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 regression (GPR), a well-known nonparametric, 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. However, they have not yet been comprehensively reviewed and analyzed 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 article is devoted to reviewing state-of-the-art scalable GPs involving two main categories: global approximations that distillate the entire data and local approximations that divide the data for subspace learning. Particularly, for global approximations, we mainly focus on sparse approximations comprising prior approximations that modify the prior but perform exact inference, posterior approximations that retain exact prior but perform approximate inference, and structured sparse approximations that 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 of scalable GPs in various scenarios are reviewed and discussed to inspire novel ideas for future research avenues.**

***Index Terms*— Big data, Gaussian process regression (GPR), local approximations, scalability, sparse approximations.**

I. Introduction

IN THE era of big data, the vast quantity of information poses the demand for effective and efficient analysis, interpretation, and prediction to explore the benefits lie ahead. Because of the big data, the machine learning community tells many success stories [1]–[4] while still leaving challenges. The Gaussian process regression (GPR) [5], also known as Kriging in geostatistics [6] and surrogates or emulators in computer experiments [7], is a nonparametric statistical model used in various scenarios, e.g., active learning [8], multitask 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 that 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 $\boldsymbol{X} = \{\boldsymbol{x}_i \in R^d\}_{i=1}^n$ and their observations $\boldsymbol{y} = \{y_i = y(\boldsymbol{x}_i) \in R\}_{i=1}^n$, GP seeks to infer the latent function $f : R^d \mapsto R$ in the functional space $\mathcal{GP}(m(\boldsymbol{x}), k(\boldsymbol{x}, \boldsymbol{x}'))$ defined by the mean $m(.)$ and the kernel $k(.,.)$. The most prominent weakness of standard GP is the cubic time complexity $\mathcal{O}(n^3)$ due to the inversion and determinant of the kernel matrix $\boldsymbol{K}_{nn} = k(\boldsymbol{X}, \boldsymbol{X}) \in R^{n\times n}$. This limits the scalability of GP and makes it unaffordable for large-scale data sets.

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

1) *Global Approximations:* It approximates the kernel matrix $\boldsymbol{K}_{nn}$ through global distillation. The distillation can be achieved by the following:
   a) a subset of the training data with m ($m \ll n$) points (subset of data (SoD) [14]), resulting in a smaller kernel matrix $\boldsymbol{K}_{mm}$;
   b) the remove of uncorrelated entries in $\boldsymbol{K}_{nn}$ (sparse kernels [15]), resulting in a sparse kernel matrix $\tilde{\boldsymbol{K}}_{nn}$ with many zero entries;
   c) the low-rank representation measured between m inducing points and n training points (sparse approximations [1], [16]–[18]), resulting in the Nyström approximation $\boldsymbol{K}_{nn} \approx \boldsymbol{K}_{nm}\boldsymbol{K}_{mm}^{-1}\boldsymbol{K}_{mn}$.
2) *Local Approximations:* It follows 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]. In addition, 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 $\mathcal{O}(nm^2)$. They can further speed up through parallel/distributed computing [20], [29]–[33]. When organizing the inducing points into the Kronecker structure, sparse approximations can further reduce the complexity to $\mathcal{O}(n)$ [18], [34]. In the meantime, by

Manuscript received July 2, 2018; revised April 9, 2019 and November 17, 2019; accepted November 28, 2019. This work was supported in part by the Rolls-Royce@NTU Corporate Laboratory, National Research Foundation (NRF) Singapore, through the Corp Lab@University Scheme, and in part by the Data Science and Artificial Intelligence Research Center (DSAIR) and the School of Computer Science and Engineering, Nanyang Technological University. *(Corresponding author: Haitao Liu.)*

H. Liu is with the Rolls-Royce@NTU Corporate Laboratory, Nanyang Technological University, Singapore 637460 (e-mail: htliu@ntu.edu.sg).

Y.-S. Ong, X. Shen, and J. Cai are with the School of Computer Science and Engineering, Nanyang Technological University, Singapore 639798 (e-mail: asysong@ntu.edu.sg; xbshen@ntu.edu.sg; asjfcai@ntu.edu.sg).

Color versions of one or more of the figures in this article are available online at http://ieeexplore.ieee.org.

Digital Object Identifier 10.1109/TNNLS.2019.2957109

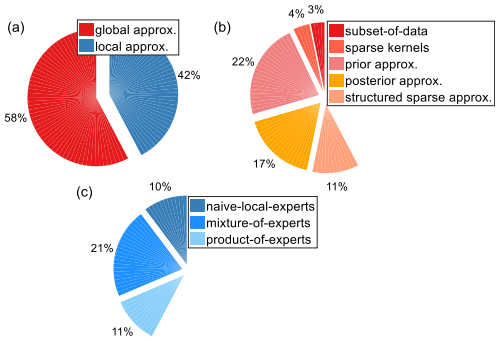


Fig. 1. Percentages of the categories for (a) scalable GPs including (b) global approximations and (c) local approximations in the literature surveyed.

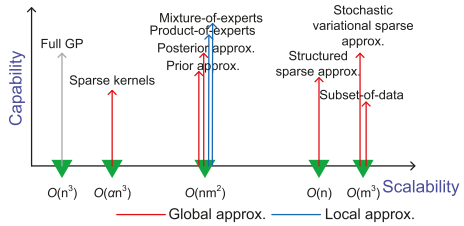


Fig. 2. Comparison of scalable GPs regarding scalability and model capability, where $0 < \alpha < 1$ and m is the inducing size for sparse approximations and the subset size for SoD and local approximations.

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 favorable predictions, i.e., good model capability. For example, though showing a remarkable complexity of $\mathcal{O}(m^3)$, we cannot expect the SoD 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 localization, local approximations favor capturing local patterns (nonstationary 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 overfitting. Recently, attempts have been made to improve the model capability through, for example, the interdomain 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 for comprehensive review, including the methodological characteristics and comparisons for better understanding. To the best of our knowledge, a detailed survey on various scalable GPs has not been conducted in the literature before.[1] Thus, such a review in the GP community is timely and important due to the explosion of data size.

To this end, we consider a skeletal overview in Fig. 3 to classify, review, and analyze state-of-the-art scalable GPs. Specifically, 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.

[1]The early survey [16] studies the prior approximations that are 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.

II. GP Regression Revisited

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

$$k_{\text{SE}}(\boldsymbol{x}, \boldsymbol{x}') = \sigma_f^2 \exp(-0.5(\boldsymbol{x} - \boldsymbol{x}')^{\mathsf{T}} \boldsymbol{\Delta}^{-1} (\boldsymbol{x} - \boldsymbol{x}')) \quad (1)$$

where $\boldsymbol{\Delta} = \text{diag}[l_1^2, \ldots, l_d^2]$ 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 $\mathcal{D} = \{\boldsymbol{X}, \boldsymbol{y}\}$, where $y(\boldsymbol{x}_i) = f(\boldsymbol{x}_i) + \epsilon$ with the i.i.d. noise $\epsilon \sim \mathcal{N}(0, \sigma_\epsilon^2)$, we obtain the model evidence (marginal likelihood) $p(\boldsymbol{y}|\boldsymbol{\theta}) = \int p(\boldsymbol{y}|\boldsymbol{f}) p(\boldsymbol{f}) d\boldsymbol{f} = \mathcal{N}(\boldsymbol{y}|\boldsymbol{0}, \boldsymbol{K}_{nn}^{\epsilon})$, where $\boldsymbol{K}_{nn}^{\epsilon} = \boldsymbol{K}_{nn} + \sigma_\epsilon^2 \boldsymbol{I}_n$. The hyperparameters $\boldsymbol{\theta}$ could be inferred by maximizing

$$\log p(\boldsymbol{y}) = -\frac{n}{2} \log 2\pi - \frac{1}{2} \log |\boldsymbol{K}_{nn}^{\epsilon}| - \frac{1}{2} \boldsymbol{y}^{\mathsf{T}} (\boldsymbol{K}_{nn}^{\epsilon})^{-1} \boldsymbol{y} \quad (2)$$

which automatically achieves the bias-variance tradeoff. We omit $\boldsymbol{\theta}$ from the conditioning of distribution for clarity.

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

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

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

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

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

$$f(\boldsymbol{x}) = \boldsymbol{\phi}(\boldsymbol{x})^{\mathsf{T}} \boldsymbol{w}, \quad y(\boldsymbol{x}) = f(\boldsymbol{x}) + \epsilon \quad (4)$$

where the Gaussian prior is placed on the weights as $p(\boldsymbol{w}) = \mathcal{N}(\boldsymbol{w}|\boldsymbol{0}, \boldsymbol{\Sigma})$, and $\boldsymbol{\phi}(\boldsymbol{x}) = [\phi_1(\boldsymbol{x}), \ldots, \phi_\upsilon(\boldsymbol{x})]^{\mathsf{T}}$ maps the d-dimensional input $\boldsymbol{x}$ into a υ-dimensional feature space. Equivalently, we derive the kernel as $k(\boldsymbol{x}, \boldsymbol{x}') = \boldsymbol{\phi}(\boldsymbol{x})^{\mathsf{T}} \boldsymbol{\Sigma} \boldsymbol{\phi}(\boldsymbol{x}')$. Particularly, the SE kernel (1) can be recovered from an infinite number ($\upsilon \to \infty$) of the Gaussian-shaped basis functions $\{\phi_c(\boldsymbol{x})\}_{c=1}^{\upsilon}$ centered everywhere.

The computational bottleneck in (2) is solving the linear system $(\boldsymbol{K}_{nn}^{\epsilon})^{-1} \boldsymbol{y}$ and the determinant $|\boldsymbol{K}_{nn}^{\epsilon}|$. Traditionally, we use the $\mathcal{O}(n^3)$ Cholesky decomposition $\boldsymbol{K}_{nn}^{\epsilon} = \boldsymbol{L}\boldsymbol{L}^{\mathsf{T}}$ such that $(\boldsymbol{K}_{nn}^{\epsilon})^{-1} \boldsymbol{y} = \boldsymbol{L}^{\mathsf{T}} \backslash (\boldsymbol{L} \backslash \boldsymbol{y})$ and $\log |\boldsymbol{K}_{nn}^{\epsilon}| = 2 \sum_{i=1}^{n} \log \boldsymbol{L}_{ii}$. As for predictions in (3), the mean costs $\mathcal{O}(n)$ and the variance costs $\mathcal{O}(n^2)$ per test case through precomputations.

The scalable GPs have been extensively presented and studied in recent years to improve the scalability for big data. In the following, we classify the scalable GPs into global approximations and local approximations and comprehensively analyze them to showcase the methodological characteristics.

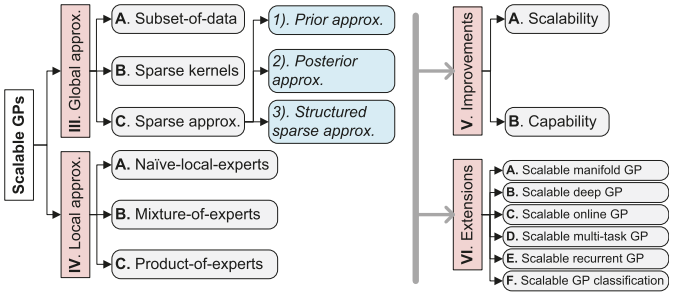


Fig. 3. Skeletal overview of scalable GPs.

III. Global Approximations

Global approximations achieve the sparsity of the full kernel matrix $\boldsymbol{K}_{nn}$, which is crucial for scalability, through: 1) using a subset of the training data (SoD); 2) removing the entries of $\boldsymbol{K}_{nn}$ with low correlations (sparse kernels); and 3) employing a low-rank representation (sparse approximations).

A. Subset of Data

SoD is the simplest strategy to approximate the full GP by using a subset $\mathcal{D}_{\text{sod}}$ of the training data $\mathcal{D}$. Hence, the SoD retains the standard GP inference at a lower time complexity of $\mathcal{O}(m^3)$ since it operates on $\boldsymbol{K}_{mm}$, 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 tradeoff 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 $\mathcal{D}_{\text{sod}}$, one could randomly choose m points from $\mathcal{D}$, use the 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 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 $\tilde{\boldsymbol{K}}_{nn}$ of $\boldsymbol{K}_{nn}$ via the particularly designed compactly supported (CS) kernel, which imposes $k(\boldsymbol{x}_i, \boldsymbol{x}_j) = 0$ when $|\boldsymbol{x}_i - \boldsymbol{x}_j|$ exceeds a certain threshold. Therefore, only the nonzero elements in $\tilde{\boldsymbol{K}}_{nn}$ are involved in the calculation. As a result, the GP using the CS kernel scales as $\mathcal{O}(\alpha n^3)$ with $0 < \alpha < 1$. The main challenge in constructing valid CS kernels is to build a positive semidefinite (PSD) $\tilde{\boldsymbol{K}}_{nn}$, i.e., $\boldsymbol{v}^\mathsf{T}\tilde{\boldsymbol{K}}_{nn}\boldsymbol{v} \geq 0 \ \forall \boldsymbol{v} \in R^n$ [15], [50]–[52]. Besides, the GP using the CS kernel is potential for capturing local patterns due to the truncation property.

C. Sparse Approximations

Typically, the eigendecomposition helps choose the first m eigenvalues to approximate the full-rank kernel matrix as $\boldsymbol{K}_{nn} \approx \boldsymbol{U}_{nm}\boldsymbol{\Lambda}_{mm}\boldsymbol{U}_{nm}^\mathsf{T}$. The inversion then can be calculated via the Sherman–Morrison–Woodbury formula

$$\left(\boldsymbol{K}_{nn}^{\epsilon}\right)^{-1} \approx \sigma_\epsilon^{-2}\boldsymbol{I}_n + \sigma_\epsilon^{-2}\boldsymbol{U}_{nm}\left(\sigma_\epsilon^2\boldsymbol{\Lambda}_{mm}^{-1} + \boldsymbol{U}_{nm}^\mathsf{T}\boldsymbol{U}_{nm}\right)^{-1}\boldsymbol{U}_{nm}^\mathsf{T}$$

and the determinant using the Sylvester determinant theorem

$$\left|\boldsymbol{K}_{nn}^{\epsilon}\right| \approx |\boldsymbol{\Lambda}_{mm}|\left|\sigma_\epsilon^2\boldsymbol{\Lambda}_{mm}^{-1} + \boldsymbol{U}_{nm}^\mathsf{T}\boldsymbol{U}_{nm}\right|$$

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

$$\boldsymbol{K}_{nn} \approx \boldsymbol{Q}_{nn} = \boldsymbol{K}_{nm}\boldsymbol{K}_{mm}^{-1}\boldsymbol{K}_{nm}^\mathsf{T}$$

which greatly improves the large-scale kernel learning [53] and enables the naive Nyström GP [54]. This scalable GP, however, may produce negative prediction variances [55] since it is not a complete generative probabilistic model as the Nyström approximation is only imposed on the training data and 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 pseudopoints) to optimally summarize the dependence of the whole training data. We introduce a set of inducing pairs $(\boldsymbol{X}_m, \boldsymbol{f}_m)$. The inducing variables $\boldsymbol{f}_m$ akin to $\boldsymbol{f}$ follow the same GP prior $p(\boldsymbol{f}_m) = \mathcal{N}(\boldsymbol{0}, \boldsymbol{K}_{mm})$. Besides, $\boldsymbol{f}_m$ is assumed to be a sufficient statistic for $\boldsymbol{f}$, i.e., for any variables $\boldsymbol{z}$, it holds $p(\boldsymbol{z}|\boldsymbol{f}, \boldsymbol{f}_m) = p(\boldsymbol{z}|\boldsymbol{f}_m)$. We could recover the joint prior $p(\boldsymbol{f}, f_*)$ by marginalizing out $\boldsymbol{f}_m$ as $p(\boldsymbol{f}, f_*) = \int p(\boldsymbol{f}, f_*|\boldsymbol{f}_m)p(\boldsymbol{f}_m)d\boldsymbol{f}_m$.

