systems*, 2012, pp. 2888–2896.
- [84] M. Li, D. G. Andersen, and A. Smola, "Distributed delayed proximal gradient methods," in *NIPS Workshop on Optimization for Machine Learning*, vol. 3, 2013, pp. 1–5.
- [85] M. Li, D. G. Andersen, A. J. Smola, and K. Yu, "Communication efficient distributed machine learning with the parameter server," in *Advances in Neural Information Processing Systems*, 2014, pp. 19–27.
- [86] C.-A. Cheng and B. Boots, "Variational inference for Gaussian process models with linear complexity," in *Advances in Neural Information Processing Systems*, 2017, pp. 5190–5200.
- [87] T. Nickson, T. Gunter, C. Lloyd, M. A. Osborne, and S. Roberts, "Blitzkriging: Kronecker-structured stochastic Gaussian processes," *arXiv preprint arXiv:1510.07965*, 2015.
- [88] P. Izmailov, A. Novikov, and D. Kropotov, "Scalable Gaussian processes with billions of inducing inputs via tensor train decomposition," *arXiv preprint arXiv:1710.07324*, 2017.
- [89] M. T. R. AUEB and M. Lázaro-Gredilla, "Variational inference for mahalanobis distance metrics in Gaussian process regression," in *Advances in Neural Information Processing Systems*, 2013, pp. 279–287.
- [90] J. Hensman, A. G. Matthews, M. Filippone, and Z. Ghahramani, "MCMC for variationally sparse Gaussian processes," in *Advances in Neural Information Processing Systems*, 2015, pp. 1648–1656.
- [91] H. Yu, T. N. Hoang, K. H. Low, and P. Jaillet, "Stochastic variational inference for fully Bayesian sparse Gaussian process regression models," *arXiv preprint arXiv:1711.00221*, 2017.
- [92] R. Sheth, Y. Wang, and R. Khordon, "Sparse variational inference for generalized GP models," in *International Conference on Machine Learning*, 2015, pp. 1302–1311.
- [93] J. Hensman, N. Durrande, and A. Solin, "Variational fourier features for Gaussian processes," *arXiv preprint arXiv:1611.06740*, 2016.
- [94] Y. Shen, M. Seeger, and A. Y. Ng, "Fast Gaussian process regression using KD-trees," in *Advances in Neural Information Processing Systems*, 2006, pp. 1225–1232.
- [95] V. I. Morariu, B. V. Srinivasan, V. C. Raykar, R. Duraiswami, and L. S. Davis, "Automatic online tuning for fast Gaussian summation," in *Advances in Neural Information Processing Systems*, 2009, pp. 1113–1120.
- [96] K. Cutajar, M. Osborne, J. Cunningham, and M. Filippone, "Preconditioning kernel matrices," in *International Conference on Machine Learning*, 2016, pp. 2529–2538.
- [97] Y. Saatçi, "Scalable inference for structured Gaussian process models," Ph.D. dissertation, University of Cambridge, 2011.
- [98] E. Gilboa, Y. Saatçi, and J. P. Cunningham, "Scaling multidimensional inference for structured Gaussian processes," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 37, no. 2, pp. 424–436, 2015.
- [99] J. P. Cunningham, K. V. Shenoy, and M. Sahani, "Fast Gaussian process methods for point process intensity estimation," in *International Conference on Machine Learning*, 2008, pp. 192–199.
- [100] A. G. Wilson, E. Gilboa, A. Nehorai, and J. P. Cunningham, "Fast kernel learning for multidimensional pattern extrapolation," in *Advances in Neural Information Processing Systems*, 2014, pp. 3626–3634.
- [101] G. Pleiss, J. Gardner, K. Weinberger, and A. G. Wilson, "Constant-time predictive distributions for Gaussian processes," in *International Conference on Machine Learning*, 2018, pp. 4111–4120.
- [102] A. G. Wilson, C. Dann, and H. Nickisch, "Thoughts on massively scalable Gaussian processes," *arXiv preprint arXiv:1511.01870*, 2015.
- [103] A. G. Wilson, Z. Hu, R. R. Salakhutdinov, and E. P. Xing, "Stochastic variational deep kernel learning," in *Advances in Neural Information Processing Systems*, 2016, pp. 2586–2594.
