Amortized Variational Inference with Graph Convolutional Networks for Gaussian Processes

Linfeng Liu
Tufts University

Li-Ping Liu
Tufts University

Abstract

GP Inference on large datasets is computationally expensive, especially when the observation likelihood is non-Gaussian. To reduce the computation, many recent variational inference methods define the variational distribution based on a small number of *inducing points*. These methods have a hard tradeoff between distribution flexibility and computational efficiency. In this paper, we focus on the approximation of GP posterior at a local level: we define a reusable template to approximate the posterior at neighborhoods while maintaining a global approximation. We first construct a variational distribution such that the inference for a data point considers only its neighborhood, thereby separating the calculation for each data point. We then train Graph Convolutional Networks as a reusable model to run inference for each data point. Comparing to previous methods, our method greatly reduces the number of parameters and also the number of optimization iterations. In empirical evaluations, the proposed method significantly speeds up the inference and often gets more accurate results than competing methods.

1 Introduction

As a Bayesian non-parametric method, Gaussian Processes (GPs) (Rasmussen & Williams, 2006) offer flexible and expressive models for various machine learning tasks. The success of a GP model hinges on its efficient inference, which is especially true when the data is large in scale.

A GP defines a distribution over a function space. The central problem of GP inference is to calculate the posterior distribution of function values at training data points. One family of methods for GP inference is variational inference (Jordan et al., 1999; Wainwright & Jordan, 2008), which solves the inference problem by maximizing the Evidence Lower BOund (ELBO). These methods approximate the GP posterior with a multivariate Gaussian distribution. Standard variational inference for GP on large datasets is still expensive, as it needs to optimize the large covariance matrix of the variational distribution and also to invert the prior covariance in gradient calculation. We need to simplify the variational distribution further to speed up the inference procedure.

Variational inference based on inducing points (Quiñonero-Candela & Rasmussen, 2005; Titsias, 2009) uses a low-rank matrix as the covariance of the variational distribution. It first defines a multivariate Gaussian distribution over function values at a small set of M inducing points and then derives distributions of all other data points from the covariance defined by the prior. This method needs space $O(NM)$ and time $O(NM^2)$ in one gradient calculation, with N being the number of data points. Even with stochastic optimization (Hoffman et al., 2013), each random estimation of the ELBO still needs to consider all inducing points (Hensman et al., 2015; Sheth et al., 2015; Dezfouli & Bonilla, 2015; Krauth et al., 2016).

Inducing-point methods devote their computation to approximate many weak correlations between data points and inducing points. At the same time, their approximation of strong correlations between non-inducing points is indirectly through inducing points, and thus likely to be inaccurate. It would be very expensive to directly approximate correlations of this type with a large number of inducing points. This issue is addressed by more recent work, e.g., decoupling estimations of the posterior mean and the posterior variance (Cheng & Boots, 2017; Salimbeni et al., 2018), using inter-domain and subspace inducing points (Hensman et al., 2017; Panos et al., 2018). However, these methods still cannot focus their com-

Proceedings of the 22nd International Conference on Artificial Intelligence and Statistics (AISTATS) 2019, Naha, Okinawa, Japan. PMLR: Volume 89.

putation on important correlations only. Furthermore, inducing-point methods lose the non-parametric flavor of GP, as the flexibility of the variational distribution is solely decided by inducing points.

In this work, we focus the approximation on strong correlations, which are mostly at the local level. We propose to 1) localize the inference for a single data point within a relatively small neighborhood, and 2) reuse a local inference procedure for all data points. With popular kernel settings (Chapter 4 of (Rasmussen & Williams, 2006)) in the prior, the function value at a data point has strong correlations with only function values in a surrounding region. Very likely strong correlations in the posterior also exist only among data points within local regions. Focusing on local correlations brings dramatic computational advantages, while neglecting weak long-distance correlations has little negative effect on the inference accuracy. Our method shares a similar principle with the local GP model (Nguyen-Tuong et al., 2009), but our method has two advantages: a unified inference objective and model reuse.

In this work, we construct a variational distribution with which the inference at a data point takes the information only from its K surrounding neighbors. The variational distribution supports fast sampling from its marginals, hence allowing efficient Monte Carlo approximation with the reparameterization technique (Kingma & Welling, 2013) when dealing with non-Gaussian data likelihoods (Sheth et al., 2015; Dezfouli & Bonilla, 2015; Krauth et al., 2016). Combined with an approximation of the prior, the variational distribution gives a highly decomposable ELBO, hence enabling the stochastic optimization with small batches of terms in the decomposed ELBO. An optimization update with a batch of size N_b takes time only $O(N_b K^3)$.

Inspired by recognition networks used in amortized inference (Kingma & Welling, 2013; Mnih & Gregor, 2014; Miao et al., 2016), we further train two Graph Convolutional Networks (GCN) (Kipf & Welling, 2017) as a global inference model to identify variational parameters for each data point. Particularly, the two networks take observations around a data point as the input and spit out local variational parameters. By reusing the two networks, this technique greatly reduces the number of optimization parameters and also the number of optimization iterations.

The proposed inference method is non-parametric in the sense that the flexibility of the variational distribution grows with the data size. Particularly, it does non-parametric inference with a parametric model.

2 Gaussian Process and Variational Inference

2.1 Gaussian Process

A Gaussian process defines a distribution over the function space and assumes any finite collection of marginals follows a multivariate Gaussian distribution. If a function $f : \mathcal{R}^d \to \mathcal{R}$ is from a Gaussian process $\text{GP}(m(\cdot), \kappa(\cdot,\cdot))$, then it implies that $\mathbb{E}[f(\mathbf{x})] = m(\mathbf{x})$ and $\text{Cov}(f(\mathbf{x}), f(\mathbf{x}')) = \kappa(\mathbf{x}, \mathbf{x}')$, $\forall \mathbf{x}, \mathbf{x}' \in \mathcal{R}^d$. Suppose we have N data points, $(\mathbf{X}, \mathbf{y}) = ((\mathbf{x}_i, y_i) : i = 1, \dots, N)$, and want to model the probability $p(\mathbf{y}|\mathbf{X})$ with GP, then we specify the distribution as follows.

$$f \sim \text{GP}(0(\cdot), \kappa(\cdot,\cdot)), \theta = \lambda(f(\mathbf{x}_i)), y_i \sim \text{DIST}(\theta). \quad (1)$$

Here $0(\cdot)$ is the zero function. The link function $\lambda(\cdot)$ and the distribution DIST can be arbitrary as long as we can calculate $\log p(y_i|f(\mathbf{x}_i))$ and take its gradient with respect to $f(\mathbf{x}_i)$. This formulation is general enough for a wide range of regression and classification problems.