The sparse approximations have three main categories.

1) *Prior Approximations:* It approximates the prior but performs exact inference.
2) *Posterior Approximations:* It retains exact prior but performs approximate inference.
3) *Structured Sparse Approximations:* It exploits specific structures in the kernel matrix.

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

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

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

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

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

where $\boldsymbol{f}_m$ is called inducing variables since the dependencies between $\boldsymbol{f}$ and f_* are induced by $\boldsymbol{f}_m$. To obtain computational gains, we modify the training and test conditionals as

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

$$q(f_*|\boldsymbol{f}_m) = \mathcal{N}\left(f_*|\boldsymbol{k}_{*m}\boldsymbol{K}_{mm}^{-1}\boldsymbol{f}_m, \tilde{Q}_{**}\right). \tag{7b}$$

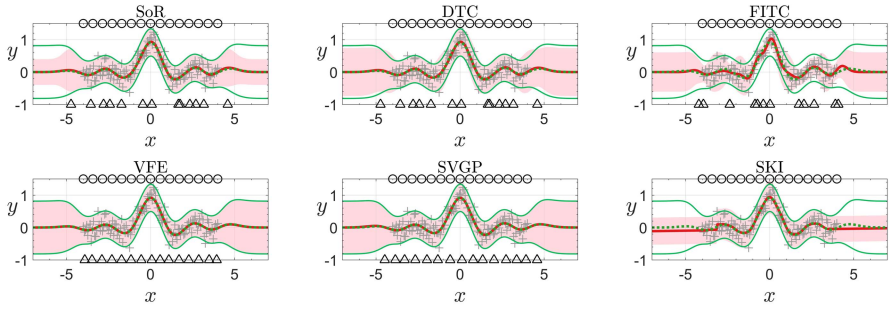


Fig. 4. Illustration of sparse approximations on a 1-D toy example with $y(x) = \text{sinc}(x) + \epsilon$, where $\epsilon \sim \mathcal{N}(0, 0.04)$. + symbols represent 120 training points. Top circles represent the initial locations of inducing points, whereas the bottom triangles represent the optimized locations of inducing points. Green dotted curves represent the prediction mean of full GP and green curves represent 95% confidence interval of the full GP predictions. 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: 1) the SoR suffers from overconfident prediction variance when leaving the training data; 2) the FITC captures heteroscedasticity in variance; and 3) 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.

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

$$\log q(\boldsymbol{y}) = -\frac{n}{2}\log 2\pi - \frac{1}{2}\log\left|\tilde{\boldsymbol{Q}}_{nn} + \boldsymbol{Q}_{nn} + \sigma_\epsilon^2 \boldsymbol{I}_n\right| - \frac{1}{2}\boldsymbol{y}^\mathsf{T}\left(\tilde{\boldsymbol{Q}}_{nn} + \boldsymbol{Q}_{nn} + \sigma_\epsilon^2 \boldsymbol{I}_n\right)^{-1}\boldsymbol{y}. \quad (8)$$

It is found that specific selections of $\tilde{\boldsymbol{Q}}_{nn}$ enable calculating $|\tilde{\boldsymbol{Q}}_{nn} + \boldsymbol{Q}_{nn} + \sigma_\epsilon^2 \boldsymbol{I}_n|$ and $(\tilde{\boldsymbol{Q}}_{nn} + \boldsymbol{Q}_{nn} + \sigma_\epsilon^2 \boldsymbol{I}_n)^{-1}$ 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., $\tilde{\boldsymbol{Q}}_{nn} = \boldsymbol{0}$ and $\tilde{Q}_{**} = 0$, as

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

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

This is equivalent to applying the Nyström approximation to both training and test data, resulting in a degenerate[2] GP with a rank (at most) m kernel

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

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 $\boldsymbol{x}$ in the feature space defined by dense basis functions $\{\phi_c(\boldsymbol{x})\}_{c=1}^v$ is equivalent to a Bayesian linear model in (4) with infinite weights. Hence, the relevance vector machine (RVM) [57] uses only m basis functions $\boldsymbol{\phi}_m(\boldsymbol{x}) = [\phi_1(\boldsymbol{x}), \ldots, \phi_m(\boldsymbol{x})]^\mathsf{T}$ for approximation

$$p(\boldsymbol{f}|\boldsymbol{w}) = \mathcal{N}\left(\boldsymbol{f}|\boldsymbol{\Phi}_{nm}\boldsymbol{w}, \boldsymbol{K}_{nn} - \boldsymbol{\Phi}_{nm}\boldsymbol{\Sigma}_{mm}\boldsymbol{\Phi}_{nm}^\mathsf{T}\right) \quad (10)$$

where $\boldsymbol{\Phi}_{nm} = [\boldsymbol{\phi}_m(\boldsymbol{x}_1), \ldots, \boldsymbol{\phi}_m(\boldsymbol{x}_n)]^\mathsf{T}$ and $p(\boldsymbol{w}) = \mathcal{N}(\boldsymbol{w}|\boldsymbol{0}, \boldsymbol{\Sigma}_{mm})$. As a consequence, the RVM is a GP from the function-space view with the kernel

$$k_{\text{RVM}}(\boldsymbol{x}_i, \boldsymbol{x}_j) = \boldsymbol{\phi}^\mathsf{T}(\boldsymbol{x}_i)\boldsymbol{\Sigma}_{mm}\boldsymbol{\phi}(\boldsymbol{x}_j)$$

which recovers k_{SoR} when $\boldsymbol{\Sigma}_{mm} = \boldsymbol{I}_m$ and $\boldsymbol{\phi}_m(\boldsymbol{x}) = \boldsymbol{L}^\mathsf{T}k^\mathsf{T}(x, \boldsymbol{X}_m)$, where $\boldsymbol{L}\boldsymbol{L}^\mathsf{T} = \boldsymbol{K}_{mm}^{-1}$ [36].[3] 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.[4]

To reverse the uncertainty behavior of SoR, the RVM is healed through augmenting the basis functions at $\boldsymbol{x}_*$ with, however, higher computing cost [60]. This augmentation by including f_* into $\boldsymbol{f}_m$ 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 the spectral representation (Fourier features), resulting in the stationary kernel

$$k(\boldsymbol{x}_i, \boldsymbol{x}_j) = \frac{\sigma_0^2}{m}\boldsymbol{\phi}_m^\mathsf{T}(\boldsymbol{x}_i)\boldsymbol{\phi}_m(\boldsymbol{x}_j) = \frac{\sigma_0^2}{m}\sum_{r=1}^{m}\cos\left(2\pi \boldsymbol{s}_r^\mathsf{T}(\boldsymbol{x}_i - \boldsymbol{x}_j)\right)$$

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

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

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

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 $\tilde{\boldsymbol{Q}}_{nn} = \boldsymbol{0}$.

Alternatively, the fully independent training conditional (FITC) [67] removes the dependence among $\{f_i\}_{i=1}^n$ such that given $\boldsymbol{V}_{nn} = \boldsymbol{K}_{nn} - \boldsymbol{Q}_{nn}$, the training conditional $q_{\text{FITC}}(\boldsymbol{f}|\boldsymbol{f}_m)$

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

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

[2]It means the kernel $k(.,.)$ has a finite number of nonzero eigenvalues.

[3]The choose of $\boldsymbol{\phi}_m(\boldsymbol{x})$ should produce a PSD kernel matrix such that $\boldsymbol{K}_{nn} - \boldsymbol{\Phi}_{nm}\boldsymbol{\Sigma}_{mm}\boldsymbol{\Phi}_{nm}^\mathsf{T} \succeq \boldsymbol{0}$. Alternatively, we can take the simple form $\boldsymbol{\phi}_m(\boldsymbol{x}) = \boldsymbol{\Lambda}_{mm}\boldsymbol{k}_m^\mathsf{T}(\boldsymbol{x})$, where $\boldsymbol{\Lambda}_{mm}$ is a diagonal matrix and $\boldsymbol{k}_m(\boldsymbol{x}) = k(\boldsymbol{x}, \boldsymbol{X}_m)$ [57]; or we take $\boldsymbol{\phi}_m(\boldsymbol{x}) = \boldsymbol{\Lambda}^{1/2}\boldsymbol{U}^\mathsf{T}\boldsymbol{k}_m^\mathsf{T}(\boldsymbol{x})$ with $\boldsymbol{\Lambda}$ and $\boldsymbol{U}$, respectively, being the eigenvalue matrix and eigenvector matrix of $\boldsymbol{K}_{mm}^{-1}$ [58], leading to a scaled Nyström approximation.

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

partially retains it, leading to a closer approximation to the prior $p(\boldsymbol{f}, f_*)$. Moreover, the full independence assumption can be extended to $q(f_*|\boldsymbol{f}_m)$ to derive the fully independent conditional (FIC) model[5] [16], which stands as a nondegenerate GP with the kernel

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

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

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

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

This is equivalent 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 $\boldsymbol{f}_i$ in each subset. However, it is argued that though being a closer approximation to $p(\boldsymbol{f}|\boldsymbol{f}_m)$, the blocking $q_{\text{PITC}}(\boldsymbol{f}|\boldsymbol{f}_m)$ brings little improvements over FITC [41]. This issue can be addressed by the extended partially independent conditional (PIC) [41] discussed in Section V-B.

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

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 $\boldsymbol{X}_m = \boldsymbol{X}$. However, this configuration is not the global optimum when maximizing $\log q(\boldsymbol{y})$, 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 in the following.

2) Posterior Approximations: Different from the prior approximations, the posterior approximations [1], [17] retain exact prior but perform approximate inference. The most well-known method is the elegant variational free energy (VFE) proposed by Titsias [17] by using variational inference (VI) [70]. Instead of modifying the prior $p(\boldsymbol{f}, f_*)$, VFE directly approximates the posterior $p(\boldsymbol{f}, \boldsymbol{f}_m|\boldsymbol{y})$, the learning of which is a central task in statistical models, by introducing a variational distribution $q(\boldsymbol{f}, \boldsymbol{f}_m|\boldsymbol{y})$. Then, we have their KL divergence $\text{KL}(q(\boldsymbol{f}, \boldsymbol{f}_m|\boldsymbol{y})||p(\boldsymbol{f}, \boldsymbol{f}_m|\boldsymbol{y}))$

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

where $\langle . \rangle_{q(.)}$ represents the expectation over the distribution $q(.)$.[6] It is found that minimizing the rigorously defined $\text{KL}(q||p) \geq 0$ is equivalent to maximizing F_q, since $\log p(\boldsymbol{y})$ is constant for $q(\boldsymbol{f}, \boldsymbol{f}_m|\boldsymbol{y})$. Thus, F_q is called evidence lower bound (ELBO) or VFE, which permits us to jointly optimize the variational parameters and hyperparameters. It is observed that maximizing F_q with respect to the hyperparameters directly improves F_q, while maximizing F_q with respect to the variational parameters implicitly drives the approximation to match both the posterior $p(\boldsymbol{f}, \boldsymbol{f}_m|\boldsymbol{y})$ and the evidence $p(\boldsymbol{y})$.

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

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

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 $\text{tr}[\boldsymbol{V}_{nn}] \geq 0$, which represents the total variance of predicting the latent variables $\boldsymbol{f}$ given $\boldsymbol{f}_m$. Particularly, $\text{tr}[\boldsymbol{V}_{nn}] = 0$ means $\boldsymbol{f}_m = \boldsymbol{f}$ and we recover the full GP. Hence, the trace term is a regularizer guards against over-fitting, seeks to deliver a good inducing set, and always improves F_q with increasing m (see the theoretical analysis in [69] and [71]). 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 overfitting [72].

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 [73]. The estimation of inducing points has also been improved in an augmented feature space [74], which is similar to the interdomain strategy [39]. The authors argued that the similarity of inducing points measured in the Euclidean space is inconsistent to that measured by the kernel. Hence, they assigned a mixture prior on $\boldsymbol{X}$ in the latent feature space and derived a regularized bound for choosing good inducing points in the kernel space. Besides, Matthews *et al.* [71] and Matthews [74] bridged the gap between the variational inducing-point framework and the more general KL divergence between the stochastic processes. Using this new interpretation, Bui *et al.* [76] approximated the general, infinite joint prior $p(\boldsymbol{f}_m, f_{\neq \boldsymbol{f}_m}, \boldsymbol{y}) = p(\boldsymbol{f}_m, f_{\neq \boldsymbol{f}_m}|\boldsymbol{y})p(\boldsymbol{y})$, 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

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

where $\log q(\boldsymbol{y})$ takes the form (8) with $\tilde{\boldsymbol{Q}}_{nn} = \alpha\text{diag}[\boldsymbol{V}_{nn}]$. By varying $\alpha \in (0, 1]$, we recover FITC when $\alpha = 1$ and VFE when $\alpha \to 0$. 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

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

[6]Matthews *et al.* [71] 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 f.