- [104] T. W. Evans and P. B. Nair, "Scalable Gaussian processes with grid-structured eigenfunctions (GP-GRIEF)," in *NIPS Workshop on Advances in Approximate Bayesian Inference*, 2017, pp. 1–5.
- [105] J. R. Gardner, G. Pleiss, R. Wu, K. Q. Weinberger, and A. G. Wilson, "Product kernel interpolation for scalable Gaussian processes," *arXiv preprint arXiv:1802.08903*, 2018.
- [106] K. Dong, D. Eriksson, H. Nickisch, D. Bindel, and A. G. Wilson, "Scalable log determinants for Gaussian process kernel learning," in *Advances in Neural Information Processing Systems*, 2017, pp. 6330–6340.
- [107] H.-M. Kim, B. K. Mallick, and C. Holmes, "Analyzing nonstationary spatial data using piecewise Gaussian processes," *Journal of the American Statistical Association*, vol. 100, no. 470, pp. 653–668, 2005.
- [108] A. Datta, S. Banerjee, A. O. Finley, and A. E. Gelfand, "On nearest-neighbor Gaussian process models for massive spatial data," *Wiley Interdisciplinary Reviews: Computational Statistics*, vol. 8, no. 5, pp. 162–171, 2016.
- [109] S. Vasudevan, F. Ramos, E. Nettleton, and H. Durrant-Whyte, "Gaussian process modeling of large-scale terrain," *Journal of Field Robotics*, vol. 26, no. 10, pp. 812–840, 2009.
- [110] M. T. Pratola, H. A. Chipman, J. R. Gattiker, D. M. Higdon, R. McCulloch, and W. N. Rust, "Parallel Bayesian additive regression trees," *Journal of Computational and Graphical Statistics*, vol. 23, no. 3, pp. 830–852, 2014.
- [111] D. Denison, N. Adams, C. Holmes, and D. Hand, "Bayesian partition modelling," *Computational Statistics & Data Analysis*, vol. 38, no. 4, pp. 475–485, 2002.
- [112] R. Urtasun and T. Darrell, "Sparse probabilistic regression for activity-independent human pose inference," in *IEEE Conference on Computer Vision and Pattern Recognition*, 2008, pp. 1–8.
- [113] A. Datta, S. Banerjee, A. O. Finley, and A. E. Gelfand, "Hierarchical nearest-neighbor Gaussian process models for large geostatistical datasets," *Journal of the American Statistical Association*, vol. 111, no. 514, pp. 800–812, 2016.
- [114] X. Emery, "The Kriging update equations and their application to the selection of neighboring data," *Computational Geosciences*, vol. 13, no. 3, pp. 269–280, 2009.
- [115] R. B. Gramacy and H. K. Lee, "Adaptive design and analysis of supercomputer experiments," *Technometrics*, vol. 51, no. 2, pp. 130–145, 2009.
- [116] R. B. Gramacy and D. W. Apley, "Local Gaussian process approximation for large computer experiments," *Journal of Computational and Graphical Statistics*, vol. 24, no. 2, pp. 561–578, 2015.
- [117] C. Park and J. Z. Huang, "Efficient computation of Gaussian process regression for large spatial data sets by patching local Gaussian processes," *Journal of Machine Learning Research*, vol. 17, no. 174, pp. 1–29, 2016.
- [118] C. Park and D. Apley, "Patchwork Kriging for large-scale Gaussian process regression," *arXiv preprint arXiv:1701.06655*, 2017.
- [119] J. Mendes-Moreira, C. Soares, A. M. Jorge, and J. F. D. Sousa, "Ensemble approaches for regression: A survey," *ACM Computing Surveys*, vol. 45, no. 1, p. 10, 2012.

- [120] R. A. Jacobs, M. I. Jordan, S. J. Nowlan, and G. E. Hinton, "Adaptive mixtures of local experts," *Neural Computation*, vol. 3, no. 1, pp. 79–87, 1991.
- [121] J. Geweke and M. Keane, "Smoothly mixing regressions," *Journal of Econometrics*, vol. 138, no. 1, pp. 252–290, 2007.
- [122] L. Xu, M. I. Jordan, and G. E. Hinton, "An alternative model for mixtures of experts," in *Advances in Neural Information Processing Systems*, 1995, pp. 633–640.
- [123] C. A. Lima, A. L. Coelho, and F. J. Von Zuben, "Hybridizing mixtures of experts with support vector machines: Investigation into nonlinear dynamic systems identification," *Information Sciences*, vol. 177, no. 10, pp. 2049–2074, 2007.
