Bayesian Posterior Approximation via Greedy Particle Optimization[*]

Futoshi Futami,[1,2] **Zhenghang Cui,**[1,2] **Issei Sato,**[1,2] **Masashi Sugiyama**[2,1]
[1]The Univiersity of Tokyo [2]RIKEN
futami@ms.k.u-tokyo.ac.jp, cui@ms.k.u-tokyo.ac.jp, sato@k.u-tokyo.ac.jp, sugi@k.u-tokyo.ac.jp

Abstract

In Bayesian inference, the posterior distributions are difficult to obtain analytically for complex models such as neural networks. Variational inference usually uses a parametric distribution for approximation, from which we can easily draw samples. Recently discrete approximation by particles has attracted attention because of its high expression ability. An example is Stein variational gradient descent (SVGD), which iteratively optimizes particles. Although SVGD has been shown to be computationally efficient empirically, its theoretical properties have not been clarified yet and no finite sample bound of the convergence rate is known. Another example is the Stein points (SP) method, which minimizes kernelized Stein discrepancy directly. Although a finite sample bound is assured theoretically, SP is computationally inefficient empirically, especially in high-dimensional problems. In this paper, we propose a novel method named *maximum mean discrepancy minimization by the Frank-Wolfe algorithm (MMD-FW)*, which minimizes MMD in a greedy way by the FW algorithm. Our method is computationally efficient empirically and we show that its finite sample convergence bound is in a linear order in finite dimensions.

Introduction

In Bayesian inference, approximating the posterior distribution $p(x)$ over parameter x is the most important task in general. When we express the prior distribution as $p_0(x)$ and the likelihood as $L(D|x)$ where D denotes observations, the posterior distribution can be obtained up to a constant factor as $p(x) \propto L(D|x)p_0(x)$. In many cases, analytical expression of the normalizing constant cannot be obtained; thus, we need an approximated posterior $\hat{p}(x)$, which can be used, e.g., for calculating the predictive distribution (Bishop 2006):

$$Z_{L,\hat{p}} = \int L(y|x)\hat{p}(x)dx, \tag{1}$$

where $L(y|x)$ is the likelihood function of a new observation y given parameter x. Variational inference (VI) is widely used as an approximation method for the posterior distribution (Blei, Kucukelbir, and McAuliffe 2017). VI approximates the target distribution with a parametric distribution, from which we can easily draw samples. In VI, we often consider the mean field assumption and use parametric models in the exponential family (Blei, Kucukelbir, and McAuliffe 2017). Although these assumptions are used to make optimization computationally tractable, they are often too restrictive to approximate the target distribution well. Therefore, the approximate distribution never converges to the target distribution in general, which means that the approximation of Eq.(1) is biased, and no theoretical guarantee is assured.

An alternative way is a discrete approximation of the target distribution by using a set of particles (Bishop 2006), $\hat{p}(x) = \sum_{n=1}^{N} \delta(x, x_n)/N$, where N is the number of particles and δ is the Dirac delta function. Particle approximation is free of VI assumptions and thus is more expressive. The Monte Carlo (MC) method is typically used to draw particles randomly and independently (Bishop 2006). However, the drawbacks of MC are that vast computational resources are required to sample from multi-modal and high-dimensional distributions, and it is hard to estimate when to stop the algorithm.

Recently, methods that optimize particles through iterative updates have been explored. A representative example is Stein variational gradient descent (SVGD) (Liu and Wang 2016), which iteratively updates all particles in the direction that is characterized by kernelized Stein discrepancy (KSD). The update is actually implemented by gradient descent and SVGD empirically works well in high-dimensional problems. However, theoretical properties of SVGD have not been clarified and no finite sample bound of the convergence rate is known (Liu 2017). Another example is the Stein points (SP) method (Chen et al. 2018), which directly minimizes KSD. Although this method is assured by a finite sample convergence bound, it is not practically feasible in high-dimensional problems due to the curse of dimensionality, because gradient descent is not available and sampling or grid search needs to be used for optimization. Moreover, the number of evaluations of the gradient of the log probability, which usually requires vast computation costs, is four times that of SVGD.

We aim to develop a discrete approximation method that is computationally efficient, works well in high-dimensional problems, and also has a theoretical guarantee for the convergence rate. In this paper, we propose *maximum mean discrep-*

[*]A longer version including Appendix is available at https://arxiv.org/abs/1805.07912

ancy minimization by the Frank-Wolfe algorithm (MMD-FW) in a greedy way. Our convex formulation of discrete approximation enables us to use the Frank-Wolfe (FW) algorithm (Jaggi 2013) and to derive a finite sample bound of the convergence rate.

Our contributions in this paper are three-fold:
1. We formulate a discrete approximation method in terms of convex optimization of MMD in a reproducing kernel Hilbert space (RKHS), and solve it with the FW algorithm.
2. Our algorithm is computationally efficient and empirically works well in high-dimensional problems. It has a guaranteed finite sample bound of the convergence rate.
3. We show empirically that our method compares favorably with existing particle optimization methods.

Preliminary

In this section, we review two existing particle optimization methods, SVGD and SP. After that, we introduce MMD which is our objective function. We assume that $x \in \mathbb{R}^d$ and let $k : X \times X \to \mathbb{R}$ be the reproducing kernel of an RKHS $\mathcal{H}$ of functions $X \to \mathbb{R}$ with the inner product $\langle \cdot, \cdot \rangle$ and $\| \cdot \|_{\mathcal{H}}$ is the assosiated norm, where $X \subseteq \mathbb{R}^d$ denotes the input domain.