$q(\boldsymbol{f}_m|\boldsymbol{y}) = \mathcal{N}(\boldsymbol{f}_m|\boldsymbol{m}, \boldsymbol{S})$ in F_q to obtain a relaxed bound

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

The first term in the right-hand side of F_q is the sum of n terms due to the i.i.d. observation noises, i.e., $p(\boldsymbol{y}|\boldsymbol{f}) = \prod_{i=1}^{n} p(y_i|f_i)$. Hence, the stochastic gradient descent (SGD) [77], which encourages large-scale learning, could be employed to obtain an unbiased estimation of F_q using a minibatch $\{\boldsymbol{X}_b, \boldsymbol{y}_b\}$ as

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

Due to the difficulty of optimizing the variational parameters $\boldsymbol{m}$ and $\boldsymbol{S}$ in the Euclidean space, one can employ the stochastic VI (SVI) [78] using natural gradients,[7] resulting in a remarkable complexity of $\mathcal{O}(m^3)$ when $|\boldsymbol{y}_b| = 1$ 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]. Though showing high scalability and desirable approximation, the SVGP has some drawbacks: 1) the bound F_q is less tight than F_{VFE} because $q(\boldsymbol{f}_m|\boldsymbol{y})$ is not optimally eliminated; 2) it optimizes over $q(\boldsymbol{f}_m|\boldsymbol{y})$ with a huge number of variational parameters, thus requiring much time to complete one epoch of training; and 3) 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 allows using flexible basis functions to incorporate various low-rank structures and provides a composite nonconvex bound enabling the speedup using an asynchronous proximal gradient-based algorithm [80]. By deploying the variational model in a distributed machine learning platform PARAMETERSERVER [81], the authors have first scaled GP up to billions of data points. Similarly, Cheng and Boots [82] also derived a stochastic variational framework from the weight-space view with the difference being that the mean and variance of $p(\boldsymbol{f}|\boldsymbol{w})$, respectively, use the decoupled basis function sets $\boldsymbol{\phi}_a$ and $\boldsymbol{\phi}_b$, leading to more flexible inference. Besides, a recent interesting work [35] presents a novel unifying, anytime variational framework akin to Hensman's variational framework for accommodating the existing sparse approximations, e.g., SoR, DTC, FIT(C), and PIT(C), such that they can be trained via the efficient SGD, which achieves asymptotic convergence to the predictive distribution of the chosen sparse model. The key is to conduct a reverse VI in which "reverse" means that we can find a prior $p(\boldsymbol{f}_m) = \mathcal{N}(\boldsymbol{f}_m|\nu, \boldsymbol{\Lambda})$ (not the conventional GP prior) such that the variational distribution $q^*(\boldsymbol{f}_m|\boldsymbol{y}) = p(\boldsymbol{f}_m|\boldsymbol{y})$ for FI(T)C and PI(T)C is the maximum of ELBO.[8] Finally, the scalability of SVGP can be further reduced to nearly $\mathcal{O}(m)$ by introducing Kronecker structures for inducing points and the variance of $q(\boldsymbol{f}_m|\boldsymbol{y})$ [83], [84].

Titsias and Hensman's models have been further improved by using: 1) the Bayesian treatment of hyperparameters [85]–[87] rather than traditional point estimation which risks overfitting when the number of hyperparameters is small and 2) the non-Gaussian likelihoods [86], [88], [89].

3) Structured Sparse Approximations: A direct speedup to solve $(\boldsymbol{K}_{nn}^{\epsilon})^{-1}\boldsymbol{y}$ in standard GP can be achieved through fast matrix-vector multiplication (MVM) [90], [91], which iteratively solves the linear system using conjugate gradients (CGs) with s ($s \ll n$) iterations,[9] resulting in a time complexity of $\mathcal{O}(sn^2)$. It was argued in [14] that the original MVM has some open issues, e.g., the determination of s, the lack of meaningful speedups, and the badly conditioned kernel matrix. Alternatively, the preconditioned CG (PCG) [92] employs a preconditioning 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 $\boldsymbol{K}_{nn}$ itself has some algebraic structure, the MVM provides massive scalability. For example, the Kronecker methods [93], [94] exploit the multivariate grid inputs $\boldsymbol{x} \in \Omega_1 \times \cdots \times \Omega_d$ and the tensor product kernel with the form $k(\boldsymbol{x}_i, \boldsymbol{x}_j) = \prod_{t=1}^{d} k(\boldsymbol{x}_i^t, \boldsymbol{x}_j^t)$.[10] Then, the kernel matrix decomposes to a Kronecker product $\boldsymbol{K}_{nn} = \boldsymbol{K}_1 \otimes \cdots \otimes \boldsymbol{K}_d$, which eases the eigendecomposition with a greatly reduced time complexity of $\mathcal{O}(d\overline{n}^{d+1})$, where $\overline{n} = \sqrt[d]{n}$ for $d > 1$.[11] Another one is the Toeplitz methods [95]—complementary to the Kronecker methods—that exploit the kernel matrix built from regularly spaced 1-D points, resulting in the time complexity of $\mathcal{O}(d\overline{n}^d \log \overline{n})$. 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.[12]

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 $\boldsymbol{K}_{mm}$ admits the Kronecker structure for $d > 1$ and the Toeplitz structure for $d = 1$, whereas the cross kernel matrix $\boldsymbol{K}_{nm}$ is approximated, for example by a local linear interpolation using the adjacent grid inducing points as

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

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

$$\boldsymbol{Q}_{nn} \approx \boldsymbol{W}_{nm} \boldsymbol{K}_{mm}^{-1} \boldsymbol{W}_{nm}^{\mathsf{T}} \quad (20)$$

where the weight matrix $\boldsymbol{W}$ is extremely sparse since it only has two nonzero entires per row for local linear interpolation, leading to an impressive time complexity of $\mathcal{O}(n + d\overline{m}^{d+1})$ with $\overline{m} = \sqrt[d]{m}$ for solving $(\boldsymbol{K}_{nn}^{\epsilon})^{-1}\boldsymbol{y}$. Also, the sparse $\boldsymbol{W}$ incurs the prediction mean with constant-time complexity $\mathcal{O}(1)$ and the prediction variance with complexity $\mathcal{O}(m)$ after precomputing. Furthermore, Pleiss *et al.* [97] derived a constant-time prediction variance using the Lanczos approximation, which admits s iterations of the MVM for calculation.

[7] The superiority of natural gradients for GPR has been verified in [79].

[8] VFE and SVGP predefine the prior $p(\boldsymbol{f}_m) = \mathcal{N}(\boldsymbol{f}_m|\boldsymbol{0}, \boldsymbol{K}_{mm})$ and then find an optimal $q^*(\boldsymbol{f}_m|\boldsymbol{y})$ that maximizes the ELBO.

[9] The solution to $\boldsymbol{A}\boldsymbol{x} = \boldsymbol{b}$ is the unique minimum of the quadratic function $0.5\boldsymbol{x}^{\mathsf{T}}\boldsymbol{A}\boldsymbol{x} - \boldsymbol{x}^{\mathsf{T}}\boldsymbol{b}$.

[10] Popular kernels, such as the SE kernel (1), fulfill the product structure.

[11] $\boldsymbol{K}_i$ ($1 \le i \le d$) is an $\overline{n} \times \overline{n}$ 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 [94, Sec. 2.2].

[12] This limitation was relaxed in the partial grid and variable noise scenario by introducing virtual observations and inputs [94], [96].

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 [98], or more interestingly, one can use the hierarchical structure of NNs to extract the latent low-dimensional feature space [34], [99]. Furthermore, continual efforts [84], [100], [101] have been made to directly reduce the time complexity to be linear with d by exploiting the row-partitioned Khatri–Rao structure of $\boldsymbol{K}_{nm}$ or imposing tensor train decomposition and Kronecker product to the mean and variance of $q(\boldsymbol{f}_m|\boldsymbol{y})$ 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 [100] exploited the row-partitioned Khatri–Rao structure of $\boldsymbol{K}_{nm}$ rather than using local weight interpolation. To have sensible uncertainty, a diagonal correlation akin to that of FITC has been considered [100], [102].

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

Choosing Inducing Points: So far, we have reviewed state-of-the-art sparse approximations. Regarding their implementations, the number and location of inducing points is crucial. As for the inducing size, theoretical analysis [103] indicates that the KL divergence between the variational approximation and the posterior can be arbitrarily small when m grows more slowly than n for regression with normally distributed inputs and the SE kernel. As for the location of inducing points, alternatively, we could use clustering techniques to select a finite set of space-filling inducing points from $\mathcal{D}$ or employ some querying criteria [49], [56], [66], [104], [105] 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 [106] shows the first attempt to simultaneously determine the number and locations of inducing points in the Bayesian framework by placing a prior on $\boldsymbol{X}_m$.

IV. Local Approximations

Inspired by D&C, local approximations use localized experts to improve the scalability of GP. Besides, compared to global approximations, the localization enables capturing nonstationary features. In what follows, we comprehensively review the naive local experts that directly employs the pure local experts for prediction, and the mixture of experts (MoEs) and product of experts (PoEs) that inherit the advantages of naive local experts but boost the predictions through model averaging.

A. Naive Local Experts

It is known that a pair of points far away from each other is lowly correlated. Hence, the local experts trained on subsets of $\mathcal{D}$ is expected to produce sensible predictions with low time complexity. Particularly, the simple naive local experts (NLEs) [107], [108] let the local expert $\mathcal{M}_i$ be completely responsible for the subregion $\boldsymbol{\Omega}_i$ defined by $\boldsymbol{X}_i$. Mathematically, we predict at $\boldsymbol{x}_* \in \boldsymbol{\Omega}_i$ as $p(y_*|\mathcal{D}, \boldsymbol{x}_*) \approx p_i(y_*|\mathcal{D}_i, \boldsymbol{x}_*)$.

According to the partition of $\mathcal{D}$, we classify NLE into two main categories: 1) inductive NLE, which first partitions the input space and trains all the experts and then chooses an appropriate one for predicting at $\boldsymbol{x}_*$ and 2) transductive NLE, which particularly chooses a neighborhood subset $\mathcal{D}_*$ around $\boldsymbol{x}_*$ and trains the relevant expert $\mathcal{M}_*$ for predicting at $\boldsymbol{x}_*$.

Inductive NLE employs a static partition of $\mathcal{D}$ using clustering techniques, e.g., the 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 $\boldsymbol{x}_*$, 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 to choose the neighborhood set $\mathcal{D}_*$ around $\boldsymbol{x}_*$. The simplest way is to use the geometric closeness criteria[13] 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].

While enjoying the capability of capturing nonstationary features due to the localized structure, the NLE produces the discontinuous predictions on the boundaries of subregions and suffers from poor generalization capability since it misses the long-term spatial correlations, as shown 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. However, the patched GPs suffer from inconsistent and even negative prediction variance, and are only available in low dimensions [106], [118]. More popularly, the model averaging strategy, which is accomplished by the mixture/product of local GPs elaborated next, well smooths the predictions from multiple experts. To address the generalization issue, it is possible to 1) share the hyperparameters across experts, such as [26]; or 2) combine local approximations with global approximations, which will be reviewed in Section V-B.

B. Mixture of Experts

The MoE devotes to combining the local and diverse experts owning individual parameters for improving the overall accuracy and reliability [21], [22].[14] As shown in Fig. 6, MoE generally expresses the combination as a Gaussian mixture

[13]The selected points should be close to $\boldsymbol{x}_*$; meanwhile, they should distribute uniformly to avoid conveying redundant information.

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

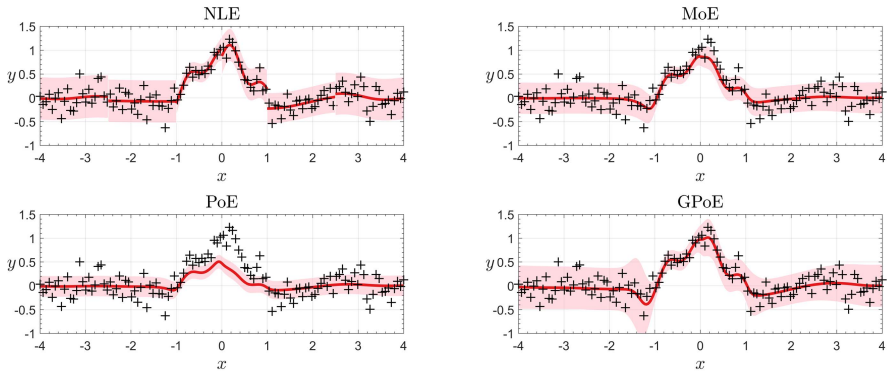


Fig. 5. Illustration of local approximations using six individual experts on the toy example. Note that for MoE, we did not jointly learn the experts and gating functions. For simplicity, we use the individual experts and the differential entropy as β_i in the softmax gating function. It is found that 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, such as the MoE, to provide desirable predictions; or use input-dependent weights, such as GPoE, to boost the predictions.

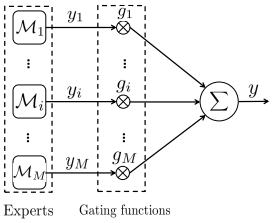


Fig. 6. Illustration of MoE.

model (GMM) [120]

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

where $g_i(\boldsymbol{x})$ is the gating function, which usually takes a parametric form such as 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., $\boldsymbol{x}$ is assigned to expert $\mathcal{M}_i$; $p_i(y|\boldsymbol{x})$ comes from $\mathcal{M}_i$ and is responsible for component i of the mixture. The gates in (21) manage the mixture through probabilistic partition of the input space for defining the subregions where the individual experts are responsible for. The experts can be a variety of machine learning models, e.g., the linear model and the support vector machines [122], [123].

The training of MoE usually assumes that the data are i.i.d. such that we maximize the factorized log likelihood $\sum_{t=1}^{n} \log p(y_t|\boldsymbol{x}_t)$ 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 probabilistic (soft) partition of the input space via both the data and the experts themselves, and diverse experts specified for different but overlapped subregions. Finally, the predictive distribution at $\boldsymbol{x}_*$ is

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

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

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

Regarding the mixture of GP experts for big data, it is observed that the original MoE is designed for multi-modal (nonstationary) modeling, and the individual global experts are responsible for all the data points, leading to high complexity; the i.i.d. data assumption does not hold here since GP fits the data dependences through joint distribution; and the parametric gating function g_i is not favored in the Bayesian nonparametric framework. In 2001, Tresp [129] first introduced the mixture of GP experts. It 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: 1) how to reduce the computational complexity (model complexity) and 2) 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 infinite mixture of GP experts (iMGPE) [23] uses a localized likelihood to get rid of the i.i.d. assumption as

$$p(\boldsymbol{y}|\boldsymbol{X}) = \sum_{z \in \mathcal{Z}} p(\boldsymbol{z}|\boldsymbol{X}) \prod_{i} p(\boldsymbol{y}_i|\boldsymbol{z}, \boldsymbol{X}_i). \tag{23}$$

Given an instance of the expert indicators $\boldsymbol{z} = [z_1, \ldots, z_n]^{\mathsf{T}}$, 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 $p(\boldsymbol{y}, \boldsymbol{X})$ as [130]

$$p(\boldsymbol{y}, \boldsymbol{X}) = \sum_{z \in \mathcal{Z}} p(\boldsymbol{z}) \prod_{i} p(\boldsymbol{y}_i|\boldsymbol{z}, \boldsymbol{X}_i) p(\boldsymbol{X}_i|\boldsymbol{z}). \tag{24}$$

The fully generative model is capable of handling partially specified data and providing inverse functional mappings. However, 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, in which the E-step assigns the data to experts through maximum *a posteriori* (MAP) of the expert indicators $\boldsymbol{z}$ or a threshold value [42], [131]–[133]. Thereafter, the M-step only operates on small subsets.