- [124] K. Chen, L. Xu, and H. Chi, "Improved learning algorithms for mixture of experts in multiclass classification," *Neural Networks*, vol. 12, no. 9, pp. 1229–1252, 1999.
- [125] M. I. Jordan and R. A. Jacobs, "Hierarchical mixtures of experts and the EM algorithm," *Neural Computation*, vol. 6, no. 2, pp. 181–214, 1994.
- [126] S.-K. Ng and G. J. McLachlan, "Extension of mixture-of-experts networks for binary classification of hierarchical data," *Artificial Intelligence in Medicine*, vol. 41, no. 1, pp. 57–67, 2007.
- [127] C. M. Bishop and M. Svenskn, "Bayesian hierarchical mixtures of experts," in *Uncertainty in Artificial Intelligence*. Morgan Kaufmann Publishers Inc., 2002, pp. 57–64.
- [128] F. Chamroukhi, "Skew t mixture of experts," *Neurocomputing*, vol. 266, pp. 390–408, 2017.
- [129] V. Tresp, "Mixtures of Gaussian processes," in *Advances in Neural Information Processing Systems*, 2001, pp. 654–660.
- [130] E. Meeds and S. Osindero, "An alternative infinite mixture of Gaussian process experts," in *Advances in Neural Information Processing Systems*, 2006, pp. 883–890.
- [131] Y. Yang and J. Ma, "An efficient EM approach to parameter learning of the mixture of Gaussian processes," in *International Symposium on Neural Networks*, 2011, pp. 165–174.
- [132] Z. Chen, J. Ma, and Y. Zhou, "A precise hard-cut EM algorithm for mixtures of Gaussian processes," in *International Conference on Intelligent Computing*, 2014, pp. 68–75.
- [133] T. N. A. Nguyen, A. Bouzerdoum, and S. L. Phung, "Variational inference for infinite mixtures of sparse Gaussian processes through KL-correction," in *Acoustics, Speech and Signal Processing*, 2016, pp. 2579–2583.
- [134] C. Yuan and C. Neubauer, "Variational mixture of Gaussian process experts," in *Advances in Neural Information Processing Systems*, 2009, pp. 1897–1904.
- [135] M. Lázaro-Gredilla, S. Van Vaerenbergh, and N. D. Lawrence, "Overlapping mixtures of Gaussian processes for the data association problem," *Pattern Recognition*, vol. 45, no. 4, pp. 1386–1395, 2012.
- [136] J. Ross and J. Dy, "Nonparametric mixture of Gaussian processes with constraints," in *International Conference on Machine Learning*, 2013, pp. 1346–1354.
- [137] J. V. Hansen, "Combining predictors: Meta machine learning methods and bias/variance & ambiguity decompositions," Ph.D. dissertation, Aarhus University, Computer Science Department, 2000.
- [138] D. Nguyen-Tuong, M. Seeger, and J. Peters, "Model learning with local Gaussian process regression," *Advanced Robotics*, vol. 23, no. 15, pp. 2015–2034, 2009.
- [139] Z. Liu, L. Zhou, H. Leung, and H. P. Shum, "Kinect posture reconstruction based on a local mixture of Gaussian process models," *IEEE Transactions on Visualization and Computer Graphics*, vol. 22, no. 11, pp. 2437–2450, 2016.
- [140] M. Huang, R. Li, H. Wang, and W. Yao, "Estimating mixture of Gaussian processes by kernel smoothing," *Journal of Business & Economic Statistics*, vol. 32, no. 2, pp. 259–270, 2014.
- [141] L. Zhao, Z. Chen, and J. Ma, "An effective model selection criterion for mixtures of Gaussian processes," in *International Symposium on Neural Networks*, 2015, pp. 345–354.
- [142] S. P. Chatzis and Y. Demiris, "Nonparametric mixtures of Gaussian processes with power-law behavior," *IEEE Transactions on Neural Networks and Learning Systems*, vol. 23, no. 12, pp. 1862–1871, 2012.
- [143] G. E. Hinton, "Training products of experts by minimizing contrastive divergence," *Neural Computation*, vol. 14, no. 8, pp. 1771–1880, 2002.
- [144] Y. Cao and D. J. Fleet, "Generalized product of experts for automatic and principled fusion of Gaussian process predictions," *arXiv preprint arXiv:1410.7827*, 2014.