Denote $\mathbf{f} = (f_i = f(\mathbf{x}_i) : i = 1, \dots, N)$, then $\mathbf{f}$ follows a multivariate Gaussian distribution with mean $\mathbf{0}$ and covariance $\mathbf{\Sigma} = \kappa(\mathbf{X}, \mathbf{X})$, the latter of which is referred as kernel matrix. As the main step of GP inference, the calculation of the posterior distribution $p(\mathbf{f}|\mathbf{X}, \mathbf{y})$ is the basis for making predictions. The posterior is often intractable when the noise distribution is non-Gaussian, so approximate inference becomes inevitable. This work focuses on variational inference and approximates $p(\mathbf{f}|\mathbf{X}, \mathbf{y})$ with a variational distribution $q(\mathbf{f})$.

2.2 Variational Inference for GP

Variational inference searches for a good approximation $q(\mathbf{f})$ by maximizing the ELBO, $L(q(\mathbf{f}))$ in (2), with respect to $q(\mathbf{f})$.

$$\log p(\mathbf{y}|\mathbf{X}) \geq L(q(\mathbf{f})) := \mathbb{E}_{q(\mathbf{f})}[\log p(\mathbf{f})] + \mathbb{E}_{q(\mathbf{f})}[\log p(\mathbf{y}|\mathbf{f})] + \mathbb{H}[q]. \quad (2)$$

Here, $\mathbb{H}[q]$ is the entropy of $q(\mathbf{f})$. The bound is tight when $q(\mathbf{f}) = p(\mathbf{f}|\mathbf{X}, \mathbf{y})$. For computational convenience, $q(\mathbf{f})$ is often chosen to be a multivariate Gaussian distribution. Denote its mean as $\boldsymbol{\mu}$ and variance as $\mathbf{V}$, then the optimization objective $L(q(\mathbf{f}))$ becomes $L(\boldsymbol{\mu}, \mathbf{V})$.

Direct calculation of (2) with a general GP model has the following issues. The first term on the right side of (2) needs the inverse of prior covariance $\mathbf{\Sigma}$. The second term often uses Monte Carlo samples to approximate the expectation $\mathbb{E}_{q(f_i)}[p(y_i|f_i)]$, so it needs

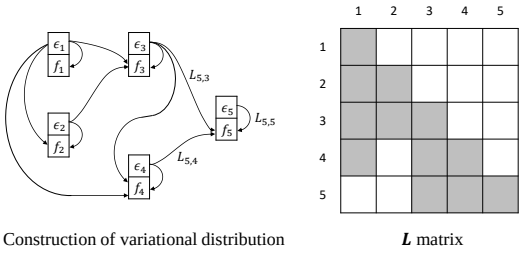


Figure 1: The variational distribution with 5 data points and $K = 2$. Left: the dependence of random variables; $\boldsymbol{\mu}$ is omitted here. Right: the sparsity pattern of the corresponding $\mathbf{L}$ matrix; each arrow on the left corresponds to a non-zero entry in $\mathbf{L}$.

cheap samples from the marginal $q(f_i)$. The entropy term $\mathbb{H}[q]$ needs the log determinant of $\mathbf{V}$.

Besides difficulties in calculating the ELBO, the large number of variables in $\boldsymbol{\mu}$ and $\mathbf{V}$ in large-scale problems poses a further challenge to optimize the ELBO.

3 Amortized Variational Inference for General GP Models

3.1 The Variational Distribution

We construct the variational distribution $q(\mathbf{f})$ as a multivariate Gaussian distribution with mean $\boldsymbol{\mu}$ and covariance $\mathbf{V}$. The covariance $\mathbf{V}$ is further parameterized as $\mathbf{V} = \mathbf{L}\mathbf{L}^\top$, with $\mathbf{L}$ being a sparse lower triangular matrix. The off-diagonal non-zero elements in each row $\mathbf{L}_i, 1 \le i \le N$ are indicated by a set $\alpha(i) \subseteq \{1, \dots, i-1\}$. The set $\alpha(i)$ has at most K non-zero elements, $|\alpha(i)| = \min(K, i-1)$. The sparsity pattern of $\mathbf{L}$ is,

$$\mathbf{L}_{ij} = 0 \quad \text{if } j \notin \alpha(i) \cup \{i\}. \tag{3}$$

The number of free variables in $\mathbf{L}$ is $N(K+1) - K(K+1)/2$. The second term $K(K+1)/2$ corresponds to the extra number of zeros for the first K rows. The size requirement of $\alpha(i)$-s allows storing $\mathbf{L}$ into a $N \times (K+1)$ matrix, which is convenient for matrix computation in implementation.

With this construction, it is easy to sample from $q(\mathbf{f})$. Let $\boldsymbol{\epsilon} \sim \mathcal{N}(\mathbf{0}, \mathbf{I})$ be an N-dimensional Gaussian random white noise, then a sample $\mathbf{f}$ from $q(\mathbf{f})$ can be generated by

$$\mathbf{f} = \boldsymbol{\mu} + \mathbf{L}\boldsymbol{\epsilon}. \tag{4}$$

It is fast to draw samples from the marginal $q(f_i)$: we only need to sample a $(K+1)$-dimensional white noise $(\boldsymbol{\epsilon}_{\alpha(i)}, \epsilon_i)$ and then calculate $f_i = \mu_i + \mathbf{L}_{i,\alpha(i)}\boldsymbol{\epsilon}_{\alpha(i)} + \mathbf{L}_{i,i}\epsilon_i$ as a sample. Figure 1 illustrates the construction of the variational distribution.