The second thread is to combine global experts with the sparse approximations reviewed in Section III-C under the variational EM framework. The dependence among inputs is broken to make VI feasible by: 1) interpreting GP as the finite Bayesian linear model in (10) [24], [134] or 2) using the FITC experts that factorize over $\boldsymbol{f}$ given the inducing set $\boldsymbol{f}_m$ [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 a mixture of implicitly localized experts (MILE) [22]. The implicit partition determines the optimal allocation of experts, thus capturing the interaction among the 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 fail due to the zero-coefficient problem caused by unreasonable initial parameters [137].

To relieve the problem of MILE, the third thread is to prepartition the input space by clustering techniques and assign points to the experts before training [138], [139]. The mixture of explicitly localized experts (MELE) [22] reduces the model complexity as well and 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 $\boldsymbol{z}$ to automatically infer the number of experts from data. The model inference, however, has to resort to a stick-breaking representation of DP [24], [133] due to the complex prior and the infinite M.

C. Product of Experts

Different from the MoE that employs a weighted sum of several probability distributions (experts) via an "*OR*" operation, the PoEs [25] multiplies these probability distributions, which sidesteps the weight assignment in MoE and is similar to an "AND" operation as

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

where Z is a normalizer, which, however, makes the inference intractable when maximizing the likelihood $\sum_{i=1}^{n}\log p(y_i|\boldsymbol{x}_i)$ [25].

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

$$p(\boldsymbol{y}|\boldsymbol{X}) = \prod_{i=1}^{M} p(\boldsymbol{y}_i|\boldsymbol{X}_i) \tag{26}$$

where $p_i(\boldsymbol{y}_i|\boldsymbol{X}_i) \sim \mathcal{N}(\boldsymbol{y}_i|\boldsymbol{0}, \boldsymbol{K}_i + \sigma_{\epsilon,i}^2\boldsymbol{I}_{n_i})$ with $\boldsymbol{K}_i = k(\boldsymbol{X}_i, \boldsymbol{X}_i) \in R^{n_i\times n_i}$ and n_i being the training size of expert $\mathcal{M}_i$. This factorization degenerates the full kernel matrix $\boldsymbol{K}_{nn}$ into a diagonal block matrix $\text{diag}[\boldsymbol{K}_1, \cdots, \boldsymbol{K}_M]$, leading to $\boldsymbol{K}_{nn}^{-1} \approx \text{diag}[\boldsymbol{K}_1^{-1}, \cdots, \boldsymbol{K}_M^{-1}]$. Hence, the complexity is substantially reduced to $\mathcal{O}(nm_0^2)$ given $n_i = m_0$.

It is found that the PoE likelihood (26) is a "point estimation" of the MoE likelihood (23); the MoE likelihood averages the PoE likelihood over possible configurations of the expert indicators $\boldsymbol{z}$. Consequently, the joint learning of gating functions and experts makes MoE achieve the optimal allocation of experts [143]. 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 the 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.[15] Particularly, the aggregations are expected to have several properties [143]: 1) the aggregated prediction is sensible in terms of probability and 2) the aggregated prediction is robust to weak experts.

Given the GP experts $\{\mathcal{M}_i\}_{i=1}^M$ with predictive distributions $\{p(y_*|\mathcal{D}_i, \boldsymbol{x}_*) = \mathcal{N}(\mu_i(\boldsymbol{x}_*), \sigma_i^2(\boldsymbol{x}_*))\}_{i=1}^M$ at $\boldsymbol{x}_*$, the PoEs [25], [143]–[145] aggregate the experts' predictions through a modified product rule as

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

where β_{*i} is a weight quantifying the contribution of $p_i(y_*|\mathcal{D}_i, \boldsymbol{x}_*)$ at $\boldsymbol{x}_*$. 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 $\sigma^{-2}(\boldsymbol{x}_*) = \sum_{i=1}^M \sigma_i^{-2}(\boldsymbol{x}_*)$, which will explode rapidly with increasing M. To alleviate the overconfident uncertainty, the generalized PoE (GPoE) [143] 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. However, 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 employ a simple weight $\beta_{*i} = 1/M$ such that the GPoE recovers the GP prior when leaving $\boldsymbol{X}$, at the cost of, however, producing underconfident prediction variance [26].[16]

Alternatively, the Bayesian committee machine (BCM) [26], [146]–[148] aggregates the experts' predictions from another point of view by imposing a conditional independence assumption $p(\boldsymbol{y}|y_*) \approx \prod_{i=1}^M p(\boldsymbol{y}_i|y_*)$, which in turn explicitly introduces a common prior $p(y_*|\boldsymbol{\theta})$ for the experts.[17] Thereafter,

[15] Note that the aggregation strategy is postprocessing or transductive since it is independent of model training but depends on the test point location.

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

[17] The BCM is a sparse GP treating the test inputs as inducing points [16].

by using the Bayes rule, we have

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

The prior correlation term helps BCM recover the GP prior when leaving $\boldsymbol{X}$, and the varying β_{*i} akin to that of GPoE helps produce robust BCM (RBCM) predictions within $\boldsymbol{X}$ [26]. The BCM, however, will produce unreliable prediction mean when leaving $\boldsymbol{X}$, which has been observed and analyzed in [26] and [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 $\boldsymbol{\theta}$.

It has been pointed out that the conventional PoEs and BCMs are inconsistent [28], [149], that is, their aggregated predictions cannot recover that of full GP when $n \to \infty$. To raise consistent aggregation, the nested pointwise aggregation of experts (NPAE) [27] removes the independence assumption by assuming that $\boldsymbol{y}_i$ has not yet been observed such that $\mu_i(\boldsymbol{x}_*)$ 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, the generalized RBCM (GRBCM) [28] introduces a global communication expert $\mathcal{M}_c$ rather than the fixed GP prior to perform correction, i.e., acting as a base expert, and considers the covariance between global and local experts to support consistent predictions when $n \to \infty$. It results in the aggregation

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

where $p_{+i}(y_*|\mathcal{D}_{+i}, \boldsymbol{x}_*)$ is the predictive distribution of the expert $\mathcal{M}_{+i}$ trained on the augmented data set $\mathcal{D}_{+i} = \{\mathcal{D}_i, \mathcal{D}_c\}$.

Note that these transductive aggregations usually share the hyperparameters across experts [26], i.e., $\boldsymbol{\theta}_i = \boldsymbol{\theta}$, because 1) it achieves automatic regularization and eases the inference due to fewer hyperparameters; 2) it allows to temporarily ignore the noise term of GP in aggregation, i.e., using $p_i(f_*|\mathcal{D}_i, \boldsymbol{x}_*)$ instead of $p_i(y_*|\mathcal{D}_i, \boldsymbol{x}_*)$ such as [26], to relieve the inconsistency of typical aggregations; and finally 3) the BCMs cannot support the experts owning individual hyperparameters as discussed before. However, the shared hyperparameters limit the capability of capturing nonstationary features, which is the superiority of local approximations.[18] Besides, another main drawback of aggregations is the Kolmogorov inconsistency [150] 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., $\boldsymbol{x}_*$ and $\boldsymbol{x}'_*$, we have $p(y_*|\mathcal{D}) \neq \int p(y_*, y'_*|\mathcal{D})dy'_*$.

V. Improvements Over Scalable GPs

A. Scalability

The 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 [151]. Alternatively, we can implement sparse approximations using advanced computing infrastructure, e.g., graphics processing units (GPUs) and distributed clusters/processors, to further speed up the computation.

Actually, the exact GP using GPU and distributed clusters has been investigated [152]–[156] in the regime of distributed learning [157]. It implements parallel and fast linear algebra algorithms, e.g., the HODLR algorithm [154] and the MVM algorithm, with modern distributed memory and multicore/multi-GPU hardware. For instance, Wang *et al.* [156] successfully trained an MVM-based exact GP over a million data points in 3 days through eight GPUs.

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 ELBO (17) or grid-inducing points in (20), the TensorFlow-based GPflow library [31] and the PyTorch-based *GPyTorch* library [158] have been developed to exploit the usage of GPU hardwares.

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 [159], [160]. Ideally, the parallelization can achieve a speedup 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) [161], 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 2 h.

The local approximations generally have the same complexity as the global approximations if $m_0 = m$. The local opinion naturally encourages the parallel/distributed implementations to further reduce computational complexity (see [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 is dominated by a few large eigenvectors. In contrast, when the latent function f has quick-varying (nonstationary) features, e.g., the complicated time-series tasks [162], the limited global inducing set is hard 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], [163], [164]. For instance, after partitioning the input space into subregions, the PIC [41] and its stochastic and distributed variants [35], [43], [87] extend PITC by retaining the conditional independence of training and test, i.e., $\boldsymbol{f} \perp f_*|\boldsymbol{f}_m$, for all the

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

subregions except for the one containing the test point $\boldsymbol{x}_*$. Thus, it enables the integration of local and global approximations in a transductive pattern. Mathematically, suppose that $\boldsymbol{x}_* \in \boldsymbol{\Omega}_j$, and we have

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

This model corresponds to an exact GP with an additive kernel

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

where $\psi_{ij} = 1$ when $\boldsymbol{x}_i$ and $\boldsymbol{x}_j$ belong to the same block; otherwise, $\psi_{ij} = 0$. Note that PIC recovers FIC by taking all the subregion sizes to one; PIC becomes the purely local GPs by taking the inducing size to zero. The additive kernel similar to k_{PIC} has also been employed in [163]–[165] by combining the CS kernel [15], [52] and the sparse approximation. Furthermore, as an extension of PIC, the tree-structured GP [166] ignores most of the intersubregion dependences of inducing points, but it concentrates on the dependence of the 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 multiresolution [40], [167]–[169]. For example, Lee *et al.* [40] extended the work of [166] 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 that is convolved from two relevant kernels, such as [170]. Similarly, Park and Choi [167] presented a two-layer model, in which a GP is placed over the centroids of the subsets as $g(\boldsymbol{c}) \sim \mathcal{GP}(0, k_g(\boldsymbol{c}, \boldsymbol{c}'))$ 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 $f_i(\boldsymbol{x}) \sim \mathcal{GP}(g(\boldsymbol{x}), k_i(\boldsymbol{x}, \boldsymbol{x}'))$. This model has also been improved into multilayer structure [168].

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], [166] 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 alleviate the discontinuity. To completely address the discontinuity, Nguyen and Bonilla [42] and Nguyen *et al.* [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 interdomain GP [39], which employs an idea similar to the convolution process in multioutput GP [10], [170] and high-dimensional GP [171]. Particularly, it uses a linear integral transform $g(\boldsymbol{z}) = \int w(\boldsymbol{x}, \boldsymbol{z}) f(\boldsymbol{x}) d\boldsymbol{x}$ to map the inducing points into another domain with possibly different dimensions.[19] The inducing variables in the new domain can own a new kernel and induce richer dependences in the old domain. The interdomain idea has also been applied to the posterior approximations [71], [76], [172]. 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 [173], [174]. This kind of weight-space nonstationary GP indeed can be derived from the interdomain 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 $p(\boldsymbol{\epsilon}) = \mathcal{N}(\boldsymbol{\epsilon}|\boldsymbol{0}, \text{diag}[\boldsymbol{h}])$ where $\boldsymbol{h} = [\sigma_\epsilon^2(\boldsymbol{x}_1), \cdots, \sigma_\epsilon^2(\boldsymbol{x}_n)]^\top$ [175]. Moreover, Hoang *et al.* [43] employed a B-th order Markov property on the correlated noise process $p(\boldsymbol{\epsilon}) = \mathcal{N}(\boldsymbol{\epsilon}|\boldsymbol{0}, \boldsymbol{K}_\epsilon)$ in a distributed variational framework. The unifying framework accommodates the existing sparse approximations, e.g., DTC and PIC, by varying the Markov order and noise structure. Yu *et al.* [87] further extended this article through Bayesian treatment of hyperparameters to guard against overfitting. More elegantly, Almosallam *et al.* [176] derived a scalable heteroscedastic Bayesian model from the weight-space view by adopting an additional log GP, which is analogous to [177] and [178], to account for noise variance as $\sigma_\epsilon^2(\boldsymbol{x}_i) = \exp(\boldsymbol{\phi}(\boldsymbol{x}_i)\boldsymbol{w} + b)$. Differently, Liu *et al.* [179] derived the stochastic and distributed variants of [177] for scalable heteroscedastic regression. 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. Finally, the heteroscedasticity can also be encoded though augmenting the inputs with latent variables [180], [181]. This has also been exploited in the regime of density estimation [182].

VI. Extensions and Open Issues

A. Scalable Manifold GP

In scalable GP literature, we usually focus on the scenario in which 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 [183].

Hence, various manifold GPs [11], [175], which are expressed as

$$y = f(\boldsymbol{\Upsilon}\boldsymbol{x}) + \epsilon \tag{31}$$

where $\boldsymbol{\Upsilon} \in R^{p\times d}$ is a mapping matrix, have been developed to tackle high-dimensional big data through linear/nonlinear dimensionality reduction [175], [184], [185] or NN-like input transformation [186]. As a result, the kernel operates the data in a lower dimensional space as $k(\boldsymbol{\Upsilon}\boldsymbol{x}, \boldsymbol{\Upsilon}\boldsymbol{x}')$. 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 the Bayesian mixture models [187], which, however, induce a heavy computational budget.

A recent exciting theoretical finding [183] turns out that the learning of the intrinsic manifold can be bypassed since

[19] $g(\boldsymbol{z})$ is still a GP by a linear transform of $f(\boldsymbol{x})$. Besides, with $w(\boldsymbol{x}, \boldsymbol{z}) = \delta(\boldsymbol{x} - \boldsymbol{z})$, where δ is a Dirac delta, the interdomain GP recovers FITC.

the GP learned in the original high-dimensional space can achieve the optimal rate when f is not highly smooth. This motivates the usage of Bayesian model averaging based on random compression over various configurations in order to reduce computational demands [188].

Continual theoretical and empirical efforts are required for designing specific components, e.g., the convolutional kernel [171], for scalable manifold GPs, because of the urgent demands in various fields, e.g., computer vision (CV).

B. Scalable Deep GP

Motivated by the enormous success of deep learning in various fields, the scalable deep GPs (DGPs) [34], [189] have been investigated in recent years.[20] A simple representative is to combine the structural NNs and the flexible nonparametric GP together, in which NNs map the original input space to the feature space for extracting nonstationary/recurrent features, and the last-layer sparse GP conducts standard regression over the latent space [34], [99], [186], [192]. The parameters of NNs and GP are jointly learned by maximizing the marginal likelihood in order to guard against overfitting. The NNs+GP structure produces sensible uncertainties and is found to be robust to adversarial examples in CV tasks [193]. Differently, Cremanns and Roos [194] employed the NNs to learn the input-dependent hyperparameters for the additive kernels. Then, the NeNe algorithm is employed to ease the GP inference. Besides, Iwata and Ghahramani [195] used the outputs of NNs as prior mean for SVGP.