- [145] T. Chen and J. Ren, "Bagging for Gaussian process regression," *Neurocomputing*, vol. 72, no. 7-9, pp. 1605–1610, 2009.
- [146] Y. Okadome, Y. Nakamura, Y. Shikauchi, S. Ishii, and H. Ishiguro, "Fast approximation method for Gaussian process regression using hash function for non-uniformly distributed data," in *International Conference on Artificial Neural Networks*, 2013, pp. 17–25.
- [147] V. Tresp, "A Bayesian committee machine," *Neural Computation*, vol. 12, no. 11, pp. 2719–2741, 2000.
- [148] J. W. Ng and M. P. Deisenroth, "Hierarchical mixture-of-experts model for large-scale Gaussian process regression," *arXiv preprint arXiv:1412.3078*, 2014.
- [149] S. Mair and U. Brefeld, "Distributed robust Gaussian process regression," *Knowledge and Information Systems*, vol. 55, no. 2, pp. 415–435, 2018.
- [150] B. Szabo and H. van Zanten, "An asymptotic analysis of distributed nonparametric methods," *arXiv preprint arXiv:1711.03149*, 2017.
- [151] Y.-L. K. Samo and S. J. Roberts, "String and membrane Gaussian processes," *Journal of Machine Learning Research*, vol. 17, no. 1, pp. 4485–4571, 2016.
- [152] K. H. Low, J. M. Dolan, and P. Khosla, "Active markov information-theoretic path planning for robotic environmental sensing," in *International Conference on Autonomous Agents and Multiagent Systems*, 2011, pp. 753–760.
- [153] M. Franey, P. Ranjan, and H. Chipman, "A short note on Gaussian process modeling for large datasets using graphics processing units," *arXiv preprint arXiv:1203.1269*, 2012.
- [154] C. J. Paciorek, B. Lipshitz, W. Zhuo, C. G. Kaufman, R. C. Thomas et al., "Parallelizing Gaussian process calculations in R," *Journal of Statistical Software*, vol. 63, no. 1, pp. 1–23, 2015.
- [155] S. Ambikasaran, D. Foreman-Mackey, L. Greengard, D. W. Hogg, and M. O'Neil, "Fast direct methods for Gaussian processes," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 38, no. 2, pp. 252–265, 2016.
- [156] M. Tavassolipour, S. A. Motahari, and M.-T. M. Shalmani, "Learning of Gaussian processes in distributed and communication limited systems," *arXiv preprint arXiv:1705.02627*, 2017.
- [157] K. A. Wang, G. Pleiss, J. R. Gardner, S. Tyree, K. Q. Weinberger, and A. G. Wilson, "Exact Gaussian processes on a million data points," *arXiv preprint arXiv:1903.08114*, 2019.
- [158] J. Dean and S. Ghemawat, "MapReduce: simplified data processing on large clusters," *Communications of the ACM*, vol. 51, no. 1, pp. 107–113, 2008.
- [159] J. Gardner, G. Pleiss, K. Q. Weinberger, D. Bindel, and A. G. Wilson, "GPpyTorch: Blackbox matrix-matrix Gaussian process inference with GPU acceleration," in *Advances in Neural Information Processing Systems*, 2018, pp. 7576–7586.
- [160] J. Chen, K. H. Low, C. K.-Y. Tan, A. Oran, P. Jaillet, J. Dolan, and G. Sukhatme, "Decentralized data fusion and active sensing with mobile sensors for modeling and predicting spatiotemporal traffic phenomena," in *Uncertainty in Artificial Intelligence*, 2012, pp. 163–173.
- [161] J. Chen, N. Cao, K. H. Low, R. Ouyang, C. K.-Y. Tan, and P. Jaillet, "Parallel Gaussian process regression with low-rank covariance matrix approximations," in *Uncertainty in Artificial Intelligence*, 2013, pp. 152–161.
- [162] K. H. Low, J. Yu, J. Chen, and P. Jaillet, "Parallel Gaussian process regression for big data: Low-rank representation meets Markov approximation," in *AAAI Conference on Artificial Intelligence*, 2015.
- [163] Y. Ding, R. Kondor, and J. Eskreis-Winkler, "Multiresolution kernel approximation for Gaussian process regression," in *Advances in Neural Information Processing Systems*, 2017, pp. 3743–3751.