The sparsity pattern of $\mathbf{L}$, or $\alpha(i)$ sets, is specified by a directed graph over data points, with $\alpha(i)$ being the parent set of i. We define $\alpha(\cdot)$ sets from the prior kernel matrix. The principle is that, if two data points i and j, $j < i$, are strongly correlated in the prior, then j is very likely to be in $\alpha(i)$.

Formally, we construct $\alpha(i)$ for each i in the following three steps. First, we form an undirected graph over data points by their neighboring relations: if i or j is one of the K most correlated neighbors of the other in prior covariance, then we add an undirected edge (i, j) to the graph. Second, we assign each edge a direction from the smaller index to the larger one, e.g. the edge (i, j) points from j to i if $j < i$. Finally, we adjust every $\alpha(i)$ to make sure that each i has $\min(K, i-1)$ parents: if $\alpha(i)$ has inadequate parents, we add to $\alpha(i)$ more nearest neighbors from data points with indices smaller than i; if $\alpha(i)$ has more than K parents, we remove these parents with larger indices.

The constructed covariance matrix $\mathbf{V}$ is good at approximating *local* correlations. Here *being local* means being *strongly correlated* in the prior kernel, but not about distances. Below we show that $q(\mathbf{f})$ is able to match any covariance values corresponding to edges in the graph mentioned above. This result indicates that the order of data points is not important as long as the graph has edges (in either direction) between data points with strong correlations.

Theorem: Suppose $\mathbf{V}^*$ is the covariance of the true posterior. By setting appropriate $\mathbf{L}$, the matrix $\mathbf{V} = \mathbf{L}\mathbf{L}^\top$ is able to match $\mathbf{V}^*$ at entries corresponding to graph edges, that is, $\mathbf{V}_{ij} = \mathbf{V}^*_{ij},\ \forall\ i,\ j \in \alpha(i)$.

proof sketch: Define $\beta(i) = \alpha(i) \cup \{i\}$. The covariance of $f_{\beta(i)}$ and f_i in the variational distribution $q(\mathbf{f})$ is

$$\mathbf{V}_{\beta(i),i} = \mathbf{L}_{\beta(i)}\mathbf{L}_i^\top = \mathbf{L}_{\beta(i),\beta(i)}\mathbf{L}_{i,\beta(i)}^\top\ . \tag{5}$$

Here, $\mathbf{L}_{\beta(i),\beta(i)}$ is a sub-matrix of $\mathbf{L}$ taking $\beta(i)$ rows and columns. Note that the row $\mathbf{L}_i$ has non-zero elements only at $\beta(i)$. Because $\mathbf{L}$ is a lower triangular matrix, its principal sub-matrix $\mathbf{L}_{\beta(i),\beta(i)}$ is a full-rank matrix. Then we just set $\mathbf{L}_{i,\beta(i)} = (\mathbf{L}^{-1}_{\beta(i),\beta(i)}\mathbf{V}^*_{\beta(i),i})^\top$ so that $\mathbf{V}_{\beta(i),i} = \mathbf{V}^*_{\beta(i),i}$. The argument above works from all $i = 1, \dots N$. □

One possible concern is that the ordering of data points in the formation of the graph and also $\alpha(i)$ sets would significantly impact the performance. However, there are only neglectable performance differences with permutations of data points in our empirical experience.

3.2 Optimization of the ELBO

In this section, we optimize the ELBO with stochastic optimization, which needs a cheap estimation of the objective and its gradient. The strategy is to first decompose the ELBO in (2) into small factors and then estimate its value based on a random selection of factors. Based on this estimation, we calculate stochastic gradients with respect to $\boldsymbol{\mu}$ and $\mathbf{L}$. For convenience of discussion below, we call the three terms in (2) as the expected likelihood term $L_{ell} = \mathbb{E}_{q(\mathbf{f})}\left[p(\mathbf{y}|\mathbf{f})\right]$, the cross entropy term $L_{cross} = \mathbb{E}_{q(\mathbf{f})}\left[p(\mathbf{f})\right]$, and the entropy term $L_{ent} = \mathbb{H}\left[q\right]$. Now we decompose the three terms as follows.

Decompose the likelihood term Observations $\mathbf{y}$ are independent given $\mathbf{f}$, so the likelihood term naturally decomposes as in (6). We only need a random batch S of data points to estimate the entire term L_{ell}. We further approximate the expectation term $\mathbb{E}_{q(f_i)}\left[\log p(y_i|f_i)\right]$ for data point $i \in S$ with a Monte Carlo sample $\hat{f}_i$ from $q(f_i)$. We use reparameterization technique (Kingma & Welling, 2013): the random sample $\hat{f}_i$ is a function of μ_i and $\mathbf{L}_i$, and the gradient of $\tilde{L}_{ell}$ with respect to μ_i and $\mathbf{L}_i$ is propagated through $\hat{f}_i$. The exact calculation and its estimation is shown in (6).

$$
\begin{aligned}
L_{ell} &= \sum_{i=1}^{N} \mathbb{E}_{q(f_i)}\left[\log p(y_i|f_i)\right], \\
\tilde{L}_{ell} &= \frac{N}{|S|}\sum_{i\in S} \log p(y_i|\hat{f}_i).
\end{aligned} \tag{6}
$$

Decompose the entropy term The entropy term is the log determinant of $\mathbf{V}$ plus a constant. We have $\log\det(\mathbf{V}) = \sum_{i=1}^{N}\log(\mathbf{L}_{i,i}^2)$ by the decomposition of $\mathbf{V}$. Then the exact entropy calculation and its estimation with a random batch S is

$$
\begin{aligned}
L_{ent} &= 0.5\left(N\log(2\pi e) + \sum_{i=1}^{N}\log\mathbf{L}_{i,i}^2\right), \\
\tilde{L}_{ent} &= 0.5\left(N\log(2\pi e) + \frac{N}{|S|}\sum_{i\in S}\log\mathbf{L}_{i,i}^2\right).
\end{aligned} \tag{7}
$$

Decompose the cross entropy term The cross entropy term is

$$
\begin{aligned}
L_{cross} = &-0.5\log\det(\boldsymbol{\Sigma}) - 0.5N\log(2\pi) \\
&- 0.5\ \mathrm{tr}\left(\boldsymbol{\Sigma}^{-1}(\mathbf{L}\mathbf{L}^\top + \boldsymbol{\mu}\boldsymbol{\mu}^\top)\right).
\end{aligned} \tag{8}
$$

If we do not learn kernel parameters, then $\log\det(\boldsymbol{\Sigma})$ can be treated as constant with respect to $q(\mathbf{f})$. The inverse of $\boldsymbol{\Sigma}$ is hard to compute, so people often appeal to approximations.