More elegantly, inspired by deep learning, the DGP [189] and its variants [196]–[200], which employ the hierarchical and functional composite

$$y(\boldsymbol{x}) = f_l(f_{l-1}(\cdots f_1(\boldsymbol{x}))) + \epsilon \tag{32}$$

to stack multiple layers of latent variable model (LVM)[21] [11], [201] for extracting features. The DGP showcases great flexibility in (un)supervised scenarios, resulting in, however, a nonstandard GP. The recently developed convolutional kernel [171] opens up the way of DGP for CV tasks [202]. Interestingly, the finite layer-to-layer transformation of inputs in (32) can be extended to infinitely deep, but infinitesimal, differential fields governed by stochastic differential equations [203].

Note that the inference in DGP is intractable and expensive. Thus, efficient training requires a sophisticated approximate inference via inducing points [198], [204], 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 Multitask GP

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

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

where $\boldsymbol{\epsilon} = [\epsilon_1, \ldots, \epsilon_T]^{\mathsf{T}}$ is the individual noises. The crucial in MTGP is the construction of a valid multitask kernel $\boldsymbol{k}_{\text{MTGP}}(\boldsymbol{x}, \boldsymbol{x}') \in R^{T \times T}$, which can be built through, for example, the linear model of coregionalization [205] and the convolution process [170]. 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 $\mathcal{O}(T^3n^3)$. Hence, since the inference in most MTGPs follows the standard process, the above-reviewed sparse approximations and local approximations have been applied to MTGPs [170], [206]–[208] 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 multitask (multioutput) learning [209].

D. Scalable Online GP

Typically, it is assumed that the entire data $\mathcal{D}$ are 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 have real-time adaptation to the streaming data and handle large-scale case since the new data are continuously arriving.

Sparse GPs are extensible for online learning since they employ a small inducing set to summarize the whole training data [210]–[212]. As a result, the arrived new data only interact with the inducing points to enhance fast online learning. This is reasonable since the updates of $\mu(\boldsymbol{x}_*)$ and $\sigma^2(\boldsymbol{x}_*)$ of FITC and PITC only rely on the inducing set and new data [213], [214]. Moreover, the stochastic variants naturally showcase the online structure [215], since the bound in (17) supports minibatch learning by stochastic optimization.

However, there are two issues for scalable online GPs. First, some of them [211], [213], [214] fix the hyperparameters to obtain constant complexity per update. It is empirically argued that the optimization improves the model significantly in the first few iterations [215]. Hence, with the advanced computing power and the demand for accurate predictions, it could update the hyperparameters online over a small number of iterations.

Second, the scalable online GPs implicitly assume that the new data and the 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 process [216]. To address the evolving online learning, Nguyen *et al.* [217] 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] and [218], Bui *et al.* [216] 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 model were also provided in [219].

[20]GP has been pointed out as a shallow but infinitely wide NN with Gaussian weights [190], [191].

[21]The unsupervised GPLVM model is equivalent to multiple independent GPs with unobserved latent inputs.

E. Scalable Recurrent GP

There exist various tasks, e.g., speech recognition, system identification, energy forecasting, and robotics, in which the data sets are sequential and the ordering matters. Here, we focus on recurrent GP [220], [221] to handle sequential data.

The popular recurrent GP is GP-based nonlinear autoregressive models with exogenous inputs (GP-NARX) [220]. It is generally expressed as

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

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 $\boldsymbol{x}_t$, f is the emission function, and ϵ_{y_t} accounts for the observation noise. Note that the observations $y_{t-1},\ldots,y_{t-L_y}$ here are considered to be deterministic. The transformed input $\boldsymbol{x}_t$ comprising the 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 [222], local modeling by incorporating prior information [223], and higher order frequency response functions [224]. A main drawback of GP-NARX, however, is that it cannot account for the observation noise in $\boldsymbol{x}_t$, leading to the errors-in-variables problem. To address this issue, we could conduct data preprocessing to remove the noise from data [225], adopt GPs considering input noise [204], and employ the more powerful state-space models (SSMs) [221] introduced next.

The GP-SSM employs a more general recurrent structure as

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

where $\boldsymbol{x}_t$ is the state of the system that 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 considers the transition noise and brings the flexibility in requiring no lagged parameters. However, this model suffers from intractable inference since we need to marginalize out all the latent variables, thus requiring approximate inference [221], [226], [227].

Finally, we again see the trend in combining recurrent GPs with NNs. For instance, the deep recurrent GP [228] attempts to mimic the well-known recurrent NNs (RNNs), with each layer modeled by a GP. Similar to DGP [189], the inference in deep recurrent GP is intractable and requires sophisticated approximations, such as [229]. Hence, to keep the model as simple as possible while retaining the recurrent capability, the long short-term memory (LSTM) model [230] is directly combined with scalable GPs, resulting in analytical inference and desirable results [192].

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 [231] with $y\in\{0,1\}$ is usually formulated as

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

where $\pi(.)\in[0,1]$ is an inverse link function[22] that squashes f into the class probability space. Differently, the multiclass GPC (MGPC) [232] with $y\in\{1,\cdots,C\}$ is

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

where $\{f^c\}_{c=1}^C$ are independent latent functions[23] for C classes, and $\boldsymbol{f}=[f^1,\cdots,f^C]^\mathsf{T}:R^d\mapsto R^C$.[24] 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 $q(f|y)$ [231].

Motivated by the success of scalable GPR, we could directly treat GPC as a regression task [235] or solve it as GPR by a transformation that interprets class labels as the outputs of a Dirichlet distribution [236]. 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 [86], [237]–[241].[25]

The main challenges of scalable GPC, especially MGPC, are: 1) the intractable inference and posterior and 2) the high training complexity for large C. For the first issue, the stochastic GPC derives the model evidence expressed as the integration over a 1-D Gaussian distribution, which can be adequately calculated using the Gaussian–Hermite quadrature [75], [238]. Furthermore, the GPC equipped with the FITC assumption owns a completely analytical model evidence [239], [240]. Particularly, when taking the logit/softmax inverse link function, the Pòlya–Gamma data augmentation [242] offers analytical inference and posterior for GPC [241], [243]. A recent work [244] presents a noisy framework to derive analytical ELBOs for scalable binary/MGPCs using conventional likelihoods. 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 such as [234], thus allowing efficient stochastic training.

VII. Conclusion

Although the GP itself has a long history, the nonparametric flexibility and the high interpretability make it popular yet posing many challenges in the era of big data. In this article, we have attempted to summarize the state of scalable GPs in order to well understand the state of the art and 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
Libraries and Data Sets

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

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

[23] In contrast, inspired by the idea of MTGP, the correlations among latent functions have been exploited in [233].

[24] Similar 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 could recover the conventional inverse link functions, e.g., the probit and logit functions, through the integration of ϵ [234].

[25] Different from sparse GPR, the inducing points optimized in sparse GPC are usually pushed toward decision boundary [238].

TABLE I
LIST OF PRIMARY LIBRARIES SUPPORTING REPRESENTATIVE SCALABLE GPS

Package	Language	Global	Local	Others	GPU supported
GPML	Matlab	FITC [67], VFE [17], SPEP [76], SKI [18]	-	-	-
GPy	Python	VFE [17], SPEP [76], SKI [18], SVGP [1]	-	-	✓
GPstuff	Matlab&R	SoR [56], DTC [65], FITC [67], VFE [17], SVGP [1], CS [15]	-	PIC [41], CS+FIC [164]	-
GPflow	Python	VFE [17], SVGP [1]	-	NNs+SVGP [193]	✓
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 [144], DDM [245]	PIC [41]	-

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

Dataset	No. of inputs	No. of observations
terrain [40]	2	40.00K
aimpeak [160]	5	41.85K
sarcos [5]	21	48.93K
natural sound [18]	1	59.31K
chem [14]	15	63.07K
kuka [246]	28	197.92K
crime [247]	2	233.09K
sdss [176]	10	300.00K
precipitation [102]	3	628.47K
mujoco [82]	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.[26] 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. They can be found in the relevant researchers' webpage.

Besides, except the well-known UCI[27] regression data sets with $n \in [15, 4.18 \times 10^6]$ and $d \in [3, 4.8 \times 10^5]$, and the LIBSVM[28] regression data sets with $n \in [152, 2.06 \times 10^4]$ and $d \in [6, 4.27 \times 10^6]$, Table II summarizes the regression data sets ($n \geq 10^4$) occurred in the scalable GP literature. It is found that researchers have assessed scalable GPs with up to about one billion training points [36].

REFERENCES

[1] J. Hensman, N. Fusi, and N. D. Lawrence, "Gaussian processes for big data," in *Proc. 29th Conf. Uncertainty Artif. Intell.*, 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," *Inf. Sci.*, vol. 285, pp. 112–137, Nov. 2014.

[3] Y. LeCun, Y. Bengio, and G. Hinton, "Deep learning," *Nature*, vol. 521, no. 7553, p. 436, 2015.

[4] D. Silver *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. I. Williams, *Gaussian Processes for Machine Learning*. Cambridge, MA, USA: MIT Press, 2006.

[6] G. Matheron, "Principles of geostatistics," *Econ. Geol.*, vol. 58, no. 8, pp. 1246–1266, 1963.

[26]The 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) GPyTorch (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/); and 10) GPLP (http://www.jmlr.org/mloss/).

[27]https://archive.ics.uci.edu/ml/index.php

[28]https://www.csie.ntu.edu.tw/∼cjlin/libsvm/

[7] J. Sacks, W. J. Welch, T. J. Mitchell, and H. P. Wynn, "Design and analysis of computer experiments," *Stat. Sci.*, 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," *Struct. Multidisciplinary Optim.*, vol. 57, no. 1, pp. 393–416, 2018.

[9] M. A. Alvarez, L. Rosasco, and N. D. Lawrence, "Kernels for vector-valued functions: A review," *Found. Trends Mach. Learn.*, vol. 4, no. 3, pp. 195–266, 2012.

[10] H. Liu, J. Cai, and Y.-S. Ong, "Remarks on multi-output Gaussian process regression," *Knowl.-Based Syst.*, vol. 144, pp. 102–121, Mar. 2018.

[11] N. Lawrence, "Probabilistic non-linear principal component analysis with Gaussian process latent variable models," *J. Mach. Learn. Res.*, vol. 6, pp. 1783–1816, Nov. 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," *Proc. IEEE*, vol. 104, no. 1, pp. 148–175, Jan. 2016.

[13] S. Yin and O. Kaynak, "Big data for modern industry: Challenges and trends," *Proc. IEEE*, vol. 103, no. 2, pp. 143–146, Feb. 2015.

[14] K. Chalupka, C. K. I. Williams, and I. Murray, "A framework for evaluating approximation methods for Gaussian process regression," *J. Mach. Learn. Res.*, vol. 14, pp. 333–350, Feb. 2013.

[15] T. Gneiting, "Compactly supported correlation functions," *J. Multivariate Anal.*, vol. 83, no. 2, pp. 493–508, 2002.

[16] J. Quiñonero-Candela and C. E. Rasmussen, "A unifying view of sparse approximate Gaussian process regression," *J. Mach. Learn. Res.*, vol. 6, pp. 1939–1959, Dec. 2005.

[17] M. Titsias, "Variational learning of inducing variables in sparse Gaussian processes," in *Proc. Artif. Intell. Statist.*, 2009, pp. 567–574.

[18] A. G. Wilson and H. Nickisch, "Kernel interpolation for scalable structured Gaussian processes (KISS-GP)," in *Proc. Int. Conf. Mach. Learn.*, 2015, pp. 1775–1784.

[19] R. B. Gramacy and H. K. H. Lee, "Bayesian treed Gaussian process models with an application to computer modeling," *J. Amer. Stat. Assoc.*, vol. 103, no. 483, pp. 1119–1130, 2008.

[20] R. B. Gramacy *et al.*, "LaGP: Large-scale spatial modeling via local approximate Gaussian processes in R," *J. Stat. Softw.*, vol. 72, no. 1, pp. 1–46, 2016.

[21] S. E. Yuksel, J. N. Wilson, and P. D. Gader, "Twenty years of mixture of experts," *IEEE Trans. Neural Netw. Learn. Syst.*, vol. 23, no. 8, pp. 1177–1193, Aug. 2012.

[22] S. Masoudnia and R. Ebrahimpour, "Mixture of experts: A literature survey," *Artif. Intell. Rev.*, vol. 42, no. 2, pp. 275–293, 2014.

[23] C. E. Rasmussen and Z. Ghahramani, "Infinite mixtures of Gaussian process experts," in *Proc. Adv. Neural Inf. Process. Syst.*, 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 Trans. Intell. Transp. Syst.*, vol. 12, no. 2, pp. 466–475, Jun. 2011.

[25] G. E. Hinton, "Training products of experts by minimizing contrastive divergence," *Neural Comput.*, vol. 14, no. 8, pp. 1771–1800, 2002.

[26] M. P. Deisenroth and J. W. Ng, "Distributed Gaussian processes," in *Proc. 32nd Int. Conf. Int. Conf. Mach. Learn.*, 2015, pp. 1481–1490.

[27] D. Rullière, N. Durrande, F. Bachoc, and C. Chevalier, "Nested Kriging predictions for datasets with a large number of observations," *Statist. Comput.*, 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 *Proc. Int. Conf. Mach. Learn.*, 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 *Proc. Adv. Neural Inf. Process. Syst.* Red Hook, NY, USA: Curran, 2014, pp. 3257–3265.

[30] Z. Dai, A. Damianou, J. Hensman, and N. Lawrence, "Gaussian process models with parallelization and GPU acceleration," 2014, *arXiv:1410.4984*. [Online]. Available: https://arxiv.org/abs/1410.4984

[31] D. G. Matthews *et al.*, "GPflow: A Gaussian process library using TensorFlow," *J. Mach. Learn. Res.*, vol. 18, no. 1, pp. 1299–1304, 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 J. 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 *Proc. Artif. Intell. Statist.*, 2016, pp. 370–378.

[35] T. N. Hoang, Q. M. Hoang, and K. H. Low, "A unifying framework of anytime sparse Gaussian process regression models with stochastic variational inference for big data," in *Proc. Int. Conf. Mach. Learn.*, 2015, pp. 569–578.

[36] H. Peng, S. Zhe, X. Zhang, and Y. Qi, "Asynchronous distributed variational Gaussian process for regression," in *Proc. Int. Conf. Mach. Learn.*, 2017, pp. 2788–2797.

[37] R. Rivera and E. Burnaev, "Forecasting of commercial sales with large scale Gaussian processes," 2017, *arXiv:1709.05548*. [Online]. Available: https://arxiv.org/abs/1709.05548

[38] H. Liu, J. Cai, Y.-S. Ong, and Y. Wang, "Understanding and comparing scalable Gaussian process regression for big data," *Knowl.-Based Syst.*, vol. 164, pp. 324–335, Jan. 2019.

[39] M. Lázaro-Gredilla and A. Figueiras-Vidal, "Inter-domain Gaussian processes for sparse inference using inducing features," in *Proc. Adv. Neural Inf. Process. Syst.*, 2009, pp. 1087–1095.