- [164] J. Vanhatalo, V. Pietiläinen, and A. Vehtari, "Approximate inference for disease mapping with sparse Gaussian processes," *Statistics in Medicine*, vol. 29, no. 15, pp. 1580–1607, 2010.
- [165] J. Vanhatalo and A. Vehtari, "Modelling local and global phenomena with sparse Gaussian processes," in *Uncertainty in Artificial Intelligence*, 2008, pp. 571–578.
- [166] D. Gu and H. Hu, "Spatial Gaussian process regression with mobile sensor networks," *IEEE Transactions on Neural Networks and Learning Systems*, vol. 23, no. 8, pp. 1279–1290, 2012.
- [167] T. D. Bui and R. E. Turner, "Tree-structured Gaussian process approximations," in *Advances in Neural Information Processing Systems*, 2014, pp. 2213–2221.
- [168] S. Park and S. Choi, "Hierarchical Gaussian process regression," in *Asian Conference on Machine Learning*, 2010, pp. 95–110.
- [169] E. Fox and D. B. Dunson, "Multiresolution Gaussian processes," in *Advances in Neural Information Processing Systems*, 2012, pp. 737–745.

- [170] D. Nychka, S. Bandyopadhyay, D. Hammerling, F. Lindgren, and S. Sain, "A multiresolution Gaussian process model for the analysis of large spatial datasets," *Journal of Computational and Graphical Statistics*, vol. 24, no. 2, pp. 579–599, 2015.
- [171] M. A. Álvarez and N. D. Lawrence, "Computationally efficient convolved multiple output Gaussian processes," *Journal of Machine Learning Research*, vol. 12, no. May, pp. 1459–1500, 2011.
- [172] M. van der Wilk, C. E. Rasmussen, and J. Hensman, "Convolutional Gaussian processes," in *Advances in Neural Information Processing Systems*, 2017, pp. 2845–2854.
- [173] F. Tobar, T. D. Bui, and R. E. Turner, "Learning stationary time series using Gaussian processes with nonparametric kernels," in *Advances in Neural Information Processing Systems*, 2015, pp. 3501–3509.
- [174] C. Walder, K. I. Kim, and B. Schölkopf, "Sparse multiscale Gaussian process regression," in *International Conference on Machine Learning*, 2008, pp. 1112–1119.
- [175] Z. Zhang, K. Duraisamy, and N. A. Gumerov, "Efficient multi-scale Gaussian process regression using hierarchical clustering," *arXiv preprint arXiv:1511.02258*, 2015.
- [176] E. Snelson and Z. Ghahramani, "Variable noise and dimensionality reduction for sparse Gaussian processes," in *Uncertainty in Artificial Intelligence*, 2006, pp. 461–468.
- [177] I. A. Almosallam, M. J. Jarvis, and S. J. Roberts, "GPZ: Non-stationary sparse Gaussian processes for heteroscedastic uncertainty estimation in photometric redshifts," *Monthly Notices of the Royal Astronomical Society*, vol. 462, no. 1, pp. 726–739, 2016.
- [178] P. W. Goldberg, C. K. Williams, and C. M. Bishop, "Regression with input-dependent noise: A Gaussian process treatment," in *Advances in Neural Information Processing Systems*, 1998, pp. 493–499.
- [179] M. Lázaro-Gredilla and M. K. Titsias, "Variational heteroscedastic Gaussian process regression," in *International Conference on Machine Learning*. Omnipress, 2011, pp. 841–848.
- [180] H. Liu, Y.-S. Ong, and J. Cai, "Large-scale heteroscedastic regression via Gaussian process," *arXiv preprint arXiv:1811.01179*, 2018.
- [181] Y. Yang, D. B. Dunson *et al.*, "Bayesian manifold regression," *The Annals of Statistics*, vol. 44, no. 2, pp. 876–905, 2016.
- [182] M. Chen, J. Silva, J. Paisley, C. Wang, D. Dunson, and L. Carin, "Compressive sensing on manifolds using a nonparametric mixture of factor analyzers: Algorithm and performance bounds," *IEEE Transactions on Signal Processing*, vol. 58, no. 12, pp. 6140–6155, 2010.
- [183] A. C. Damianou, M. K. Titsias, and N. D. Lawrence, "Variational inference for uncertainty on the inputs of Gaussian process models," *arXiv preprint arXiv:1409.2287*, 2014.