We use the approximation in (Datta et al., 2016): the prior distribution approximately decomposes as $p(\mathbf{f}) \approx \prod_{i=1}^{N} p(f_i|f_{\alpha(i)})$. The conditional $p(f_i|f_{\alpha(i)})$ is from the GP prior and has mean and variance

$$
\begin{aligned}
\eta_{i|\alpha(i)} = \mathbf{b}_i f_{\alpha(i)},\ \boldsymbol{\Sigma}_{i|\alpha(i)} &= \boldsymbol{\Sigma}_{ii} - \boldsymbol{\Sigma}_{i,\alpha(i)}\mathbf{b}_i^\top \\
\text{with}\quad \mathbf{b}_i &= \boldsymbol{\Sigma}_{i,\alpha(i)}\boldsymbol{\Sigma}_{\alpha(i),\alpha(i)}^{-1}.
\end{aligned} \tag{9}
$$

This approximation facilitates decomposable calculations, as it shares a similar structure with $q(\mathbf{f})$. Then the cross entropy term is approximated by

$$
\begin{aligned}
\tilde{L}_{cross} =& \frac{N}{|S|}\sum_{i\in S}\mathbb{E}_{q(f_i,f_{\alpha(i)})}\left[\log p(f_i|f_{\alpha(i)})\right] \\
=& \frac{N}{|S|}\sum_{i\in S} -\frac{1}{2}\boldsymbol{\Sigma}_{i|\alpha(i)}^{-1}\left(\mathbf{r}\mathbf{r}^\top + (\mu_i - \mathbf{b}_i^\top\mu_{\alpha(i)})^2\right) \\
&+ const.
\end{aligned} \tag{10}
$$

Here $\mathbf{r} = \mathbf{L}_i - \mathbf{b}_i^\top\mathbf{L}_{\alpha(i)}$ is a row vector.

We get an estimation of the ELBO by putting together estimations of the three terms. The first two terms are unbiased while the third term is biased because of the approximation of the prior.

$$
L \approx \tilde{L}_{ell} + \tilde{L}_{cross} + \tilde{L}_{ent}. \tag{11}
$$

This estimation of the ELBO is highly decomposable and supports stochastic optimization. Direct optimization of $\boldsymbol{\mu}$ and $\mathbf{L}$ is already an inference method. However, the number of parameters in $\boldsymbol{\mu}$ and $\mathbf{L}$ has a scale of $O(NK)$. Even with SGD, the optimization needs at least a few epochs to converge.

3.3 Inference Network to Reduce Optimization Parameters

In this subsection, we reduce the number of parameters by training a reusable inference network, which directly recognizes variational parameters μ_i and $\mathbf{L}_i$ from observations and the prior.

We define an inference network $\mathcal{M}$ such that, $(\mu_i, \mathbf{L}_i) = \mathcal{M}(\mathbf{y}_{(i,\alpha(i))}, \boldsymbol{\Sigma}_{(i,\alpha(i)),(i,\alpha(i))})$. Ideally, all observations $\mathbf{y}$ and the entire covariance matrix $\boldsymbol{\Sigma}$ should be in the input, as they all affect μ_i and $\mathbf{L}_i$ for i. For the sake of feasibility, here we only use a small part of related data as the input and hope that the network still identifies good values for the two parameters without complete information. Note that the network only needs to output non-zero elements of $\mathbf{L}_i$.

The inference network $\mathcal{M}$ consists of two separate Graph Convolutional Networks (GCN) (Kipf & Welling, 2017) for μ_i and $\mathbf{L}_i$ respectively. A GCN computes hidden layers with the adjacency matrix of a graph. The inputs to the GCN are graph node values.

Here we treat data points $\{i\} \cup \alpha(i)$ as a small graph and then use the kernel sub-matrix $\mathbf{\Sigma}_{(i,\alpha(i)),(i,\alpha(i))}$ as its adjacency matrix. We treat the observations $\mathbf{y}_{(i,\alpha(i))}$ as graph node values. In this graph, i is unique because we are computing parameters for i. To break the symmetry relation between i and any other data point in $\alpha(i)$, we set $\mathbf{A} = [\mathbf{\Sigma}_{i,i}, \mathbf{0}; \mathbf{\Sigma}_{\alpha(i),i}, \mathbf{\Sigma}_{\alpha(i),\alpha(i)}]$ as the adjacency matrix for the GCN. The input to the GCN is $\mathbf{H}^{(0)} = \mathbf{y}_{(i,\alpha(i))}$.

The hidden layer $\mathbf{H}^{(l)}$ of a GCN is computed from its previous layer $\mathbf{H}^{(l-1)}$ as follows,

$$\mathbf{H}^{(l)} = \sigma(\mathbf{D}^{-\frac{1}{2}}\mathbf{A}\mathbf{D}^{-\frac{1}{2}}\mathbf{H}^{(l-1)}\mathbf{W}^{(l)}), \quad l = 1, \ldots, L. \tag{12}$$

Here $\mathbf{D}$ is a diagonal matrix, whose diagonal is the row sum of $\mathbf{A}$. $\mathbf{W}^{(l)}$ is the trainable weight matrix for the l^{th} layer. The activation function $\sigma(\cdot)$ is set to be identity for the last layer and ReLU for other layers.