[40] B.-J. Lee, J. Lee, and K.-E. Kim, "Hierarchically-partitioned Gaussian process approximation," in *Proc. Artif. Intell. Statist.*, 2017, pp. 822–831.

[41] E. Snelson and Z. Ghahramani, "Local and global sparse Gaussian process approximations," in *Proc. Artif. Intell. Statist.*, 2007, pp. 524–531.

[42] T. Nguyen and E. Bonilla, "Fast allocation of Gaussian process experts," in *Proc. Int. Conf. Mach. Learn.*, 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 *Proc. Int. Conf. Mach. Learn.*, 2016, pp. 382–391.

[44] G. Camps-Valls, J. Verrelst, J. Muñoz-Marí, V. Laparra, F. Mateo-Jiménez, and J. Gomez-Dans, "A survey on Gaussian processes for earth-observation data analysis: A comprehensive investigation," *IEEE Geosci. Remote Sens. Mag.*, vol. 4, no. 2, pp. 58–78, Jun. 2016.

[45] K. Hayashi, M. Imaizumi, and Y. Yoshida, "On random subsampling of Gaussian process regression: A graphon-based analysis," 2019, *arXiv:1901.09541*. [Online]. Available: https://arxiv.org/abs/1901.09541

[46] F. P. Preparata and M. I. Shamos, *Computational Geometry: An Introduction*. Berlin, Germany: Springer-Verlag, 2012.

[47] R. Herbrich, N. D. Lawrence, and M. Seeger, "Fast sparse Gaussian process methods: The informative vector machine," in *Proc. Adv. Neural Inf. Process. Syst.*, 2003, pp. 625–632.

[48] M. Seeger and C. Williams, "Bayesian Gaussian process models: PAC-Bayesian generalisation error bounds and sparse approximations," Ph.D. dissertation, School Comput. Commun. Sci., Univ. Edinburgh, Edinburgh, Scotland, 2003.

[49] S. Keerthi and W. Chu, "A matching pursuit approach to sparse Gaussian process regression," in *Proc. Adv. Neural Inf. Process. Syst.*, 2006, pp. 643–650.

[50] A. Melkumyan and F. T. Ramos, "A sparse covariance function for exact Gaussian process inference in large datasets," in *Proc. Int. Joint Conf. Artif. Intell.*, vol. 9, 2009, pp. 1936–1942.

[51] M. Buhmann, "A new class of radial basis functions with compact support," *Math. Comput.*, vol. 70, no. 233, pp. 307–318, 2001.

[52] H. Wendland, *Scattered Data Approximation*. Cambridge, U.K.: Cambridge Univ. Press, 2004.

[53] A. Gittens and M. W. Mahoney, "Revisiting the nyström method for improved large-scale machine learning," *J. Mach. Learn. Res.*, vol. 17, no. 1, pp. 3977–4041, 2016.

[54] C. K. Williams and M. Seeger, "Using the Nyström method to speed up kernel machines," in *Proc. Adv. Neural Inf. Process. Syst.*, 2001, pp. 682–688.

[55] C. Williams, C. Rasmussen, A. Scwaighofer, and V. Tresp, "Observations on the nyström method for Gaussian process prediction," Division Inform., Univ. Edinburgh, Edinburgh, Scotland, Tech. Rep., 2002.

[56] A. J. Smola and P. L. Bartlett, "Sparse greedy Gaussian process regression," in *Proc. Adv. Neural Inf. Process. Syst.*, 2001, pp. 619–625.

[57] M. E. Tipping, "Sparse Bayesian learning and the relevance vector machine," *J. Mach. Learn. Res.*, vol. 1, pp. 211–244, Sep. 2001.

[58] H. Peng and Y. Qi, "EigenGP: Gaussian process models with adaptive eigenfunctions," in *Proc. Int. Joint Conf. Artif. Intell.*, 2015, pp. 3763–3769.

[59] N. Cressie and G. Johannesson, "Fixed rank Kriging for very large spatial data sets," *J. Roy. Statist. Soc. B (Statist. Methodol.)*, vol. 70, no. 1, pp. 209–226, 2007.

[60] C. E. Rasmussen and J. Quiñonero-Candela, "Healing the relevance vector machine through augmentation," in *Proc. Int. Conf. Mach. Learn.*, 2005, pp. 689–696.

[61] M. Lázaro-Gredilla, J. Quiñonero-Candela, C. E. Rasmussen, and A. R. Figueiras-Vidal, "Sparse spectrum Gaussian process regression," *J. Mach. Learn. Res.*, vol. 11, pp. 1865–1881, Jun. 2010.

[62] Y. Gal and R. Turner, "Improving the Gaussian process sparse spectrum approximation by representing uncertainty in frequency inputs," in *Proc. Int. Conf. Mach. Learn.*, 2015, pp. 655–664.

[63] L. S. L. Tan, V. M. H. Ong, D. J. Nott, and A. Jasra, "Variational inference for sparse spectrum Gaussian process regression," *Statist. Comput.*, 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 *Proc. AAAI Conf. Artif. Intell.*, 2017, pp. 2007–2014.

[65] L. Csató and M. Opper, "Sparse on-line Gaussian processes," *Neural Comput.*, 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 *Proc. Artif. Intell. Statist.*, 2003, Art. no. 161318.

[67] E. Snelson and Z. Ghahramani, "Sparse Gaussian processes using pseudo-inputs," in *Proc. Adv. Neural Inf. Process. Syst.*, 2006, pp. 1257–1264.

[68] E. L. Snelson, "Flexible and efficient Gaussian process models for machine learning," Ph.D. dissertation, Gatsby Comput. Neurosci. Unit, Univ. College London, London, U.K., 2008.

[69] M. Bauer, M. van der Wilk, and C. E. Rasmussen, "Understanding probabilistic sparse Gaussian process approximations," in *Proc. Adv. Neural Inf. Process. Syst.*, 2016, pp. 1533–1541.

[70] D. M. Blei, A. Kucukelbir, and J. D. McAuliffe, "Variational inference: A review for statisticians," *J. Amer. Stat. Assoc.*, vol. 112, no. 518, pp. 859–877, 2017.

[71] A. G. D. G. Matthews, J. Hensman, R. E. Turner, and Z. Ghahramani, "On sparse variational methods and the Kullback–Leibler divergence between stochastic processes," *J. Mach. Learn. Res.*, vol. 51, pp. 231–239, May 2016.

[72] M. K. Titsias, "Variational model selection for sparse Gaussian process regression," School Comput. Sci., Univ. Manchester, Manchester, U.K., Tech. Rep., 2009.

[73] Y. Cao, M. A. Brubaker, D. J. Fleet, and A. Hertzmann, "Efficient optimization for sparse Gaussian process regression," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 37, no. 12, pp. 2415–2427, Dec. 2015.

[74] S. Zhe, "Regularized variational sparse Gaussian processes," in *Proc. NIPS Workshop Approx. Inference*, 2017, pp. 1–5.

[75] A. G. de Garis Matthews, "Scalable Gaussian process inference using variational methods," Ph.D. dissertation, Dept. Eng., Univ. Cambridge, Cambridge, U.K., 2017.

[76] T. D. Bui, J. Yan, and R. E. Turner, "A unifying framework for Gaussian process pseudo-point approximations using power expectation propagation," *J. Mach. Learn. Res.*, vol. 18, no. 1, pp. 3649–3720, 2017.

[77] D. P. Kingma and J. Ba, "Adam: A method for stochastic optimization," 2014, *arXiv:1412.6980*. [Online]. Available: https://arxiv.org/abs/1412.6980

[78] M. D. Hoffman, D. M. Blei, C. Wang, and J. Paisley, "Stochastic variational inference," *J. Mach. Learn. Res.*, vol. 14, pp. 1303–1347, 2013.

[79] H. Salimbeni, S. Eleftheriadis, and J. Hensman, "Natural gradients in practice: Non-conjugate variational inference in Gaussian process models," in *Proc. Artif. Intell. Statist.*, 2018, pp. 689–697.

[80] M. Li, D. G. Andersen, and A. Smola, "Distributed delayed proximal gradient methods," in *Proc. NIPS Workshop Optim. Mach. Learn.*, vol. 3, 2013, pp. 1–5.

[81] M. Li, D. G. Andersen, A. J. Smola, and K. Yu, "Communication efficient distributed machine learning with the parameter server," in *Proc. Adv. Neural Inf. Process. Syst.*, 2014, pp. 19–27.

[82] C.-A. Cheng and B. Boots, "Variational inference for Gaussian process models with linear complexity," in *Proc. Adv. Neural Inf. Process. Syst.*, 2017, pp. 5184–5194.

[83] T. Nickson, T. Gunter, C. Lloyd, M. A. Osborne, and S. Roberts, "Blitzkriging: Kronecker-structured stochastic Gaussian processes," 2015, *arXiv:1510.07965*. [Online]. Available: https://arxiv.org/abs/1510.07965

[84] P. Izmailov, A. Novikov, and D. Kropotov, "Scalable Gaussian processes with billions of inducing inputs via tensor train decomposition," 2017, *arXiv:1710.07324*. [Online]. Available: https://arxiv.org/abs/1710.07324

[85] M. T. R. C. Aueb and M. Lázaro-Gredilla, "Variational inference for Mahalanobis distance metrics in Gaussian process regression," in *Proc. Adv. Neural Inf. Process. Syst.*, 2013, pp. 279–287.

[86] J. Hensman, A. G. Matthews, M. Filippone, and Z. Ghahramani, "MCMC for variationally sparse Gaussian processes," in *Proc. Adv. Neural Inf. Process. Syst.*, 2015, pp. 1648–1656.

[87] H. Yu, T. N. Hoang, K. H. Low, and P. Jaillet, "Stochastic variational inference for Bayesian sparse Gaussian process regression," 2017, *arXiv:1711.00221*. [Online]. Available: https://arxiv.org/abs/1711.00221

[88] R. Sheth, Y. Wang, and R. Khardon, "Sparse variational inference for generalized GP models," in *Proc. Int. Conf. Mach. Learn.*, 2015, pp. 1302–1311.

[89] J. Hensman, N. Durrande, and A. Solin, "Variational Fourier features for Gaussian processes," 2016, *arXiv:1611.06740*. [Online]. Available: https://arxiv.org/abs/1611.06740

[90] Y. Shen, A. Y. Ng, and M. Seeger, "Fast Gaussian process regression using KD-trees," in *Proc. Adv. Neural Inf. Process. Syst.*, 2006, pp. 1225–1232.

[91] V. I. Morariu, B. V. Srinivasan, V. C. Raykar, R. Duraiswami, and L. S. Davis, "Automatic online tuning for fast Gaussian summation," in *Proc. Adv. Neural Inf. Process. Syst.*, 2009, pp. 1113–1120.

[92] K. Cutajar, M. Osborne, J. Cunningham, and M. Filippone, "Preconditioning kernel matrices," in *Proc. Int. Conf. Mach. Learn.*, 2016, pp. 2529–2538.

[93] Y. Saatçi, "Scalable inference for structured Gaussian process models," Ph.D. dissertation, St. Edmund's College, Univ. Cambridge, Cambridge, U.K., 2011.

[94] E. Gilboa, Y. Saatçi, and J. P. Cunningham, "Scaling multidimensional inference for structured Gaussian processes," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 37, no. 2, pp. 424–436, Feb. 2015.

[95] J. P. Cunningham, K. V. Shenoy, and M. Sahani, "Fast Gaussian process methods for point process intensity estimation," in *Proc. Int. Conf. Mach. Learn.*, 2008, pp. 192–199.

[96] A. G. Wilson, E. Gilboa, A. Nehorai, and J. P. Cunningham, "Fast kernel learning for multidimensional pattern extrapolation," in *Proc. Adv. Neural Inf. Process. Syst.*, 2014, pp. 3626–3634.

[97] G. Pleiss, J. R. Gardner, K. Q. Weinberger, and A. G. Wilson, "Constant-time predictive distributions for Gaussian processes," in *Proc. Int. Conf. Mach. Learn.*, 2018, pp. 4111–4120.

[98] A. G. Wilson, C. Dann, and H. Nickisch, "Thoughts on massively scalable Gaussian processes," 2015, *arXiv:1511.01870*. [Online]. Available: https://arxiv.org/abs/1511.01870

[99] A. G. Wilson, Z. Hu, R. R. Salakhutdinov, and E. P. Xing, "Stochastic variational deep kernel learning," in *Proc. Adv. Neural Inf. Process. Syst.*, 2016, pp. 2586–2594.

[100] T. W. Evans and P. B. Nair, "Scalable Gaussian processes with grid-structured eigenfunctions (GP-GRIEF)," in *Proc. NIPS Workshop Adv. Approx. Bayesian Inference*, 2017, pp. 1416–1425.

[101] J. R. Gardner, G. Pleiss, R. Wu, K. Q. Weinberger, and A. G. Wilson, "Product kernel interpolation for scalable Gaussian processes," 2018, *arXiv:1802.08903*. [Online]. Available: https://arxiv.org/abs/1802.08903

[102] K. Dong, D. Eriksson, H. Nickisch, D. Bindel, and A. G. Wilson, "Scalable log determinants for Gaussian process kernel learning," in *Proc. Adv. Neural Inf. Process. Syst.*, 2017, pp. 6327–6337.

[103] D. R. Burt, C. E. Rasmussen, and M. van der Wilk, "Rates of convergence for sparse variational Gaussian process regression," in *Proc. Int. Conf. Mach. Learn.*, 2019, pp. 862–871.

[104] L. Csató and M. Opper, "Sparse representation for Gaussian process models," in *Proc. Adv. Neural Inf. Process. Syst.*, 2001, pp. 444–450.

[105] J. Schreiter, D. Nguyen-Tuong, and M. Toussaint, "Efficient sparsification for Gaussian process regression," *Neurocomputing*, vol. 192, pp. 29–37, Jun. 2016.

[106] A. Pourhabib, F. Liang, and Y. Ding, "Bayesian site selection for fast Gaussian process regression," *IIE Trans.*, vol. 46, no. 5, pp. 543–555, 2014.

[107] H.-M. Kim, B. K. Mallick, and C. C. Holmes, "Analyzing nonstationary spatial data using piecewise Gaussian processes," *J. Amer. Stat. Assoc.*, 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 Interdiscipl. Rev., Comput. Statist.*, 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," *J. Field Robot.*, 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," *J. Comput. Graph. Statist.*, vol. 23, no. 3, pp. 830–852, 2014.

[111] D. G. T. Denison, N. M. Adams, C. C. Holmes, and D. J. Hand, "Bayesian partition modelling," *Comput. Statist. Data Anal.*, vol. 38, no. 4, pp. 475–485, 2002.

[112] R. Urtasun and T. Darrell, "Sparse probabilistic regression for activity-independent human pose inference," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, Jun. 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," *J. Amer. Stat. Assoc.*, vol. 111, no. 514, pp. 800–812, 2016.