Stein variational gradient descent (SVGD)

We first prepare initial particles $\hat{p}_0(x) = \sum_{n=1}^{N} \delta(x, x_n)/N$ and iteratively update them by a transformation, $T(x) = x + \epsilon\phi(x)$, where $\phi(x)$ is a perturbation direction. When the current empirical distribution is $\hat{p}(x) = \sum_{n=1}^{N} \delta(x, x_n)/N$, then $\phi(x)$ is chosen to maximally decrease the Kullback-Leibler (KL) divergence between the empirical distribution $\hat{p}$ formed by the particles and the target distribution p, $\phi^*(x) = \arg\max_{\phi \in \mathcal{F}} \left\{ -\frac{d}{d\epsilon} \mathrm{KL}(\hat{p}_{[T]} \| p)|_{\epsilon=0} \right\}$, where $\mathcal{F}$ denotes a set of candidate functions from which we choose map ϕ, and $\hat{p}_{[T]}(z) = \hat{p}(T^{-1}(z)) \cdot |\det(\nabla_z T^{-1}(z))|$. Liu et al. (Liu and Wang 2016) proved that this problem is characterized by the Stein operator, $-\frac{d}{d\epsilon} \mathrm{KL}(\hat{p}_{[\epsilon\phi]} \| p)|_{\epsilon=0} = \mathbb{E}_{x \sim \hat{p}}[\mathcal{S}_p \phi(x)]$, where $\mathcal{S}_p$ denotes the Stein operator $\mathcal{S}_p \phi(x) = \nabla \ln p(x) \phi(x)^\top + \nabla \cdot \phi(x)$ which acts on a $d \times 1$ vector function ϕ and returns a scalar value function. Thus, the optimization problem is $\mathcal{S}(\hat{p} \| p) := \max_{\phi \in \mathcal{F}} \{\mathbb{E}_{x \sim \hat{p}}[\mathcal{S}_p \phi(x)]\}$. The problem is how to choose an appropriate $\mathcal{F}$. Liu et al. (Liu and Wang 2016) showed that when $\mathcal{F}$ is the unit ball in an RKHS with kernel k, the optimal map can be expressed in the following way. Let $\mathcal{H}_0$ be an RKHS defined by a kernel $k(x, x')$ and $\mathcal{H} = \mathcal{H}_0 \times \cdots \times \mathcal{H}_0$ be the $d \times 1$ vector-valued RKHS. We define $\mathcal{S}_p \otimes k(x, \cdot) := \nabla \ln p(x) k(x, \cdot) + \nabla_x k(x, \cdot)$, then, the optimal direction is given by $\phi^*_{\hat{p},p}(\cdot) = \mathbb{E}_{x \sim \hat{p}}[\nabla_x \ln p(x) k(x, \cdot) + \nabla_x k(x, \cdot)]$. We iteratively update particles following the above direction and obtain the empirical approximation with $\{x_n\}_{n=1}^{N}$. Theoretical analysis has been conducted in terms of the gradient flow and has shown convergence to the true target distribution asymptotically (Liu 2017). However, no finite sample bound has been established. The norm of the optimal direction, $\mathcal{S}(\hat{p} \| p) = \|\phi^*_{\hat{p},p}\|_{\mathcal{H}} = \sqrt{\mathbb{E}_{x,y \sim \hat{p}} k_s(x, y)}$ where $k_s(x, y) = \nabla_x \nabla_y k(x, y) + \nabla_x k(x, y) \nabla_y \ln p(y) + \nabla_y k(x, y) \nabla_x \ln p(x) + k(x, y) \nabla_x \ln p(x) \nabla_y \ln p(y)$, is called kernelized stein discrepancy (KSD)(Liu, Lee, and Jordan 2016).

Stein points (SP)

SP (Chen et al. 2018) minimizes the above KSD directly. When q is given by a discrete approximation $\hat{p} = \sum_{n=1}^{N} \delta(x, x_n)/N$, KSD can be written as $\mathcal{S}(\hat{p} \| p) = \sqrt{\sum_{i,j=1}^{N} k_s(x_i, x_j)}$. In SP, to obtain the n-th particle, we solve $\arg\min_x \sum_{i=1}^{n-1} k_s(x_i, x)$ or $\arg\min_x \sum_{i=1}^{n-1} k_s(x_i, x) + k_s(x, x)/2$. To solve these problems, the authors of the paper (Chen et al. 2018) proposed using sampling methods or grid search. However, those methods are not applicable to high-dimensional problems due to the curse of dimensionality. Although an alternative way is to use gradient descent, this is computationally difficult in high-dimensional problems since this method needs to calculate the Hessian at each iteration. Moreover, the computation cost for evaluating the derivative of the log probability is 4 times compared to SVGD. An advantage of this method is that a finite sample convergence bound is assured theoretically.

Maximum mean discrepancy

SVGD and SP use KSD as the direction of the update and the objective function. In our proposed method, we use MMD as the objective function. MMD is a kind of the worst-case error between expectations. For a given test function f, we express the integral with respect to the true posterior distribution p as $Z_{f,p} = \int f(x)p(x)dx$. We denote an approximation of $Z_{f,p}$ as $Z_{f,\hat{p}}$, where p is approximated by $\hat{p}$ in the same way as Eq. (1). From here, we consider the weighted empirical distribution $\hat{p}(x) = \sum_{n=1}^{N} w_n \delta(x, x_n)$, where w_n are the weights of each particle. Then MMD (Gretton et al. 2012) is defined as

$$\begin{aligned} \mathrm{MMD}(\{w_i, x_i\}_{i=1}^{N})^2 &:= \frac{1}{2} \sup_{f \in \mathcal{H}: \|f\|_{\mathcal{H}}=1} \left| Z_{f,p} - \sum_{i=1}^{N} w_i f(x_i) \right|^2 \\ &= \frac{1}{2} \|\mu_p - \mu_{\hat{p}}\|^2_{\mathcal{H}} \\ &= \frac{1}{2} \left\| \mu_p - \sum_{i=1}^{N} w_i k(x_i, \cdot) \right\|^2_{\mathcal{H}}, \end{aligned} \tag{2}$$

where $\mu_p = \int k(\cdot, x)p(x)dx \in \mathcal{H}$ and we introduce the coefficient $\frac{1}{2}$ for convenience in later calculation. We also express $\mathrm{MMD}(\{w_i, x_i\}_{i=1}^{N})^2$ as $\mathrm{MMD}(\mu_{\hat{p}})^2$ for simplicity.

Proposed methods

In this section, we formally develop our MMD-FW. We will introduce the FW algorithm in an RKHS, propose our MMD-FW, and give a finite sample convergence bound of our method.