By setting $W^{(L)}$ as a column vector, the output $\mathbf{H}^{(L)}$ of GCN is a vector with length $K+1$. The variational mean μ_i is obtained by taking the mean of the output $\mathbf{H}^{(L)}$ from the first GCN, while non-zero elements of $\mathbf{L}_i$ are set to be $\mathbf{H}^{(L)}$ from the second GCN. GCNs are suitable for this inference task: they use the adjacency matrix to consider relations among nearby observations, which is in a similar manner as the inference procedure.

We apply the two GCNs to the ELBO estimation in (11), so they are learned by the stochastic optimization of the ELBO. The number of optimization parameters is reduced to a constant, i.e. the size of the inference network. The optimization procedure needs much less iterations in training the two GCNs than direct optimizing the variational parameters.

We call this inference method as Amortized Inference for Gaussian Processes, and AIGP for short. In the calculation of objective values and gradients, the most expensive calculation of a single data point is (10), which takes time $O(K^3)$. For a batch S of N_b data points, the calculation in (11) only takes time $O(N_b K^3)$. The gradient calculation takes a similar amount of time. Prior the run of our algorithm, we need to check nearest neighbors in prior covariance and construct the graph – the time complexity is $O(n \log n)$ with advanced data structures such as k-d trees.

3.4 Prediction

Now we have the approximation $q(\mathbf{f})$ of the posterior $p(\mathbf{f}|\mathbf{X}, \mathbf{y})$ and are ready to make predictions for new data points. Let $\mathbf{x}_\star$ be a new data point, then the predictive distribution is.

$$\begin{aligned} p(y_\star|\mathbf{x}_\star, \mathbf{X}, \mathbf{y}) &= \int_{f_\star} p(y_\star|f_\star) p(f_\star|\mathbf{x}_\star, \mathbf{X}, \mathbf{y}) \, \mathrm{d}f_\star \\ &\approx \int_{f_\star} p(y_\star|f_\star) q(f_\star|\mathbf{x}_\star, \mathbf{X}, \mathbf{y}) \mathrm{d}f_\star \ . \end{aligned} \tag{13}$$

Here $q(f_\star|\mathbf{x}_\star, \mathbf{X}, \mathbf{y}) = \int_{\mathbf{f}_{\alpha_\star}} p(f_\star|\mathbf{f}_{\alpha_\star}) q(\mathbf{f}_{\alpha_\star}) \mathrm{d}\mathbf{f}_{\alpha_\star}$, and $p(f_\star|\mathbf{f}_{\alpha_\star})$ is from the prior. The mean and variance of $q(f_\star|\mathbf{x}_\star, \mathbf{X}, \mathbf{y})$ are

$$\begin{aligned} \mu_\star &= \mathbf{b}_\star \mu_{\alpha_\star}, \\ \sigma_\star^2 &= \mathbf{\Sigma}_{\star,\star} - \mathbf{\Sigma}_{\star,\alpha_\star}\mathbf{b}_\star^\top + \mathbf{b}_\star(\mathbf{L}_{\alpha_\star}\mathbf{L}_{\alpha_\star}^T)\mathbf{b}_\star^\top. \end{aligned} \tag{14}$$

Here $\alpha_\star$ denotes K neighbors that have the largest prior covariance with $\mathbf{x}_\star$, and $\mathbf{b}_\star = \mathbf{\Sigma}_{\star,\alpha_\star}\mathbf{\Sigma}_{\alpha_\star,\alpha_\star}^{-1}$.

The integral in (13) may not have a closed-form solution, but we can easily approximate it with Monte Carlo samples: $\frac{1}{S}\sum_{s=1}^{S} p(y_\star|f_\star^s)$, with $f_\star^s$ being samples from $q(f_\star|\mathbf{x}_\star, \mathbf{X}, \mathbf{y})$. The approximation is accurate as the integral variable is in one dimension.

3.5 Discussion of the Approximation of the Prior

One possible criticism is that our method loses information of the prior when it uses a directed graphical model as the approximation (see the equation above (9)). However, the empirical evaluation does not indicate the approximation hurting the predictive performance. Actually, the lost of prior information is inevitable when an inference method simplifies the posterior covariance. Inducing-point methods ignore the prior covariance between any pair of non-inducing points: the corresponding entries of the prior kernel matrix do not enter the calculation (see Eq. (9) in Titsias (2009) and Eq. (19) in Hensman et al. (2015)). Comparing to inducing-point methods, which may neglect some strong covariance among non-inducing points, our method truncates weak conditional dependency among data points that have low precision values in the prior.

4 Experiment

We evaluate our method on three tasks, bird abundency estimation, raster data modeling, and flight delay regression. The details of the three tasks are in their respective subsections.

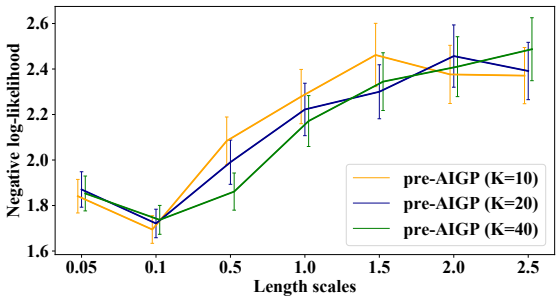
We compared our method with three state-of-the-art methods, Scalable Variational Gaussian Process (SVGP) (Hensman et al., 2015), Scalable Automated Variational Inference for Gaussian Process (SAVIGP) (Dezfouli & Bonilla, 2015), and Decoupled

Table 1: Negative log-likelihood of comparison methods on eBird data. Smaller values are better.