[114] X. Emery, "The Kriging update equations and their application to the selection of neighboring data," *Comput. Geosci.*, vol. 13, no. 3, pp. 269–280, 2009.

[115] R. B. Gramacy and H. K. H. 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," *J. Comput. Graph. Statist.*, vol. 24, no. 2, pp. 561–578, 2015.

[117] C. Park and J. Huang, "Efficient computation of Gaussian process regression for large spatial data sets by patching local Gaussian processes," *J. Mach. Learn. Res.*, vol. 17, no. 174, pp. 1–29, 2016.

[118] C. Park and D. Apley, "Patchwork Kriging for large-scale Gaussian process regression," 2017, *arXiv:1701.06655*. [Online]. Available: https://arxiv.org/abs/1701.06655

[119] J. Mendes-Moreira, C. Soares, A. M. Jorge, and J. F. D. Sousa, "Ensemble approaches for regression: A survey," *ACM Comput. Surv.*, vol. 45, no. 1, p. 10, Nov. 2012.

[120] R. A. Jacobs, M. I. Jordan, S. J. Nowlan, and G. E. Hinton, "Adaptive mixtures of local experts," *Neural Comput.*, vol. 3, no. 1, pp. 79–87, 2012.

[121] J. Geweke and M. Keane, "Smoothly mixing regressions," *J. 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 *Proc. Adv. Neural Inf. Process. Syst.*, 1995, pp. 633–640.

[123] C. A. M. Lima, A. L. V. Coelho, and F. J. von Zuben, "Hybridizing mixtures of experts with support vector machines: Investigation into nonlinear dynamic systems identification," *Inf. Sci.*, 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 Netw.*, 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 Comput.*, 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," *Artif. Intell. Med.*, vol. 41, no. 1, pp. 57–67, 2007.

[127] C. M. Bishop and M. Svenskn, "Bayesian hierarchical mixtures of experts," in *Proc. 19th Conf. Uncertainty Artif. Intell.* San Mateo, CA, USA: Morgan Kaufmann, 2002, pp. 57–64.

[128] F. Chamroukhi, "Skew t mixture of experts," *Neurocomputing*, vol. 266, pp. 390–408, Nov. 2017.

[129] V. Tresp, "Mixtures of Gaussian processes," in *Proc. Adv. Neural Inf. Process. Syst.*, 2001, pp. 654–660.

[130] E. Meeds and S. Osindero, "An alternative infinite mixture of Gaussian process experts," in *Proc. Adv. Neural Inf. Process. Syst.*, 2006, pp. 883–890.

[131] Y. Yang and J. Ma, "An efficient EM approach to parameter learning of the mixture of Gaussian processes," in *Proc. Int. Symp. Neural Netw.*, 2011, pp. 165–174.

[132] Z. Chen, J. Ma, and Y. Zhou, "A precise hard-cut EM algorithm for mixtures of Gaussian processes," in *Proc. Int. Conf. Intell. Comput.*, 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 *Proc. IEEE Int. Conf. Acoust., Speech Signal Process.*, Mar. 2016, pp. 2579–2583.

[134] C. Yuan and C. Neubauer, "Variational mixture of Gaussian process experts," in *Proc. Adv. Neural Inf. Process. Syst.*, 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 Recognit.*, vol. 45, no. 4, pp. 1386–1395, 2012.

[136] J. Ross and J. Dy, "Nonparametric mixture of Gaussian processes with constraints," in *Proc. Int. Conf. Mach. Learn.*, 2013, pp. 1346–1354.

[137] J. V. Hansen, "Combining predictors: Meta machine learning methods and bias/variance & ambiguity decompositions," Ph.D. dissertation, Dept. Comput. Sci., Aarhus Univ., Aarhus, Denmark, 2000.

[138] D. Nguyen-Tuong, M. Seeger, and J. Peters, "Model learning with local Gaussian process regression," *Adv. Robot.*, 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 Trans. Vis. Comput. Graphics*, vol. 22, no. 11, pp. 2437–2450, Nov. 2016.

[140] M. Huang, R. Li, H. Wang, and W. Yao, "Estimating mixture of Gaussian processes by kernel smoothing," *J. Bus. Econ. Statist.*, 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 *Proc. Int. Symp. Neural Netw.*, 2015, pp. 345–354.

[142] S. P. Chatzis and Y. Demiris, "Nonparametric mixtures of Gaussian processes with power-law behavior," *IEEE Trans. Neural Netw. Learn. Syst.*, vol. 23, no. 12, pp. 1862–1871, Dec. 2012.

[143] Y. Cao and D. J. Fleet, "Generalized product of experts for automatic and principled fusion of Gaussian process predictions," 2014, *arXiv:1410.7827*. [Online]. Available: https://arxiv.org/abs/1410.7827

[144] T. Chen and J. Ren, "Bagging for Gaussian process regression," *Neurocomputing*, vol. 72, nos. 7–9, pp. 1605–1610, Mar. 2009.

[145] 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 *Proc. Int. Conf. Artif. Neural Netw.*, 2013, pp. 17–25.

[146] V. Tresp, "A Bayesian committee machine," *Neural Comput.*, vol. 12, no. 11, pp. 2719–2741, 2000.

[147] J. W. Ng and M. P. Deisenroth, "Hierarchical mixture-of-experts model for large-scale Gaussian process regression," 2014, *arXiv:1412.3078*. [Online]. Available: https://arxiv.org/abs/1412.3078

[148] S. Mair and U. Brefeld, "Distributed robust Gaussian process regression," *Knowl. Inf. Syst.*, vol. 55, no. 2, pp. 415–435, 2018.

[149] B. Szabo and H. van Zanten, "An asymptotic analysis of distributed nonparametric methods," 2017, *arXiv:1711.03149*. [Online]. Available: https://arxiv.org/abs/1711.03149

[150] Y.-L. K. Samo and S. J. Roberts, "String and membrane Gaussian processes," *J. Mach. Learn. Res.*, vol. 17, no. 1, pp. 4485–4571, 2016.

[151] K. H. Low, J. M. Dolan, and P. Khosla, "Active Markov information-theoretic path planning for robotic environmental sensing," in *Proc. 10th Int. Conf. Auton. Agents Multiagent Syst.*, 2011, pp. 753–760.

[152] M. Franey, P. Ranjan, and H. Chipman, "A short note on Gaussian process modeling for large datasets using graphics processing units," 2012, *arXiv:1203.1269*. [Online]. Available: https://arxiv.org/abs/1203.1269

[153] C. J. Paciorek, B. Lipshitz, W. Zhuo, C. G. G. Kaufman, and R. C. Thomas, "Parallelizing Gaussian process calculations in R," *J. Stat. Softw.*, vol. 63, no. i10, pp. 1–23, 2015.

[154] S. Ambikasaran, D. Foreman-Mackey, L. Greengard, D. W. Hogg, and M. O'Neil, "Fast direct methods for Gaussian processes," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 38, no. 2, pp. 252–265, Feb. 2016.

[155] M. Tavassolipour, S. A. Motahari, and M. T. M. Shalmani, "Learning of Gaussian processes in distributed and communication limited systems," 2017, *arXiv:1705.02627*. [Online]. Available: https://arxiv.org/abs/1705.02627

[156] 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," 2019, *arXiv:1903.08114*. [Online]. Available: https://arxiv.org/abs/1903.08114

[157] J. Dean and S. Ghemawat, "MapReduce: Simplified data processing on large clusters," *Commun. ACM*, vol. 51, no. 1, pp. 107–113, 2008.

[158] J. Gardner, G. Pleiss, K. Q. Weinberger, D. Bindel, and A. G. Wilson, "GPyTorch: Blackbox matrix-matrix Gaussian process inference with GPU acceleration," in *Proc. Adv. Neural Inf. Process. Syst.*, 2018, pp. 7576–7586.

[159] J. Chen *et al.*, "Decentralized data fusion and active sensing with mobile sensors for modeling and predicting spatiotemporal traffic phenomena," in *Proc. Uncertainty Artif. Intell.*, 2012, pp. 163–173.

[160] 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 *Proc. Uncertainty Artif. Intell.*, 2013, pp. 152–161.

[161] K. H. Low, J. Yu, J. Chen, and P. Jaillet, "Parallel Gaussian process regression for big data: Low-rank representation meets Markov approximation," in *Proc. AAAI Conf. Artif. Intell.*, 2015, pp. 2821–2827.

[162] Y. Ding, R. Kondor, and J. Eskreis-Winkler, "Multiresolution kernel approximation for Gaussian process regression," in *Proc. Adv. Neural Inf. Process. Syst.*, 2017, pp. 3740–3748.

[163] J. Vanhatalo, V. Pietiläinen, and A. Vehtari, "Approximate inference for disease mapping with sparse Gaussian processes," *Statist. Med.*, vol. 29, no. 15, pp. 1580–1607, 2010.

[164] J. Vanhatalo and A. Vehtari, "Modelling local and global phenomena with sparse Gaussian processes," in *Proc. 24th Conf. Uncertainty Artif. Intell.*, 2008, pp. 571–578.

[165] D. Gu and H. Hu, "Spatial Gaussian process regression with mobile sensor networks," *IEEE Trans. Neural Netw. Learn. Syst.*, vol. 23, no. 8, pp. 1279–1290, Aug. 2012.

[166] T. D. Bui and R. E. Turner, "Tree-structured Gaussian process approximations," in *Proc. Adv. Neural Inf. Process. Syst.*, 2014, pp. 2213–2221.

[167] S. Park and S. Choi, "Hierarchical Gaussian process regression," in *Proc. Asian Conf. Mach. Learn.*, 2010, pp. 95–110.

[168] E. Fox and D. B. Dunson, "Multiresolution Gaussian processes," in *Proc. Adv. Neural Inf. Process. Syst.*, 2012, pp. 737–745.

[169] D. Nychka, S. Bandyopadhyay, D. Hammerling, F. Lindgren, and S. Sain, "A multiresolution Gaussian process model for the analysis of large spatial datasets," *J. Comput. Graph. Statist.*, vol. 24, no. 2, pp. 579–599, 2015.

[170] M. A. Álvarez and N. D. Lawrence, "Computationally efficient convolved multiple output Gaussian processes," *J. Mach. Learn. Res.*, vol. 12, pp. 1459–1500, May 2011.

[171] M. van der Wilk, C. E. Rasmussen, and J. Hensman, "Convolutional Gaussian processes," in *Proc. Adv. Neural Inf. Process. Syst.*, 2017, pp. 2849–2858.

[172] F. Tobar, T. D. Bui, and R. E. Turner, "Learning stationary time series using Gaussian processes with nonparametric kernels," in *Proc. Adv. Neural Inf. Process. Syst.*, 2015, pp. 3501–3509.

[173] C. Walder, K. I. Kim, and B. Schölkopf, "Sparse multiscale Gaussian process regression," in *Proc. Int. Conf. Mach. Learn.*, 2008, pp. 1112–1119.

[174] Z. Zhang, K. Duraisamy, and N. A. Gumerov, "Efficient multiscale Gaussian process regression using hierarchical clustering," 2015, *arXiv:1511.02258*. [Online]. Available: https://arxiv.org/abs/1511.02258

[175] E. Snelson and Z. Ghahramani, "Variable noise and dimensionality reduction for sparse Gaussian processes," in *Proc. 22nd Conf. Uncertainty Artif. Intell.*, 2006, pp. 461–468.

[176] 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 Roy. Astron. Soc.*, vol. 462, no. 1, pp. 726–739, 2016.

[177] P. W. Goldberg, C. K. I. Williams, and C. M. Bishop, "Regression with input-dependent noise: A Gaussian process treatment," in *Proc. Adv. Neural Inf. Process. Syst.*, 1998, pp. 493–499.

[178] M. Lázaro-Gredilla and M. K. Titsias, "Variational heteroscedastic Gaussian process regression," in *Proc. 28th Int. Conf. Int. Conf. Mach. Learn.* Madison, WI, USA: Omnipress, 2011, pp. 841–848.

[179] H. Liu, Y.-S. Ong, and J. Cai, "Large-scale heteroscedastic regression via Gaussian process," 2018, *arXiv:1811.01179*. [Online]. Available: https://arxiv.org/abs/1811.01179

[180] C. Wang and R. M. Neal, "Gaussian process regression with heteroscedastic or non-Gaussian residuals," 2012, *arXiv:1212.6246*. [Online]. Available: https://arxiv.org/abs/1212.6246

[181] H. Salimbeni, V. Dutordoir, J. Hensman, and M. Deisenroth, "Deep Gaussian processes with importance-weighted variational inference," in *Proc. 36th Int. Conf. Mach. Learn.*, 2019, pp. 5589–5598.

[182] V. Dutordoir, H. Salimbeni, J. Hensman, and M. Deisenroth, "Gaussian process conditional density estimation," in *Proc. Adv. Neural Inf. Process. Syst.*, 2018, pp. 2385–2395.

[183] Y. Yang and D. B. Dunson, "Bayesian manifold regression," *Ann. Statist.*, vol. 44, no. 2, pp. 876–905, 2016.

[184] 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 Trans. Signal Process.*, vol. 58, no. 12, pp. 6140–6155, Dec. 2010.

[185] A. C. Damianou, M. K. Titsias, and N. D. Lawrence, "Variational inference for uncertainty on the inputs of Gaussian process models," 2014, *arXiv:1409.2287*. [Online]. Available: https://arxiv.org/abs/1409.2287

[186] R. Calandra, J. Peters, C. E. Rasmussen, and M. P. Deisenroth, "Manifold Gaussian processes for regression," in *Proc. Int. Joint Conf. Neural Netw.*, Jul. 2016, pp. 3338–3345.

[187] 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.

[188] R. Guhaniyogi and D. B. Dunson, "Compressed Gaussian process for manifold regression," *J. Mach. Learn. Res.*, vol. 17, no. 1, pp. 2472–2497, 2016.

[189] A. Damianou and N. Lawrence, "Deep Gaussian processes," in *Proc. Artif. Intell. Statist.*, 2013, pp. 207–215.

[190] R. M. Neal, *Bayesian Learning for Neural Networks*, vol. 118. Berlin, Germany: Springer-Verlag, 1996.

[191] A. G. D. G. Matthews, M. Rowland, J. Hron, R. E. Turner, and Z. Ghahramani, "Gaussian process behaviour in wide deep neural networks," 2018, *arXiv:1804.11271*. [Online]. Available: https://arxiv.org/abs/1804.11271

[192] M. Al-Shedivat, A. G. Wilson, Y. Saatchi, Z. Hu, and E. P. Xing, "Learning scalable deep kernels with recurrent structure," *J. Mach. Learn. Res.*, vol. 18, no. 1, pp. 2850–2886, 2017.