MMD minimization by the FW algorithm (MMD-FW)

On the basis of the existing methods reviewed in Section 2, we would like to obtain a method to approximate the posterior

Algorithm 1: Frank-Wolfe (FW) Algorithm

```
1: Let $\boldsymbol{x}_0 \in \mathcal{D}$
2: for $n = 0, \dots, N$ do
3:     Compute $\boldsymbol{s} = \mathrm{argmin}_{\boldsymbol{s} \in \mathcal{D}} \langle \boldsymbol{s}, \nabla f(\boldsymbol{x}_n) \rangle$
4:     Constant step: $\lambda_n = \frac{1}{n+1}$
5:     [Instead of constant step, use line search:
       $\lambda_n = \mathrm{argmin}_{\lambda \in [0,1]} f((1-\lambda)\boldsymbol{x}_n + \lambda \boldsymbol{s})$]
6:     Update $\boldsymbol{x}_{n+1} = (1-\lambda_n)\boldsymbol{x}_n + \lambda_n \boldsymbol{s}$
7: end for
```

by discrete particles, which has high computational efficiency and theoretical guarantee. The key idea is to perform discrete approximation by minimizing MMD, instead of KSD since it causes computational problems as we mentioned in the previous section. We minimize $\mathrm{MMD}(\mu_{\hat{p}})^2 = \frac{1}{2}\|\mu_p - \mu_{\hat{p}}\|^2_{\mathcal{H}}$, introduced by Eq. (2), in a greedy way. Since this is a convex function in an RKHS, we can use the FW algorithm.

The FW algorithm, also known as the conditional gradient method (Jaggi 2013), is a convex optimization method. It focuses on the problem $\min_{x \in \mathcal{D}} f(x)$, where f is a convex and continuous differentiable function and $\mathcal{D}$ is the domain of the problem, which is also convex. As the procedure is shown in Alg. 1, the FW algorithm optimizes the objective in a greedy way. In each step, we solve the linearization of the original f at the current state $\boldsymbol{x}_n$ as shown in Line 3 of Alg. 1. This step is often called the linear minimization oracle (LMO). The new state $\boldsymbol{x}_{n+1}$ is obtained by a convex combination of the previous state $\boldsymbol{x}_n$ and the solution of the LMO, $\boldsymbol{s}$, in Line 6 of Alg. 1. The common choice of the coefficient of the convex combination is the constant step or the line search.

Bach et al. (Bach, Lacoste-Julien, and Obozinski 2012) and Briol et al. (Briol et al. 2015) clarified the equivalence between kernel herding (Chen, Welling, and Smola 2010) and the FW algorithm for MMD. In our situation, we minimize MMD on the marginal polytope $\mathcal{M}$ of the RKHS $\mathcal{H}$, which is defined as the closure of the convex hull of $k(\cdot, x)$. We also assume that all sample points x_i are uniformly bounded in the RKHS, i.e., for any sample point x_i, $\exists r > 0 : \|k(\cdot, x)\|_{\mathcal{H}} \le r$.

By applying the FW algorithm, we want to obtain $\mu_{\hat{p}}$ which minimizes the objective $\mathrm{MMD}(\mu_{\hat{p}})^2 = \frac{1}{2}\|\mu_p - \mu_{\hat{p}}\|^2_{\mathcal{H}}$. We express the solution after n-steps FW algorithm as $\mu^n_{\hat{p}} = \sum_{i=1}^n w^n_i k(x, x_i)$, where $\{x_i\}_{i=1}^n$ are the particles and w^n_i denote the weights of the i-th particle at the n-th iteration. We can obtain $\{x_i\}_{i=1}^n$ in a greedy way by the FW algorithm. The method of deriving the weights are discussed later.

The LMO calculation in each step is $\mathrm{argmin}_{g \in \mathcal{M}} \langle \mu^n_{\hat{p}} - \mu_p, g \rangle$. It is known that the minimizer of a linear function in a convex set is one of the extreme points of the domain (Bach, Lacoste-Julien, and Obozinski 2012), and thus we derive

$$\begin{aligned}\arg\min_{g \in \mathcal{M}} \langle \mu^n_{\hat{p}} - \mu_p, g \rangle &= \arg\min_{x} \langle \mu^n_{\hat{p}} - \mu_p, k(\cdot, x) \rangle \\ &= \arg\min_{x} \sum_{i=1}^n w^n_i k(x_i, x) - \mu_p(x). \quad (3)\end{aligned}$$

We solve this LMO by gradient descent. We initialize each x to prepare $g = k(\cdot, x)$ in LMO by sampling it from the prior distribution. Since the objective of LMO is non-convex, we cannot obtain the global optimum by gradient descent in general. Fortunately, even if we solve LMO approximately, FW enables us to establish a finite sample convergence bound (Locatello et al. 2017a; Jaggi 2013; Lacoste-Julien et al. 2013; Lacoste-Julien and Jaggi 2015; Locatello et al. 2017b). In such an approximate LMO, we set the accuracy parameter $\delta \in (0, 1]$ and consider the following approximate problem which returns approximate minimizer $\tilde{g}$ of Eq.(3) instead of the original strict LMO:

$$\begin{aligned}\langle \mu^{(n)}_{\hat{p}} - \mu_p, \tilde{g} \rangle &= \delta \min_{g \in \mathcal{M}} \langle \mu^{(n)}_{\hat{p}} - \mu_p, g \rangle \\ &= \delta \min_x \sum_{i=1}^n w^n_i k(x_i, x) - \mu_p(x). \quad (4)\end{aligned}$$

This kind of relaxation of the LMO has been widely used and shown to be reliable (Locatello et al. 2017a; Jaggi 2013; Lacoste-Julien et al. 2013; Lacoste-Julien and Jaggi 2015; Locatello et al. 2017b), which is much easier to solve than the original strict LMO. We call this step Approx-LMO, and we will use gradient descent to solve Approx-LMO. The derivative with respect to x when we use the symmetric kernel k can be written as follows:

$$\begin{aligned}&\nabla_x \langle \mu^{(n)}_{\hat{p}} - \mu_p, g \rangle \\ &\approx \frac{1}{n} \sum_{i=1}^n w^{(n)}_i \left(\nabla_x k(x_i, x) + k(x, x_i) \nabla_{x_i} \ln p(x_i) \right). \quad (5)\end{aligned}$$

The derivation of Eq.(5) is given in Appendix. Using this gradient, we solve Eq.(4). As repeatedly pointed out (Locatello et al. 2017a; Jaggi 2013; Lacoste-Julien et al. 2013; Lacoste-Julien and Jaggi 2015; Locatello et al. 2017b), an approximate solution of the LMO is enough to assure the convergence which we describe later. For this reason, we will use gradient descent in our algorithm and also a rough estimate of the gradient is enough in our situation. A similar technique has also been discussed in (Locatello et al. 2017a).