M	SVGP	SAVIGP	DGP	K	pre-AIGP	AIGP
200	2.29±.10 \691s	2.26±.07 \220s	2.05±.06 \201s	10	**1.57±.04** \23ks	1.78±.05 \5.1s
1000	2.28±.10 \50ks	2.20±.07 \939s	1.99±.06 \212s	20	**1.61±.04** \15ks	1.79±.05 \10s
2000	2.38±.10 \50ks	2.20±.07 \2.7ks	1.98±.06 \256s	40	**1.60±.04** \50ks	1.79±.05 \36s

M	SVGP	SAVIGP	DGP
200	0.5	1.0	1.5
1000	0.5	0.5	1.0
2000	0.5	0.5	1.0
K	pre-AIGP	AIGP	
10	0.1	0.1	
20	0.1	0.1	
40	0.1	0.1	

(a)



(b)

Figure 2: (a) Different length scale chosen through cross-validation by SVGP, SAVIGP, DGP, pre-AIGP, and AIGP with different configurations. (b) Negative log-likelihood with different length scales on validation set.

Gaussian Processes (DGP) (Cheng & Boots, 2017). All three methods follow the scheme of inducing points. We use the implementation of SVGP by GPFlow (Matthews et al., 2017), the implementations of SAVIGP by its authors, and the implementation of DGP by Faust (2018). We implement likelihood distribution for three methods when necessary. We also compare the method that directly optimizes the ELBO in (11) without inference network. Denote this method as pre-AIGP.

We use the RBF kernel for all experiments. The length scale is a hyperparameter, which is selected from the candidate set $\{0.05, 0.1, 0.5, 1.0, 1.5, 2.0, 2.5\}$ by checking performance on a validation set. We randomly split each dataset into three subsets for training (70%), validation (10%), and testing (20%). We report negative log-likelihood on the test dataset as the performance measure for the five algorithms in comparison. The maximum training time for each algorithm on the training dataset is 13.89 hours (50ks).

SVGP, DGP, pre-AIGP, and AIGP are all optimized by stochastic optimization with mini-batches. The optimizer for these methods is Tensorflow AdaGrad (Abadi et al., 2016). SAVIGP uses L-BFGS-B optimizer with default settings from the package. We test SVGP and SAVIGP with $M = 200, 1000, 2000$ random inducing points. We allow SVGP to choose inducing points through K-means clustering to seek better performance. With any number of inducing points over 2000, SVGP and SAVIGP will take very long time to converge. DGP uses separate inducing points to estimate the variational mean and covariance. The paper suggests to use a smaller number of inducing points for covariance estimation. We use 200 inducing points for covariance estimation and vary the number of inducing points ($M = 200, 1000, 2000$) for mean estimation. We also have tried $M = 4000$ for DGP and found only insignificant performance improvement. We collect the result when the algorithm converges on the validation set or reaches the time limit. For pre-AIGP and AIGP, we vary K as $K = 10, 20, 40$. We have tested different GCN structures for AIGP and found typical configurations of the network can give reasonably good performances. We finally use a network structure with three layers with dimensions 20, 10, and 1.

4.1 eBird Dataset

The eBird project (Sullivan et al., 2009) hosts a repository for the bird-watching community to report bird observations at different locations around the world. In this experiment, we consider the population of the bird species *Savannah Sparrow* in February from 2012 to 2016 across the contiguous United States. In our data preprocessing step, we retain all positive observations and a random selection from exceedingly many zero observations – the dataset after preprocessing has 15,085 observations with 6,477 positive data points. We also rescale the input location values such that an Euclidean distance of 0.1 corresponds to a geographic distance of 20 miles. The link function in (1) is defined as the softplus function, $\lambda(x) = \log(1 + \exp(x))$. The likelihood distribution of an observation y_i at location i is Poisson with mean $\lambda(f_i)$.

Table 1 shows the negative predictive log-likelihood

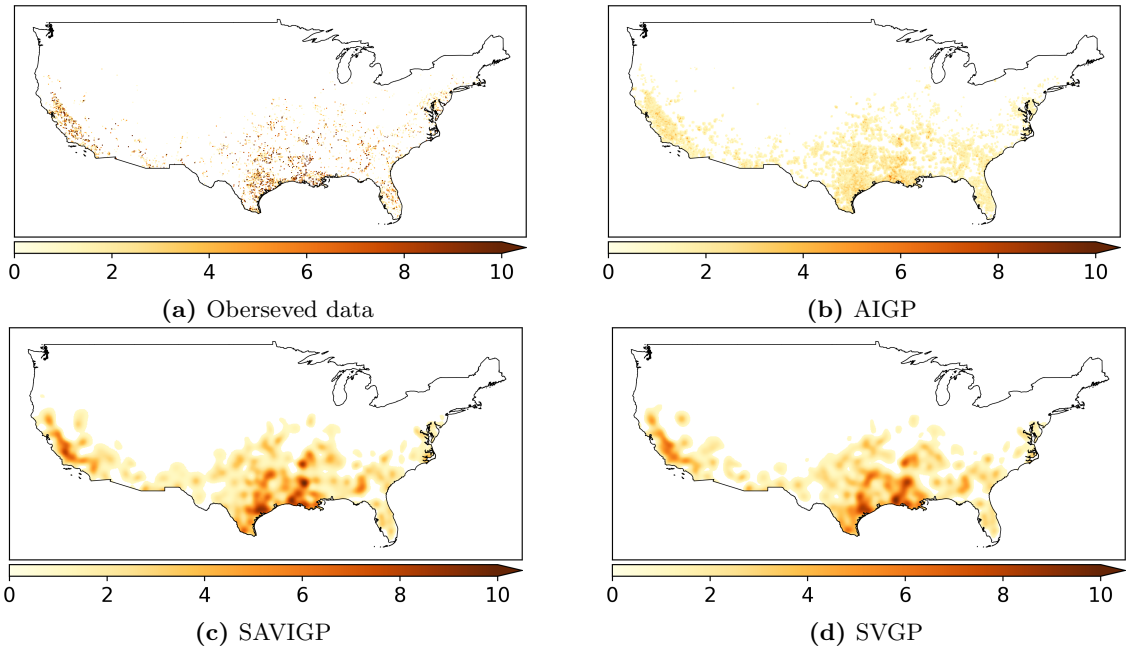


Figure 3: eBird dataset for the distribution of savannah sparrow in February between 2012 and 2016 across US. (a) the observed distribution, (b)-(d) the regression surfaces of the AIGP, SAVIGP, and SVGP.

on the test set. Smaller values mean better predictive performances. We observe that pre-AIGP and AIGP significantly outperform the other three approaches. The standard deviations obtained from pre-AIGP and AIGP are also smaller than the SVGP, SAVIGP, and DGP. This result also indicates that pre-AIGP and AIGP is able to achieve good inference performance with a small number K.