- [184] R. Calandra, J. Peters, C. E. Rasmussen, and M. P. Deisenroth, "Manifold Gaussian processes for regression," in *International Joint Conference on Neural Networks*, 2016, pp. 3338–3345.
- [185] B. J. Reich, H. D. Bondell, and L. Li, "Sufficient dimension reduction via Bayesian mixture modeling," *Biometrics*, vol. 67, no. 3, pp. 886–895, 2011.
- [186] R. Guhaniyogi and D. B. Dunson, "Compressed Gaussian process for manifold regression," *Journal of Machine Learning Research*, vol. 17, no. 69, pp. 1–26, 2016.
- [187] A. Damianou and N. Lawrence, "Deep Gaussian processes," in *Artificial Intelligence and Statistics*, 2013, pp. 207–215.
- [188] R. M. Neal, *Bayesian learning for neural networks*. Springer Science & Business Media, 1996, vol. 118.
- [189] A. G. d. G. Matthews, M. Rowland, J. Hron, R. E. Turner, and Z. Ghahramani, "Gaussian process behaviour in wide deep neural networks," *arXiv preprint arXiv:1804.11271*, 2018.
- [190] M. Al-Shedivat, A. G. Wilson, Y. Saatchi, Z. Hu, and E. P. Xing, "Learning scalable deep kernels with recurrent structure," *Journal of Machine Learning Research*, vol. 18, no. 1, pp. 2850–2886, 2017.
- [191] J. Bradshaw, A. G. d. G. Matthews, and Z. Ghahramani, "Adversarial examples, uncertainty, and transfer testing robustness in Gaussian process hybrid deep networks," *arXiv preprint arXiv:1707.02476*, 2017.
- [192] K. Cremanns and D. Roos, "Deep Gaussian covariance network," *arXiv preprint arXiv:1710.06202*, 2017.
- [193] T. Iwata and Z. Ghahramani, "Improving output uncertainty estimation and generalization in deep learning via neural network Gaussian processes," *arXiv preprint arXiv:1707.05922*, 2017.
- [194] Z. Dai, A. Damianou, J. González, and N. Lawrence, "Variational auto-encoded deep Gaussian processes," *arXiv preprint arXiv:1511.06455*, 2015.
- [195] T. Bui, D. Hernández-Lobato, J. Hernandez-Lobato, Y. Li, and R. Turner, "Deep Gaussian processes for regression using approximate expectation propagation," in *International Conference on Machine Learning*, 2016, pp. 1472–1481.
- [196] H. Salimbeni and M. Deisenroth, "Doubly stochastic variational inference for deep Gaussian processes," in *Advances in Neural Information Processing Systems*, 2017, pp. 4591–4602.
- [197] K. Cutajar, E. V. Bonilla, P. Michiardi, and M. Filippone, "Random feature expansions for deep Gaussian processes," in *International Conference on Machine Learning*, 2017, pp. 884–893.
- [198] M. Havasi, J. M. Hernández-Lobato, and J. J. Murillo-Fuentes, "Inference in deep Gaussian processes using stochastic gradient hamiltonian monte carlo," in *Advances in Neural Information Processing Systems*, 2018, pp. 7517–7527.
- [199] V. Kumar, V. Singh, P. Srijith, and A. Damianou, "Deep Gaussian processes with convolutional kernels," *arXiv preprint arXiv:1806.01655*, 2018.
- [200] A. C. Damianou, M. K. Titsias, and N. D. Lawrence, "Variational inference for latent variables and uncertain inputs in Gaussian processes," *Journal of Machine Learning Research*, vol. 17, no. 1, pp. 1425–1486, 2016.
- [201] E. V. Bonilla, K. M. Chai, and C. Williams, "Multi-task Gaussian process prediction," in *Advances in Neural Information Processing Systems*, 2008, pp. 153–160.
- [202] T. V. Nguyen, E. V. Bonilla *et al.*, "Collaborative multi-output Gaussian processes," in *Uncertainty in Artificial Intelligence*, 2014, pp. 643–652.
- [203] J. Zhao and S. Sun, "Variational dependent multi-output Gaussian process dynamical systems," *Journal of Machine Learning Research*, vol. 17, no. 1, pp. 4134–4169, 2016.
- [204] A. Chiplunkar, E. Rachelson, M. Colombo, and J. Morlier, "Approximate inference in related multi-output Gaussian process regression," in *International Conference on Pattern Recognition Applications and Methods*, 2016, pp. 88–103.