For the FW algorithm, we have to specify the initial particle x_1 and the step size choice of the algorithm. We found that the initial particle x_1 by the MAP estimation or approximate MAP estimation shows good performance empirically and it is recommended to prepare x_1 as a near MAP point (we will discuss other choices later). In this approach, the constant step size and line search are not recommended because those methods uniformly reduce the weights of all the particles which has already been obtained. When we use x_1 as a near MAP point, it is located near the highest probability mass regions, and thus we should not reduce its weight uniformly. Based on this observation, we set the step size in the same way as the fully corrective Frank-Wolfe algorithm (Lacoste-Julien and Jaggi 2015), this method calculates all the weights at each iteration, and we can circumvent the above problem. For full correction, we use the Bayesian quadrature (BQ) weight (Huszár and Duvenaud 2012), $w_i = \sum_m z_m K^{-1}_{im}$, where K is the Gram matrix, $z_m = \int k(x, x_m) p(x) dx$, and we approximately compute the integral with particles. Since we use the empirical approximation, this makes the convergence rate slower. We will analyze the effect of this inexact step size later.

Algorithm 2: Approx-LMO

```
1: Input: $\mu_{\hat{p}}^{(n)}$
2: Output: $k(\cdot, x^{L+1})$
3: Prepare $g^0 = k(\cdot, x^0)$ where $x$ is initialized by
   randomly or sample from prior
4: for $l = 0 \dots L$ do
5:     Compute $\nabla_x \langle \mu_{\hat{p}}^{(n)} - \mu_p, g^l \rangle$ by Eq.(5)
6:     Update $x^{(l+1)} \leftarrow x^{(l)} + \epsilon^{(l)} \cdot \nabla_x \langle \mu_{\hat{p}}^{(n)} - \mu_p, g^i \rangle$
7: end for
```

Algorithm 3: MMD minimization by Frank-Wolfe algorithm (MMD-FW)

```
1: Input: A target density $p(x)$
2: Output: A set of particles $(\{w_i, x_i\}_{i=1}^N)$
3: Calculate approximate MAP estimation for $\mu_{\hat{p}}^{(1)}$
4: for $n = 2 \dots N$ do
5:     $k(\cdot, x_n)$ =Approx-LMO($\mu_{\hat{p}}^{(n-1)}$)
6:     Empirical BQ weight:
       $\hat{w}_i^n = \sum_{m=1}^n \hat{z}_m K_{im}^{-1}, \hat{z}_m = \sum_{l=1}^n k(x_l, x_m)/n$
7:     Update $\mu_{\hat{p}}^{(n+1)} = \sum_{i=1}^n \hat{w}_i^n k(x, x_i)$
8: end for
```

To summarize, our proposed algorithms are given in Alg. 2 and Alg. 3, which greedily increase the number of particles whithin the FW framework to minimize MMD.

Theoretical guarantee

First, we describe the condition to limit the deviation of empirically approximated BQ weights from the true ones so that the condition described below are satisfied. This is necessary for the theoretical guarantee of particle approximation.

Theorem 1 (Approximate step size) *In Alg. 3 at the n-th iteration, let β_i^n be the ratio between $\hat{z}_i^n$ and z_i^n, i.e., $\beta_i^n = \hat{z}_i^n / z_i^n$. When $\mathcal{H}$ is finite dimensional, if*

$$\int k(x,y)p(x)p(y)dxdy - \sum_{i,j=1}^{n} \beta_i^n \beta_j^n z_i^n K_{ij}^{-1} z_j^n > 0 \quad (6)$$

holds, then Theorems 2 and 3 hold. When $\mathcal{H}$ is infinite dimensional, no condition about the deviation of the weight is needed for Theorems 2 and 3 to hold.

In Eq.(6), since $\int k(x,y)p(x)p(y)dxdy$ is fixed and $\int k(x,y)p(x)p(y)dxdy - \sum_{i,j=1}^n z_i^n K_{ij}^{-1} z_j^n > 0$, β_i^n should be in some *moderate* range to satisfy the condition of Eq.(6). More intuitively, this condition states that if the deviation of the empirical estimate of BQ weights from the true ones is below a certain criterion, then convergence guarantee of the algorithm still holds even if the step size is inexact. The proof is given in Appendix. We also analyzed the effect of inexact step size in line search; see Appendix for details.

Next, we state the theoretical guarantee of our algorithm. We obtain $\hat{p}(x) = \sum_{n=1}^N w_n \delta(x, x_n)$ by Alg. 3 which approximates the true posterior $p(x)$. Let f be the test function, then we can bound the error $|Z_{f,p} - Z_{f,\hat{p}}| = |\int f(x)p(x)dx - \sum_{i=1}^N w_i f(x_i)|$ as follows:

Theorem 2 (Consistency) *Under the condition of Theorem 1, the error $|Z_{f,p} - Z_{f,\hat{p}}|$ of Alg. 3 is bounded at the following rate:*

$$\begin{aligned} &|Z_{f,p} - Z_{f,\hat{p}}| \\ &\leq \mathrm{MMD}(\{(w_n, x_n)\}_{n=1}^N) \\ &\leq \begin{cases} \sqrt{2} r e^{-\delta_{BQ} \frac{R^2 \delta^2 N}{2r^2}} & \textit{if } \mathcal{H} \textit{ is finite dimensional,} \\ \sqrt{\frac{(\delta_{BQ}\delta+1)2^2 r^2}{\delta(N\delta_{BQ}\delta+2)}} & \textit{if } \mathcal{H} \textit{ is infinite dimensional,} \end{cases} \end{aligned} \quad (7)$$

where r is the diameter of the marginal polytope $\mathcal{M}$, δ is the accuracy parameter of the LMO, and R is the radius of the smallest ball centered at μ_p included $\mathcal{M}$ (R is strictly above 0 only when the dimension of $\mathcal{H}$ is finite). δ_{BQ} denote the error caused by the empirical approximation of the BQ weights; for details, please see Appendix.