To check how data ordering affects the results, we further run AIGP 10 times and each time randomly permute the training examples. The standard deviation of negative log-likelihood for K=10, 20, 40 are all 0.02. This results suggest that impacts of data ordering are imperceptible, at least for eBird dataset.

We also investigate how the inference method impacts the choice of hyperparameters. Figure 2(a) shows the length scale chosen by different methods through cross validation. The results indicate that SVGP, SAVIGP, and DGP prefer a smoother prior than pre-AIGP and AIGP. As we have analyzed before, a small length scale weakens the correlation between data points and inducing points, then the three methods with a too small number of inducing points will get poor inference results. With this limitation, SVGP, SAVIGP, and DGP have to choose large length scale values. The trend in 2(a) also indicates a smaller length scale should be used if there are more inducing points.

Figure 2(b) shows validation performance of pre-AIGP with different length scale and K values. Pre-AIGP consistently prefers length scale value 0.1. Combining Figure 2 (a) and (b), we hypothesize that the length scale 0.1 chosen by pre-AIGP is ideal for the application. This choice limits the influence of a data point within the distance of 50 miles, which is also reasonable in ecology.

In Figure 3, we plot the data (a) and the surfaces predicted by the AIGP (b), SAVIGP (c), and SVGP (d). For visualization purpose in (a), we use large pixels to indicate data points. The surfaces inferred by AIGP keeps more detailed information, which is good for prediction considering its performance in Table 1.

4.2 NightLight Raster Dataset

In this task, we fit spatial raster data, the NightLight image, with GP models. The dataset is a satellite image of the nighttime light over US contiguous states by Earth at Night project (Figure 4). We use GP models to fit pixel values given their coordinates. In this experiment, we preprocess the image by first converting the RGB image to greyscale and then downscaling the image. Finally, we get 56,722 pixels with their values rescaled to the range [0, 1]. We use a log-normal distribution with variance parameter $\sigma^2 = 0.01$ as the likelihood distribution DIST (the actual standard deviation of the distribution is between 0.1 and 0.27). The mean parameter of this likelihood distribution is the Gaussian output f_i.

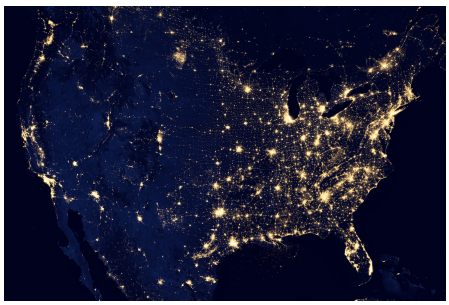


Figure 4: The raster data of nighttime light.

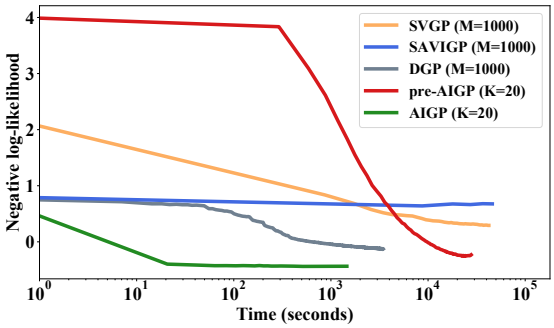


Figure 5: Training time on validation set.

Table 2: Negative predictive log-likelihood on NightLight data. Smaller values are better.

M	SVGP	SAVIGP	DGP	K	pre-AIGP	AIGP
200	0.71±.04 \674s	0.72±.01 \867s	0.45±.01 \1.1ks	10	-0.36±.01 \22ks	**-0.47±.01** \17s
1000	0.34±.03 \50ks	0.64±.02 \11ks	-0.12±.01 \1.3ks	20	-0.20±.03 \50ks	**-0.42±.01** \27s
2000	0.41±.03 \50ks	0.64±.03 \16ks	-0.24±.01 \1.5ks	40	-0.07±.02 \50ks	**-0.46±.01** \93s

Table 3: Negative predictive log-likelihood on Flight data. Smaller values are better.

M	SVGP	SAVIGP	DGP	K	AIGP
200	-1.07±.04 \43s	-0.90±.06 \13ks	-1.10±.002 \770s	10	**-1.27±.002** \34s
1000	-1.01±.03 \1.1ks	-0.84±.04 \50ks	-1.04±.002 \900s	20	**-1.28±.002** \86s
2000	-1.02±.05 \3.8ks	-0.86±.07 \50ks	-1.11±.002 \3.3ks	40	**-1.27±.002** \533s

Table 2 shows the negative predictive log-likelihood of all methods. Pre-AIGP and AIGP both outperform the other three methods. On this dataset, AIGP performs even better than pre-AIGP. It is because pre-AIGP does not fully converge within time limit. AIGP often converges in the first training epoch.

We compare the running speed of all methods. Figure 5 shows the negative log-likelihood values in validation against training time for all five methods. There is not an equivalent setting for our methods and a setting for inducing-point methods, so we choose typical settings for both type of methods. Pre-AIGP outperforms SVGP and SAVIGP within reasonable time. AIGP converges much faster than all other methods.

4.3 Flight Delay Dataset

This data set contains 5.9 million records of flight arrivals and departures within the US in 2008[1]. Following Hensman et al. (2013), we select 8 dimensions as the attributes and randomly choose 800,000 instances from the original dataset as the dataset for this experiment. The task is to use the 8 attributes to fit the delay time. In the data preprocessing step, we standardize input features, remove extreme delay time values, and then translate and scale of all delay time to [0, 1]. We set the likelihood distribution DIST to be the log-normal distribution with $\sigma^2 = 0.01$ and μ being the GP output.