- [205] D. Xu, Y. Shi, I. W. Tsang, Y.-S. Ong, C. Gong, and X. Shen, "A survey on multi-output learning," *arXiv preprint arXiv:1901.00248*, 2019.
- [206] D. Petelin and J. Kocijan, "Evolving Gaussian process models for predicting chaotic time-series," in *IEEE Conference on Evolving and Adaptive Intelligent Systems*, 2014, pp. 1–8.
- [207] M. F. Huber, "Recursive Gaussian process: On-line regression and learning," *Pattern Recognition Letters*, vol. 45, pp. 85–91, 2014.
- [208] T. Le, K. Nguyen, V. Nguyen, T. D. Nguyen, and D. Phung, "GoGP: Fast online regression with Gaussian processes," in *IEEE International Conference on Data Mining*, 2017, pp. 257–266.
- [209] H. Bijl, J.-W. van Wingerden, T. B. Schön, and M. Verhaegen, "Online sparse Gaussian process regression using FITC and PITC approximations," *IFAC-PapersOnLine*, vol. 48, no. 28, pp. 703–708, 2015.
- [210] H. Bijl, T. B. Schön, J.-W. van Wingerden, and M. Verhaegen, "System identification through online sparse Gaussian process regression with input noise," *IFAC Journal of Systems and Control*, vol. 2, pp. 1–11, 2017.
- [211] C.-A. Cheng and B. Boots, "Incremental variational sparse Gaussian process regression," in *Advances in Neural Information Processing Systems*, 2016, pp. 4410–4418.
- [212] T. D. Bui, C. Nguyen, and R. E. Turner, "Streaming sparse Gaussian process approximations," in *Advances in Neural Information Processing Systems*, 2017, pp. 3301–3309.
- [213] D. Nguyen-Tuong, J. R. Peters, and M. Seeger, "Local Gaussian process regression for real time online model learning," in *Advances in Neural Information Processing Systems*, 2009, pp. 1193–1200.
- [214] N. Xu, K. H. Low, J. Chen, K. K. Lim, and E. B. Özgül, "GP-Localize: Persistent mobile robot localization using online sparse Gaussian process observation model," in *AAAI Conference on Artificial Intelligence*. AAAI Press, 2014, pp. 2585–2592.
- [215] C. V. Nguyen, T. D. Bui, Y. Li, and R. E. Turner, "Online variational bayesian inference: Algorithms for sparse Gaussian processes and theoretical bounds," in *ICML Time Series Workshop*, 2017, pp. 1–5.
- [216] J. Kocijan, A. Girard, B. Banko, and R. Murray-Smith, "Dynamic systems identification with Gaussian processes," *Mathematical and Computer Modelling of Dynamical Systems*, vol. 11, no. 4, pp. 411–424, 2005.
- [217] R. Frigola, Y. Chen, and C. E. Rasmussen, "Variational Gaussian process state-space models," in *Advances in Neural Information Processing Systems*, 2014, pp. 3680–3688.
- [218] C. L. C. Mattos, J. D. A. Santos, and G. A. Barreto, "An empirical evaluation of robust Gaussian process models for system identification," in *International Conference on Intelligent Data Engineering and Automated Learning*, 2015, pp. 172–180.
- [219] K. Azman and J. Kocijan, "Dynamical systems identification using Gaussian process models with incorporated local models," *Engineering Applications of Artificial Intelligence*, vol. 24, no. 2, pp. 398–408, 2011.

- [220] K. Worden, G. Manson, and E. J. Cross, "On Gaussian process narx models and their higher-order frequency response functions," in *Solving Computationally Expensive Engineering Problems*, 2014, pp. 315–335.
- [221] R. Frigola and C. E. Rasmussen, "Integrated pre-processing for Bayesian nonlinear system identification with Gaussian processes," in *IEEE Annual Conference on Decision and Control*, 2013, pp. 5371–5376.
- [222] R. Frigola, F. Lindsten, T. B. Schön, and C. E. Rasmussen, "Bayesian inference and learning in Gaussian process state-space models with particle MCMC," in *Advances in Neural Information Processing Systems*, 2013, pp. 3156–3164.
- [223] A. Svensson, A. Solin, S. Särkkä, and T. Schön, "Computationally efficient Bayesian learning of Gaussian process state space models," in *Artificial Intelligence and Statistics*, 2016, pp. 213–221.