A proof of Theorem 2 can be found in Appendix. Moreover, on the basis of the Bayesian quadrature, we can regard $Z_{f,\hat{p}}$ as the posterior distribution of the Gaussian process (Huszár and Duvenaud 2012) (see Appendix for details) and assure the posterior contraction rate (Briol et al. 2015). Intuitively, the posterior contraction rate indicates how fast the probability of the estimated parameter residing outside a specified region (which includes the true parameter) decreases when the size of the region is increased.

Theorem 3 (Contraction) *Let $S \subseteq \mathbb{R}$ be an open neighborhood of the true integral $Z_{f,p}$ and let $\gamma = \inf_{r' \in S^c} |r' - Z_{f,p}| > 0$. Then the posterior probability on $S^c = \mathbb{R} \setminus S$ vanishes at the following rate:*

$$\begin{aligned} &\mathrm{prob}(S^c) \\ &\leq \begin{cases} \frac{2r}{\sqrt{\pi}\gamma} e^{-\delta_{BQ}\frac{R^2\delta^2 N}{2r^2} - \frac{\gamma^2}{4r^2} e^{\delta_{BQ}\frac{R^2\delta^2 N}{r^2}}} \\ \qquad \textit{if } \mathcal{H} \textit{ is finite dimensional,} \\ \sqrt{\frac{2}{\pi}} \sqrt{\frac{(\delta_{BQ}\delta+1)2^2r^2}{\delta(N\delta_{BQ}\delta+2)}} e^{-\frac{\gamma^2}{2}\frac{\delta(N\delta_{BQ}\delta+2)}{(\delta_{BQ}+\delta)r^2 2 r^2}} \\ \qquad \textit{if } \mathcal{H} \textit{ is infinite dimensional,} \end{cases} \end{aligned} \quad (8)$$

where r is the diameter of the marginal polytope $\mathcal{M}$, δ is the accuracy parameter, and R is the radius of the smallest ball centered at μ_p that includes $\mathcal{M}$. δ_{BQ} denotes the error caused by the empirical approximation of the BQ weights; for details, please see Appendix.

In the proposed method, kernel selection is crucial both numerically and theoretically. In the above convergence proof, linear convergence occurs only under the assumption that there exists a ball with centered at μ_p whose radius R is positive within the affine hull $\mathcal{M}$. Bach et al. (Bach, Lacoste-Julien, and Obozinski 2012) proved that, for infinite dimensional RKHSs, such as the case of radial basis function (RBF) kernels, such an assumption never holds. Thus, we can only have sub-linear convergence for RBF kernels in general. However, as pointed out by Briol et al. (Briol et al. 2015) , even if we use RBF kernels, thanks to finite-precision rounding error in computers, we are treating in simulations are actually essentially finite dimensional. This also holds in our situation, and in experiments, we empirically observed the linear convergence of our algorithm. We will show such a numerical result later.

A theory for the constant step size and line search is shown in Appendix.

Discussion

For specifying the initial particle x_1, we can sample it from the prior distribution. The merit of this approach is that we can choose the step size in a computationally less demanding way such as the constant step size and line search (shown in Appendix) since the initial particle is not in a high probability mass region, uniformly decreasing less important weights by constant step size or line search. However, we empirically found in our preliminary experiments that this initialization does not perform well compared to MAP initialization. We suspect that the gradient of Eq.(5) is too inexact when initial particles are sampled from the prior.

Let us analyze the reason why MAP initialization performs well as follows. Although the gradient is incorrect, the LMO can be solved with error to some extent because the first particle is close to the MAP estimation and the evaluation points of the expectation include, at least, a high density region on $p(x)$. If the LMO is δ-close to the true value, the weights of old incorrect particles will be updated to be small enough to be ignored as the algorithm proceeds. For such a reason, the framework using processed particles works.

The empirical approximation of the BQ weights can also be justified almost in the same way as above. Since the empirical distribution includes, at least, a high density region on $p(x)$, the deviation of the step size (e.g., error due to the empirical approximation) from the exact BQ weight is smaller than the criterion in Theorem 1.

In summary, since we prepare the initial particles at a high probability mass region, the FW algorithm successfully finds the next particle even though the gradient for LMO or weights are inexact. As the algorithm proceeds, the weights of less reliable particles become small and accuracy of the estimation is increased. This is an intuition how the proposed algorithm works.

Related works

In this section, we discuss the relationship between our method and SVGD, SP and variational boosting.

Relation to SVGD

SVGD is a method of optimizing a fixed number of particles simultaneously. On the other hand, MMD-FW is a greedy method adding new particles one per step. Both methods can work in high-dimensional problems since they use the information of the gradient of the score function. To approximate a high-dimensional target distribution, we may need many particles, but it is unclear how many particles are needed beforehand. Thus, a greedy approach is preferable for high-dimensional problems. Since in SVGD it is unclear how we can increase the number of particles after we finish the optimization, MMD-FW is more convenient in such a case. However, simultaneous optimization is sometimes computationally more efficient and show better performance compared to a greedy approach(See the experimental results).

Based on this fact, we combine SVGD and MMD-FW by focusing on the fact that the update equations of SVGD and MMD-FW are almost the same except for the weights. More specifically, we prepare particles by SVGD first, and then apply MMD-FW by treating particles obtained by SVGD as the initial state of each greedy particle. For details, please see Appendix. This combination enables us to enjoy the efficient simultaneous optimization of SVGD and the greedy property and theoretical guarantee of MMD-FW.

In terms of computation costs, SVGD is $\mathcal{O}(N^2)$ per iteration. In MMD-FW, we only optimize one particle, and thus, its computation cost is $\mathcal{O}(N)$ at each step inside Approx-LMO . Up to the N-th particle, the total cost is $\mathcal{O}(N(N+1)/2)$, which is in the same order as SVGD. However, the number of LMO iterations in MMD-FW is much smaller than that of SVGD since the problem involves only one particle in MMD-FW, which is much easier to solve than SVGD which treats N particles simultaneously. Therefore, we can expect the computation cost of MMD-FW to be cheaper than SVGD.