[193] J. Bradshaw, A. G. D. G. Matthews, and Z. Ghahramani, "Adversarial examples, uncertainty, and transfer testing robustness in Gaussian process hybrid deep networks," 2017, *arXiv:1707.02476*. [Online]. Available: https://arxiv.org/abs/1707.02476

[194] K. Cremanns and D. Roos, "Deep Gaussian covariance network," 2017, *arXiv:1710.06202*. [Online]. Available: https://arxiv.org/abs/1710.06202

[195] T. Iwata and Z. Ghahramani, "Improving output uncertainty estimation and generalization in deep learning via neural network Gaussian processes," 2017, *arXiv:1707.05922*. [Online]. Available: https://arxiv.org/abs/1707.05922

[196] Z. Dai, A. Damianou, J. González, and N. Lawrence, "Variational auto-encoded deep Gaussian processes," 2015, *arXiv:1511.06455*. [Online]. Available: https://arxiv.org/abs/1511.06455

[197] T. Bui, D. Hernández-Lobato, J. Hernandez-Lobato, Y. Li, and R. Turner, "Deep Gaussian processes for regression using approximate expectation propagation," in *Proc. Int. Conf. Mach. Learn.*, 2016, pp. 1472–1481.

[198] H. Salimbeni and M. Deisenroth, "Doubly stochastic variational inference for deep Gaussian processes," in *Proc. Adv. Neural Inf. Process. Syst.*, 2017, pp. 4588–4599.

[199] K. Cutajar, E. V. Bonilla, P. Michiardi, and M. Filippone, "Random feature expansions for deep Gaussian processes," in *Proc. 34th Int. Conf. Mach. Learn.*, 2017, pp. 884–893.

[200] M. Havasi, J. M. Hernández-Lobato, and J. J. Murillo-Fuentes, "Inference in deep Gaussian processes using stochastic gradient Hamiltonian Monte Carlo," in *Proc. Adv. Neural Inf. Process. Syst.*, 2018, pp. 7506–7516.

[201] M. K. Titsias and N. D. Lawrence, "Bayesian Gaussian process latent variable model," in *Proc. 13th Int. Conf. Artif. Intell. Statist.*, 2010, pp. 844–851.

[202] V. Kumar, V. Singh, P. K. Srijith, and A. Damianou, "Deep Gaussian processes with convolutional kernels," 2018, *arXiv:1806.01655*. [Online]. Available: https://arxiv.org/abs/1806.01655

[203] P. Hegde, M. Heinonen, H. Lähdesmäki, and S. Kaski, "Deep learning with differential Gaussian process flows," in *Proc. Int. Conf. Artif. Intell. Statist.*, 2019, pp. 1812–1821.

[204] A. C. Damianou, M. K. Titsias, and N. D. Lawrence, "Variational inference for latent variables and uncertain inputs in Gaussian processes," *J. Mach. Learn. Res.*, vol. 17, no. 1, pp. 1425–1486, 2016.

[205] E. V. Bonilla, K. M. Chai, and C. K. I. Williams, "Multi-task Gaussian process prediction," in *Proc. Adv. Neural Inf. Process. Syst.*, 2008, pp. 153–160.

[206] T. V. Nguyen and E. V. Bonilla, "Collaborative multi-output Gaussian processes," in *Proc. 13th Conf. Uncertainty Artif. Intell.*, 2014, pp. 643–652.

[207] J. Zhao and S. Sun, "Variational dependent multi-output Gaussian process dynamical systems," *J. Mach. Learn. Res.*, vol. 17, no. 1, pp. 4134–4169, 2016.

[208] A. Chiplunkar, E. Rachelson, M. Colombo, and J. Morlier, "Approximate inference in related multi-output Gaussian process regression," in *Proc. Int. Conf. Pattern Recognit. Appl. Methods*, 2016, pp. 88–103.

[209] D. Xu, Y. Shi, I. W. Tsang, Y.-S. Ong, C. Gong, and X. Shen, "A survey on multi-output learning," 2019, *arXiv:1901.00248*. [Online]. Available: https://arxiv.org/abs/1901.00248

[210] D. Petelin and J. Kocijan, "Evolving Gaussian process models for predicting chaotic time-series," in *Proc. IEEE Conf. Evolving Adapt. Intell. Syst.*, Jun. 2014, pp. 1–8.

[211] M. F. Huber, "Recursive Gaussian process: On-line regression and learning," *Pattern Recognit. Lett.*, vol. 45, pp. 85–91, Aug. 2014.

[212] T. Le, K. Nguyen, V. Nguyen, T. D. Nguyen, and D. Phung, "GoGP: Fast online regression with Gaussian processes," in *Proc. IEEE Int. Conf. Data Mining*, Nov. 2017, pp. 257–266.

[213] 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.

[214] H. Bijl, T. B. Schön, J.-W. van Wingerden, and M. Verhaegen, "System identification through online sparse Gaussian process regression with input noise," *IFAC J. Syst. Control*, vol. 2, pp. 1–11, Dec. 2017.

[215] C.-A. Cheng and B. Boots, "Incremental variational sparse Gaussian process regression," in *Proc. Adv. Neural Inf. Process. Syst.*, 2016, pp. 4410–4418.

[216] T. D. Bui, C. Nguyen, and R. E. Turner, "Streaming sparse Gaussian process approximations," in *Proc. Adv. Neural Inf. Process. Syst.*, 2017, pp. 3299–3307.

[217] D. Nguyen-Tuong, J. R. Peters, and M. Seeger, "Local Gaussian process regression for real time online model learning," in *Proc. Adv. Neural Inf. Process. Syst.*, 2009, pp. 1193–1200.

[218] 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 *Proc. AAAI Conf. Artif. Intell.* Palo Alto, CA, USA: AAAI Press, 2014, pp. 2585–2592.

[219] 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 *Proc. ICML Time Ser. Workshop*, 2017, pp. 1–5.

[220] J. Kocijan, A. Girard, B. Banko, and R. Murray-Smith, "Dynamic systems identification with Gaussian processes," *Math. Comput. Model. Dyn. Syst.*, vol. 11, no. 4, pp. 411–424, 2005.

[221] R. Frigola, Y. Chen, and C. E. Rasmussen, "Variational Gaussian process state-space models," in *Proc. Adv. Neural Inf. Process. Syst.*, 2014, pp. 3680–3688.

[222] C. L. C. Mattos, J. D. A. Santos, and G. A. Barreto, "An empirical evaluation of robust Gaussian process models for system identification," in *Proc. Int. Conf. Intell. Data Eng. Automated Learn.*, 2015, pp. 172–180.

[223] K. Ažman and J. Kocijan, "Dynamical systems identification using Gaussian process models with incorporated local models," *Eng. Appl. Artif. Intell.*, vol. 24, no. 2, pp. 398–408, 2011.

[224] 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*. Cham, Switzerland: Springer, 2014, pp. 315–335.

[225] R. Frigola and C. E. Rasmussen, "Integrated pre-processing for Bayesian nonlinear system identification with Gaussian processes," in *Proc. 52nd IEEE Conf. Decis. Control*, Dec. 2013, pp. 5371–5376.

[226] 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 *Proc. Adv. Neural Inf. Process. Syst.*, 2013, pp. 3156–3164.

[227] A. Svensson, A. Solin, S. Särkkä, and T. Schön, "Computationally efficient Bayesian learning of Gaussian process state space models," in *Proc. Artif. Intell. Statist.*, 2016, pp. 213–221.

[228] C. L. C. Mattos, Z. Dai, A. Damianou, G. A. Barreto, and N. D. Lawrence, "Deep recurrent Gaussian processes for outlier-robust system identification," *J. Process Control*, vol. 60, no. 6, pp. 82–94, Dec. 2017.

[229] C. L. C. Mattos and G. A. Barreto, "A stochastic variational framework for recurrent Gaussian processes models," *Neural Netw.*, vol. 114, pp. 54–72, Apr. 2019.

[230] S. Hochreiter and J. Schmidhuber, "Long short-term memory," *Neural Comput.*, vol. 9, no. 8, pp. 1735–1780, 1997.

[231] H. Nickisch and C. E. Rasmussen, "Approximations for binary Gaussian process classification," *J. Mach. Learn. Res.*, vol. 9, pp. 2035–2078, Oct. 2008.

[232] H.-C. Kim and Z. Ghahramani, "Bayesian Gaussian process classification with the EM-EP algorithm," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 28, no. 12, pp. 1948–1959, Dec. 2006.

[233] K. M. A. Chai, "Variational multinomial logit Gaussian process," *J. Mach. Learn. Res.*, vol. 13, pp. 1745–1808, Jun. 2012.

[234] F. J. R. Ruiz, M. K. Titsias, A. B. Dieng, and D. M. Blei, "Augment and reduce: Stochastic inference for large Categorical distributions," in *Proc. Int. Conf. Mach. Learn.*, vol. 10, 2018, pp. 6997–7006.

[235] B. Fröhlich, E. Rodner, M. Kemmler, and J. Denzler, "Large-scale Gaussian process multi-class classification for semantic segmentation and facade recognition," *Mach. Vis. Appl.*, vol. 24, no. 5, pp. 1043–1053, 2013.

[236] D. Milios, R. Camoriano, P. Michiardi, L. Rosasco, and M. Filippone, "Dirichlet-based Gaussian processes for large-scale calibrated classification," in *Proc. Adv. Neural Inf. Process. Syst.*, 2018, pp. 6008–6018.

[237] A. Naish-Guzman and S. Holden, "The generalized FITC approximation," in *Proc. Adv. Neural Inf. Process. Syst.*, 2008, pp. 1057–1064.

[238] J. Hensman, A. Matthews, and Z. Ghahramani, "Scalable variational Gaussian process classification," in *Proc. Artif. Intell. Statist.*, 2015, pp. 351–360.

[239] D. Hernández-Lobato and J. M. Hernández-Lobato, "Scalable Gaussian process classification via expectation propagation," in *Proc. Artif. Intell. Statist.*, 2016, pp. 168–176.

[240] C. Villacampa-Calvo and D. Hernández-Lobato, "Scalable multi-class Gaussian process classification using expectation propagation," in *Proc. 34th Int. Conf. Mach. Learn.*, 2017, pp. 3550–3559.

[241] F. Wenzel, T. Galy-Fajou, C. Donner, M. Kloft, and M. Opper, "Efficient Gaussian process classification using Pólya-Gamma data augmentation," in *Proc. Int. Conf. Mach. Learn.*, 2018, pp. 5417–5424.

[242] N. G. Polson, J. G. Scott, and J. Windle, "Bayesian inference for logistic models using Pólya–Gamma latent variables," *J. Amer. Stat. Assoc.*, vol. 108, no. 504, pp. 1339–1349, 2013.

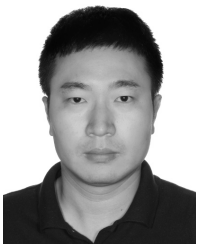
[243] T. Galy-Fajou, F. Wenzel, C. Donner, and M. Opper, "Scalable multi-class Gaussian process classification via data augmentation," in *Proc. NIPS Workshop Approx. Inference*, 2018, pp. 1–12.

[244] H. Liu, Y.-S. Ong, Z. Yu, J. Cai, and X. Shen, "Scalable Gaussian process classification with additive noise for various likelihoods," *arXiv preprint arXiv:1909.06541*, 2019. [Online]. Available: https://arxiv.org/abs/1909.06541

[245] C. Park, J. Z. Huang, and Y. Ding, "Domain decomposition approach for fast Gaussian process regression of large spatial data sets," *J. Mach. Learn. Res.*, vol. 12, pp. 1697–1728, May 2011.

[246] F. Meier, P. Hennig, and S. Schaal, "Incremental local Gaussian regression," in *Proc. Adv. Neural Inf. Process. Syst.*, 2014, pp. 972–980.

[247] S. Flaxman, A. Wilson, D. Neill, H. Nickisch, and A. Smola, "Fast Kronecker inference in Gaussian processes with non-Gaussian likelihoods," in *Proc. Int. Conf. Mach. Learn.*, 2015, pp. 607–616.



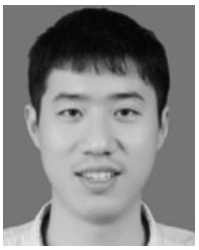
Haitao Liu received the Ph.D. degree from the School of Energy and Power Engineering, Dalian University of Technology, Dalian, China, in 2016.

He is currently a Research Fellow with the Rolls-Royce@NTU Corp Laboratory, Nanyang Technological University, Singapore. His current research interests include multitask learning, large-scale Gaussian process, active learning, and optimization.

Yew-Soon Ong (F'18) received the Ph.D. degree in artificial intelligence in complex design from the Computational Engineering and Design Center, University of Southampton, Southampton, U.K., in 2003.

He is currently a Professor and the Director of the Data Science and Artificial Intelligence Research Center and a Principal Investigator of the Data Analytics and Complex System Program, Rolls-Royce@NTU Corporate Laboratory, Nanyang Technological University, Singapore. His current research interests include computational intelligence, memetic computation, complex design optimization, and machine learning.

Dr. Ong was a recipient of the 2015 *IEEE Computational Intelligence Magazine* Outstanding Paper Award and the 2012 IEEE TRANSACTIONS ON EVOLUTIONARY COMPUTATION Outstanding Paper Award for his work pertaining to Memetic Computation. He is also the Founding Editor-In-Chief of the IEEE TRANSACTIONS ON EMERGING TOPICS IN COMPUTATIONAL INTELLIGENCE and an Associate Editor of the IEEE TRANSACTIONS ON EVOLUTIONARY COMPUTATION, the IEEE TRANSACTIONS ON NEURAL NETWORKS AND LEARNING SYSTEMS, and the IEEE TRANSACTIONS ON CYBERNETICS.



Xiaobo Shen received the Ph.D. degree from the School of Computer Science and Engineering, Nanjing University of Science and Technology, Nanjing, China, in 2017.

From 2015 to 2016, he visited The University of Queensland, Brisbane, QLD, Australia, and the University of Technology Sydney, Ultimo, NSW, Australia. He is currently a Research Fellow with the Rolls-Royce@NTU Corp Laboratory, Nanyang Technological University, Singapore. His current research interests include multiview learning, multilabel learning, and hashing.

Jianfei Cai (SM'07) received the Ph.D. degree in electrical engineering from the University of Missouri at Columbia, Columbia, MO, USA, in 2002.

He is currently a Full Professor and has served as the Head of the Visual & Interactive Computing Division and the Head of Computer Communication Division, School of Computer Engineering, Nanyang Technological University, Singapore. His major research interests include computer vision, visual computing, and multimedia networking.

Dr. Cai has served as the Leading Technical Program Chair of the IEEE International Conference on Multimedia & Expo 2012 and the Leading General Chair of the Pacific-Rim Conference on Multimedia 2012. He has served as an Associate Editor for the IEEE TRANSACTIONS ON CIRCUITS AND SYSTEMS FOR VIDEO TECHNOLOGY from 2006 to 2013. Since 2013, he has been serving as an Associate Editor for the IEEE TRANSACTIONS ON IMAGE PROCESSING.