- [224] C. L. C. Mattos, Z. Dai, A. Damianou, G. A. Barreto, and N. D. Lawrence, "Deep recurrent Gaussian processes for outlier-robust system identification," *Journal of Process Control*, vol. 60, pp. 82–94, 2017.
- [225] C. L. C. Mattos and G. A. Barreto, "A stochastic variational framework for recurrent Gaussian processes models," *Neural Networks*, vol. 114, no. April, pp. 54–72, 2019.
- [226] S. Hochreiter and J. Schmidhuber, "Long short-term memory," *Neural Computation*, vol. 9, no. 8, pp. 1735–1780, 1997.
- [227] H. Nickisch and C. E. Rasmussen, "Approximations for binary Gaussian process classification," *Journal of Machine Learning Research*, vol. 9, no. Oct, pp. 2035–2078, 2008.
- [228] H.-C. Kim and Z. Ghahramani, "Bayesian Gaussian process classification with the EM-EP algorithm," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 28, no. 12, pp. 1948–1959, 2006.
- [229] K. M. A. Chai, "Variational multinomial logit Gaussian process," *Journal of Machine Learning Research*, vol. 13, no. Jun, pp. 1745–1808, 2012.
- [230] F. Ruiz, M. Titsias, A. Dieng, and D. Blei, "Augment and reduce: Stochastic inference for large Categorical distributions," in *International Conference on Machine Learning*, vol. 10, 2018, pp. 6997–7006.
- [231] B. Fröhlich, E. Rodner, M. Kemmler, and J. Denzler, "Large-scale Gaussian process multi-class classification for semantic segmentation and facade recognition," *Machine Vision and Applications*, vol. 24, no. 5, pp. 1043–1053, 2013.
- [232] D. Milios, R. Camoriano, P. Michiardi, L. Rosasco, and M. Filippone, "Dirichlet-based Gaussian processes for large-scale calibrated classification," in *Advances in Neural Information Processing Systems*, 2018, pp. 6008–6018.
- [233] A. Naish-Guzman and S. Holden, "The generalized FITC approximation," in *Advances in Neural Information Processing Systems*, 2008, pp. 1057–1064.
- [234] J. Hensman, A. Matthews, and Z. Ghahramani, "Scalable variational Gaussian process classification," in *Artificial Intelligence and Statistics*, 2015, pp. 351–360.
- [235] D. Hernández-Lobato and J. M. Hernández-Lobato, "Scalable Gaussian process classification via expectation propagation," in *Artificial Intelligence and Statistics*, 2016, pp. 168–176.
- [236] C. Villacampa-Calvo and D. Hernández-Lobato, "Scalable multi-class Gaussian process classification using expectation propagation," in *International Conference on Machine Learning*. JMLR. org, 2017, pp. 3550–3559.
- [237] F. Wenzel, T. Galy-Fajou, C. Donner, M. Kloft, and M. Opper, "Efficient Gaussian process classification using pölya-gamma data augmentation," in *International Conference on Machine Learning*, 2018.
- [238] A. G. d. G. Matthews, "Scalable Gaussian process inference using variational methods," Ph.D. dissertation, University of Cambridge, 2017.
- [239] N. G. Polson, J. G. Scott, and J. Windle, "Bayesian inference for logistic models using pölya-Gamma latent variables," *Journal of the American Statistical Association*, vol. 108, no. 504, pp. 1339–1349, 2013.
- [240] T. Galy-Fajou, F. Wenzel, C. Donner, and M. Opper, "Scalable multi-class Gaussian process classification via data augmentation," in *NIPS workshop on approximate inference*, 2018.
- [241] C. Park, J. Z. Huang, and Y. Ding, "Domain decomposition approach for fast Gaussian process regression of large spatial data sets," *Journal of Machine Learning Research*, vol. 12, no. May, pp. 1697–1728, 2011.
- [242] F. Meier, P. Hennig, and S. Schaal, "Incremental local Gaussian regression," in *Advances in Neural Information Processing Systems*, 2014, pp. 972–980.
- [243] S. Flaxman, A. Wilson, D. Neill, H. Nickisch, and A. Smola, "Fast Kronecker inference in Gaussian processes with non-Gaussian likelihoods," in *International Conference on Machine Learning*, 2015, pp. 607–616.