The predictive performances and training time of the four methods are shown in Table 3. AIGP has the best predictive performance and also runs much faster than the other three methods. We observe that AIGP converges after training with about 100k data points (about one sixth epoch). Pre-AIGP cannot run on this dataset due to too many parameters.

5 Conclusion

In this work, we propose a new variational inference method, AIGP, for GP models with general noises. The proposed method has a highly decomposible ELBO , so it can run stochastic optimization efficiently to learn the variational distribution. AIGP retains the flavor of non-parametric learning by inferring function values directly at data points instead of through inducing points. AIGP also reduces the number of variational parameters by training an inference network to recognize variational parameters from nearby data points. It only uses a relatively small number of iterations to converge and greatly speeds up the inference procedure.

[1]From http://stat-computing.org/dataexpo/2009/

References

Abadi, M., Barham, P., Chen, J., Chen, Z., Davis, A., Dean, J., Devin, M., Ghemawat, S., Irving, G., Isard, M., et al. Tensorflow: A system for large-scale machine learning. In *OSDI*, volume 16, pp. 265–283, 2016.

Cheng, C.-A. and Boots, B. Variational inference for gaussian process models with linear complexity. In *Advances in Neural Information Processing Systems*, pp. 5190–5200, 2017.

Datta, A., Banerjee, S., Finley, A. O., and Gelfand, A. E. Hierarchical nearest-neighbor gaussian process models for large geostatistical datasets. *Journal of the American Statistical Association*, 111(514):800–812, 2016.

Dezfouli, A. and Bonilla, E. V. Scalable inference for gaussian process models with black-box likelihoods. In Cortes, C., Lawrence, N. D., Lee, D. D., Sugiyama, M., and Garnett, R. (eds.), *Advances in Neural Information Processing Systems 28*, pp. 1414–1422. Curran Associates, Inc., 2015.

Earth at Night. https://earthobservatory.nasa.gov/IOTD/view.php?id=79800.

Faust, A. Decoupled gaussian process (github implementation). https://github.com/google/decoupled_gaussian_process, 2018.

Hensman, J., Fusi, N., and Lawrence, N. D. Gaussian processes for big data. In *Conference on Uncertainty in Artificial Intellegence*, pp. 282–290, 2013.

Hensman, J., Matthews, A. G. d. G., and Ghahramani, Z. Scalable variational Gaussian process classification. In *Proceedings of the Eighteenth International Conference on Artificial Intelligence and Statistics*, 2015.

Hensman, J., Durrande, N., and Solin, A. Variational fourier features for gaussian processes. *The Journal of Machine Learning Research*, 18(1):5537–5588, 2017.

Hoffman, M. D., Blei, D. M., Wang, C., and Paisley, J. Stochastic variational inference. *The Journal of Machine Learning Research*, 14(1):1303–1347, 2013.

Jordan, M. I., Ghahramani, Z., Jaakkola, T. S., and Saul, L. K. An introduction to variational methods for graphical models. *Machine Learning*, 37(2):183–233, 1999.

Kingma, D. P. and Welling, M. Auto-encoding variational bayes. In *Proceedings of the 2nd International Conference on Learning Representations (ICLR)*, number 2014, 2013.

Kipf, T. N. and Welling, M. Semi-supervised classification with graph convolutional networks. In *International Conference on Learning Representations (ICLR)*, 2017.

Krauth, K., Bonilla, E. V., Cutajar, K., and Filippone, M. AutoGP: Exploring the Capabilities and Limitations of Gaussian Process Models. *ArXiv e-prints*, October 2016.

Matthews, A. G. d. G., van der Wilk, M., Nickson, T., Fujii, K., Boukouvalas, A., León-Villagrá, P., Ghahramani, Z., and Hensman, J. GPflow: A Gaussian process library using TensorFlow. *Journal of Machine Learning Research*, 18(40):1–6, apr 2017. URL http://jmlr.org/papers/v18/16-537.html.

Miao, Y., Yu, L., and Blunsom, P. Neural variational inference for text processing. In *Proceedings of the 33rd International Conference on International Conference on Machine Learning*, pp. 1727–1736, 2016.

Mnih, A. and Gregor, K. Neural variational inference and learning in belief networks. In *Proceedings of the 31th International Conference on Machine Learning*, pp. 1791–1799, 2014.

Nguyen-Tuong, D., Peters, J. R., and Seeger, M. Local gaussian process regression for real time online model learning. In *Advances in Neural Information Processing Systems*, pp. 1193–1200, 2009.

Panos, A., Dellaportas, P., and Titsias, M. K. Fully Scalable Gaussian Processes using Subspace Inducing Inputs. *ArXiv e-prints*, July 2018.

Quiñonero-Candela, J. and Rasmussen, C. E. A unifying view of sparse approximate gaussian process regression. *Journal of Machine Learning Research*, 6(Dec):1939–1959, 2005.

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

Salimbeni, H., Cheng, C.-A., Boots, B., and Deisenroth, M. Orthogonally Decoupled Variational Gaussian Processes. *ArXiv e-prints*, September 2018.

Sheth, R., Wang, Y., and Khardon, R. Sparse variational inference for generalized gaussian process models. In *Proceedings of the 32Nd International Conference on International Conference on Machine Learning*, ICML'15, pp. 1302–1311. JMLR.org, 2015.

Sullivan, B. L., Wood, C. L., Iliff, M. J., Bonney, R. E., Fink, D., and Kelling, S. ebird: A citizen-based

bird observation network in the biological sciences. *Biological Conservation*, 142(10):2282–2292, 2009.

Titsias, M. K. Variational learning of inducing variables in sparse gaussian processes. In *Proceedings of the Twelfth International Conference on Artificial Intelligence and Statistics*, pp. 567–574, 2009.

Wainwright, M. J. and Jordan, M. I. Graphical models, exponential families, and variational inference. *Foundations and Trends in Machine Learning*, 1(1–2):1–305, 2008.