Relation to SP

The biggest difference between MMD-FW and SP is the objective function. Due to this difference, we use gradient descent to obtain new particles which is still computationally effective in high-dimensional problems. However, SP minimizes KSD, so we cannot use gradient descent since the calculation requires evaluations of the Hessian at each step, which is impossible in high-dimensional problems. To cope with this problem, SP uses sampling or grid search for optimization, which does not work in high-dimensional problems due to the curse of dimensionality. As we will see later, SP does not work well with complex models such as a Bayesian neural net.

Another difference is that our method can reliably use an approximate step size for the weights of particles. We have shown how the deviation of the approximate weights from the exact ones affects the convergence rate, which justified the use of our method even when the exact step size is unavailable.

Lastly, we use FW to establish a greedy algorithm. This enables us to utilize many useful variants of the FW algorithm such as a distributed variant(Wang et al. 2016). For details, see Appendix.

However, compared with SP, we cannot evaluate the objective function directly, so we resort to other performance measures such as the log likelihood, accuracy, or RMSE in test datasets. For SP, we can directly evaluate KSD at each iteration.

Relation to variational boosting

The proposed method is closely related to variational boosting (Locatello et al. 2017a). In (Locatello et al. 2017a), the authors analyzed the variational boosting by using the FW algorithm and showed the convergence to the target distribution. In variational boosting, a mixture of Gaussian distributions are used as an approximate posterior and its flexibility is increased the number of components in the mixture of Gaussian distributions. An intuition behind the convergence of variational boosting is that any distribution can be expressed by appropriately combining Gaussian mixture distributions. That situation is quite similar to MMD-FW, where we increase the number of particles greedily. In MMD-FW, we can regard each particle as being corresponding to each component of variational boosting. In both methods, the flexibility of the approximate posterior grows as we increase the number of components or particles and this allows us to establish the linear convergence under certain conditions. The difference is that we consider the solution in an RKHS and minimize MMD to approximate the posterior for MMD-FW, while variational boosting minimizes the KL divergence and treats the posterior in the parameter space.

Relation to kernel herding and Bayesian quadrature

In this paper, we are assuming that $p(x)$ is the posterior distribution. On the other hand, if $p(x)$ is a prior distribution, kernel herding (Chen, Welling, and Smola 2010) or Bayesian quadrature (Ghahramani and Rasmussen 2003), are useful. In those methods, x_n's are decided to directly minimize some criterions. For example, the kernel herding method (Chen, Welling, and Smola 2010; Bach, Lacoste-Julien, and Obozinski 2012) minimizes MMD in a greedy way. The biggest difference from our method is that if $p(x)$ is the prior distribution, we can sample many particles from $p(x)$ and thus we can only choose the best particle that decreases the objective function maximally at each iteration. In MMD-FW, on the other hand, we cannot prepare the particles beforehand, and thus, we directly derive particles by gradient descent.

Other related work

Recently, there has been a tendency to combine an approximation of the posterior with optimization methods, which assures us of some theoretical guarantee, e.g, (Locatello et al. 2017a; Dai et al. 2016). Our approach also performs discrete approximation by convex optimization in an RKHS. Another related example is sequential kernel herding(Lacoste-Julien, Lindsten, and Bach 2015). They applied the FW algorithm to particle filtering in state space models. While their method focused on the state space models, our proposed method is a general approximation method for Bayesian inference.

Numerical experiments

We experimentally confirmed the usefulness of the proposed method compared with SVGD and SP in both toy datasets and real world datasets. Other than comparing the performance measured in terms of the accuracy or RMSE of the proposed method with SVGD and SP, we also have the following two purposes for the experiments. The first purpose of the experiments is to confirm that our algorithm is faster than SVGD in terms of wall clock time. This is because, as mentioned before in the section of relation to SVGD, it solves simple problems compared with SVGD, thus we need less number of iterations to optimize each particle than that of SVGD. The second purpose is to confirm the convergence behavior.

In all experiments, we used the radial basis function kernel, $k(x, x') = \exp(-\frac{1}{2h^2}|x - x'|^2)$ for proposed method and SVGD, where h is the kernel bandwidth. The choice of h is critical to the success of the algorithms. There are three methods to specify the bandwidth, fixed bandwidth, median trick, gradient descent. We experimented on the above three choices and found that a fixed kernel bandwidth and the median trick are stable in general, and thus, we only show the results obtained by the median trick in this section. The results of other methods are shown with other detailed experimental settings in Appendix. For the kernel of SP, we used the three kernels proposed by the original paper (Chen et al. 2018): IMQ kernel $k_1(x, x') = (\alpha + ||x - x'||_2^2)^\beta$, inverse log kernel $k_2(x, x') = (\alpha + \log(1 + ||x - x'||_2^2))^{-1}$, and IMQ score kernel $k_3(x, x') = (\alpha + ||\nabla \log p(x) - \nabla \log p(x')||_2^2)^\beta$, where $\alpha = 1.0$ and $\beta = 0.5$ are used as suggested in the original paper.

For the approx-LMO, we used Adam (Kingma and Ba 2014) for all experiments. Due to space limitations, the toy data results are shown in Appendix. About the benchmark experiment, we split dataset 90% for training and 10% for testing.

Bayesian logistic regression

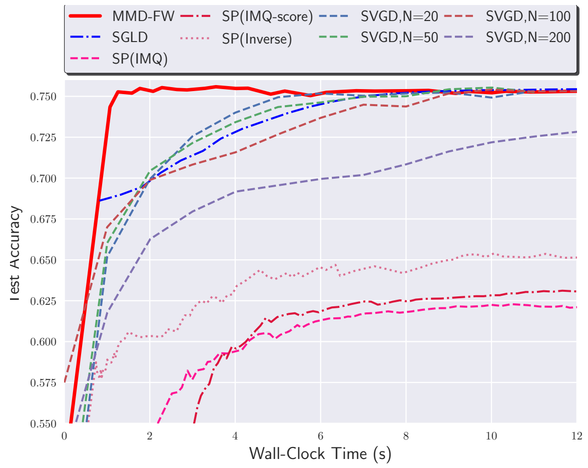
We considered Bayesian logistic regression for binary classification. The settings were the same as in those (Liu and Wang 2016), where we put a Gaussian prior $p_0(w|\alpha) = N(0, \alpha^{-1})$ for regression weights w and $p_0(\alpha) = \mathrm{Gamma}(1, 0.01)$. As the dataset, we used Covertype (Dheeru and Karra Taniskidou 2017), with 581,012 data points and 54 features. The posterior dimension is 56. The results are shown in Fig. 1. In Fig. 1(a), the vertical axis is the test accuracy and the horizontal axis is wall clock time. As we discussed in Section 4.1, our algorithm was faster than SVGD in terms of wall clock time. SP did not work well. We also compared MMD-FW with stochastic gradient Langevin dynamics(SGLD) (Welling and Teh 2011) and faster than SGLD. In Appendix, we also studied the situation where the first particle does not correspond to MAP estimation, and instead random initialization.

Fig. 1(b) shows the convergence behavior, where the vertical axis is MMD^2 and the horizontal one is the number of particles in the log scale. To calculate MMD, we generated "true samples" by Hamiltonian Monte Carlo (Neal and others 2011). Since RBF kernel is an infinite dimensional kernel, to further check the convergence behavior under the finite dimensional kernel, we approximated the RBF kernel by random Fourier expansion (RFF) (See Appendix for the details of the RFF). In Fig. 1(b), D is the number of frequency of RFF. Also, we still compared with SP on MMD although this comparison is a little unfair since the objective of SP is kernelized Stein discrepancy. As discussed in the previous section, although the convergence is sub-linear order theoretically since we used RBF kernel which is an infinite dimensional kernel, we observed the linear convergence thanks to the rounding error in the computer. The convergence speed of RBF kernel approximated by RFF showed the linear, which is the expected behavior since the approximated kernel by RFF is the finite dimensional kernel.

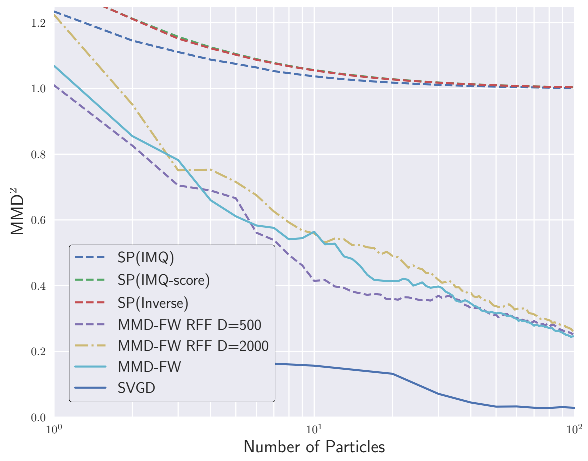
SVGD had a smaller MMD than the proposed method, which is due to the fact that SVGD simultaneously optimizes all particles and tries to put particles in the best position in

Table 1: Benchmark results on test RMSE and log likelihood by Bayesian neural net regression model

Dataset	**Posterior dimension**	Avg. Test RMSE		Avg. Test log likelihood		Fixed Wall clock Time (Secs)
		SVGD	Ours	SVGD	Ours	
Naval (N=11,934, D=17)	953	4.9e-4±7.5e-5	**4.2e-4±5.3e-5**	6.08 ± 0.11	**6.00±0.12**	150
Protein (N=45730, D=9)	553	4.51 ± 0.057	**4.43±0.035**	−2.93 ± 0.013	**-2.91±0.0073**	40
Year (N=515344, D=91)	9203	9.54 ± 0.08	**9.50±0.09**	**-3.65±0.005**	**-3.65±0.011**	300

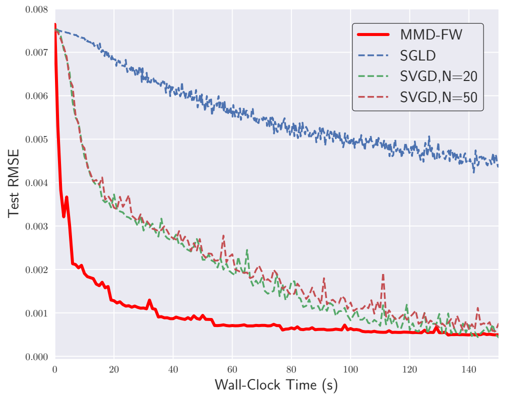


(a) Comparison of MMD-FW and SVGD in terms of wall clock time with the test accuracy

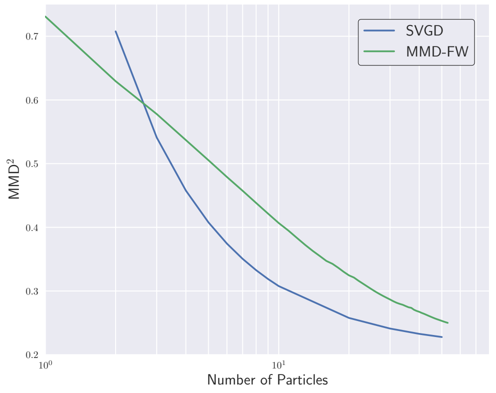


(b) Convergence behavior in terms of number of the particles with MMD^2

Figure 1: Comparison for the logistic regression model



(a) Comparison of MMD-FW and SVGD in terms of wall clock time with test RMSE



(b) Convergence behavior in terms of the number of the particles with MMD^2

Figure 2: Comparison for the Bayesian neural net regression model

correspondence with the global optima. In contrast, MMD-FW only increased the particles greedily, and this resulted in local optima. Hence, the better performance of SVGD compared with MMD-FW with the same number of particles in terms of MMD is a natural result.

Bayesian neural net regression

We experimented with Bayesian neural networks for regression. The settings were the same as those in (Liu and Wang 2016). We used a neural network with one hidden layer, 50 units, and the ReLU activation function. As the dataset, we used the Naval data from the UCI (Dheeru and Karra Taniskidou 2017), which contains 11,934 data points and 17 features. The posterior dimension was 953. The results are shown in Fig. 2. In Fig. 2a, the vertical axis is the test RMSE, and the horizontal axis is wall clock time. In Fig 2b, the vertical axis is the MMD^2, and the horizontal axis is the number of particles. Since it is difficult to prepare MAP initialization for Bayesian neural networks at first in MMD-FW, we consider non-MAP initialization, and we gradually reduced earlier weight sizes by adjusting the step size. The posterior dimension was much higher than that of the logistic regression, but our algorithm was faster than SVGD in terms of wall clock time and linearly converged, which is consistent with the theory.

Results for other datasets are shown in Table 1, where we fixed the wall clock time and applied MMD-FW and SVGD within that period. SP did not work well because of the high dimensionality so its results are not shown. We experimented 5 random trials for changing the splitting of the dataset. For the Protein data, we used the same model as the Naval data, and for the Year data, we used the same model as others except that the number of hidden units is 100. From these benchmark dataset experiments, we confirmed that our method shows almost the same performance as SVGD in many cases but shows faster optimization. Moreover, it shows linear convergence.

Conclusions

In this work, we proposed MMD-FW, a novel approximation method for posterior distributions. Our method enjoys empirically good performance and theoretical guarantee simultaneously. In practice, our algorithm is faster than existing methods in terms of wall clock time and works well even in high-dimensional problems. As future work, we will further apply this framework other than the posterior approximation and further analyze the effect of rounding error on convergence rate.

Acknowledgment

FF was supported by JST AIP-PRISM Grant Number JPMJCR18ZH Japan and Google Fellowship, IS was supported by KAKENHI 17H04693, and MS was supported by KAKENHI 17H00757.

References

Bach, F.; Lacoste-Julien, S.; and Obozinski, G. 2012. On the equivalence between herding and conditional gradient algorithms. In *Proceedings of the International Conference on Machine Learning*, 1355–1362.

Bishop, C. M. 2006. *Pattern Recognition and Machine Learning*. Springer.

Blei, D. M.; Kucukelbir, A.; and McAuliffe, J. D. 2017. Variational inference: A review for statisticians. *Journal of the American Statistical Association* 112(518):859–877.

Briol, F.-X.; Oates, C.; Girolami, M.; and Osborne, M. A. 2015. Frank-Wolfe Bayesian quadrature: Probabilistic integration with theoretical guarantees. In *Advances in Neural Information Processing Systems*, 1162–1170.

Chen, W. Y.; Mackey, L.; Gorham, J.; Briol, F.-X.; and Oates, C. J. 2018. Stein points. *arXiv preprint arXiv:1803.10161*.

Chen, Y.; Welling, M.; and Smola, A. 2010. Super-samples from kernel herding. In *Proceedings of the Twenty-Sixth Conference on Uncertainty in Artificial Intelligence*, 109–116. AUAI Press.

Dai, B.; He, N.; Dai, H.; and Song, L. 2016. Provable Bayesian inference via particle mirror descent. In *Proceedings of the International Conference on Artificial Intelligence and Statistics*, 985–994.

Dheeru, D., and Karra Taniskidou, E. 2017. UCI machine learning repository. http://archive.ics.uci.edu/ml.

Ghahramani, Z., and Rasmussen, C. E. 2003. Bayesian Monte Carlo. In *Advances in Neural Information Processing Systems*, 505–512.

Gretton, A.; Borgwardt, K. M.; Rasch, M. J.; Schölkopf, B.; and Smola, A. 2012. A kernel two-sample test. *Journal of Machine Learning Research* 13(Mar):723–773.

Huszár, F., and Duvenaud, D. 2012. Optimally-weighted herding is Bayesian quadrature. In *Proceedings of the Twenty-Eighth Conference on Uncertainty in Artificial Intelligence*, 377–386. AUAI Press.

Jaggi, M. 2013. Revisiting Frank-Wolfe: Projection-free sparse convex optimization. In *Proceedings of the International Conference on Machiine Learning*, 427–435.

Kingma, D. P., and Ba, J. 2014. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*.

Lacoste-Julien, S., and Jaggi, M. 2015. On the global linear convergence of Frank-Wolfe optimization variants. In *Advances in Neural Information Processing Systems*, 496–504.

Lacoste-Julien, S.; Jaggi, M.; Schmidt, M.; and Pletscher, P. 2013. Block-coordinate Frank-Wolfe optimization for structural SVMs. In *Proceedings of International Conference on Machine Learning*, 53–61.

Lacoste-Julien, S.; Lindsten, F.; and Bach, F. 2015. Sequential kernel herding: Frank-Wolfe optimization for particle filtering. *arXiv preprint arXiv:1501.02056*.

Liu, Q., and Wang, D. 2016. Stein variational gradient descent: A general purpose Bayesian inference algorithm. In *Advances In Neural Information Processing Systems*, 2378–2386.

Liu, Q.; Lee, J.; and Jordan, M. 2016. A kernelized stein discrepancy for goodness-of-fit tests. In *Proceedings of International Conference on Machine Learning*, 276–284.

Liu, Q. 2017. Stein variational gradient descent as gradient flow. In *Advances in Neural Information Processing Systems*, 3118–3126.

Locatello, F.; Khanna, R.; Ghosh, J.; and Rätsch, G. 2017a. Boosting variational inference: an optimization perspective. *arXiv preprint arXiv:1708.01733*.

Locatello, F.; Khanna, R.; Tschannen, M.; and Jaggi, M. 2017b. A unified optimization view on generalized matching pursuit and Frank-Wolfe. In *Proceedings of the 20th International Conference on Artificial Intelligence and Statistics*, 860–868.

Neal, R. M., et al. 2011. MCMC using Hamiltonian dynamics. In *Handbook of Markov Chain Monte Carlo*. CRC Press New York, NY. 113–162.

Wang, Y.-X.; Sadhanala, V.; Dai, W.; Neiswanger, W.; Sra, S.; and Xing, E. 2016. Parallel and distributed block-coordinate frank-wolfe algorithms. In *International Conference on Machine Learning*, 1548–1557.

Welling, M., and Teh, Y. W. 2011. Bayesian learning via stochastic gradient Langevin dynamics. In *Proceedings of the 28th International Conference on Machine Learning*, 681–688.