ESPs: a new cost-efficient sampler for expensive posterior distributions

July 18, 2022

Abstract

Keywords:

1 Introduction

(Simon: See rough skeleton of intro below - expand on for writing. Irene: feel free to help with a bit of the writing, esp on motivation.)

First paragraph: Why is the problem of expensive posterior sampling important?

- Bayesian inverse problems: each posterior evaluation requires a forward run of a simulation system
- These systems solve complex scientific systems and can take (Irene: $\mathcal{O}(10^3)$ CPU hours for a single set of input parameters)
- Take, for example, the Bayesian parameter estimation problem in JETSCAPE (brief intro). Each run costs millions of CPU hours per run, thus we can only afford a few hundred evaluations of the posterior at most!
- The goal is therefore to obtain as accurate a representation of the posterior distribution as possible, given limited evaluations from a tight computational constraint.
- (Simon: Irene can help with this)

A fundamental task of inverse problems is to infer unknown parameters that govern the physical processes of interest. In Bayesian context, this often involves sampling from an unnormalized posterior distribution of the parameters given observed data. With improvements in computational efficiency, computer models have been increasingly used to simulate these physical processes. Thus, each posterior evaluation often requires a forward run of the simulation system. However, as the physical processes become more sophisticated, each simulation run could demand $O(10^3)$ CPU hours for a single set of input parameters, which makes posterior sampling a prohibitively expensive task. (JETSCAPE Problem) Therefore, given limited evaluations from a tight computational constraint, the key challenge

is to obtain efficient samples that can represent the posterior distribution as accurate as possible. The efficiency has two folds: (i) sample efficiency, which means that fewer samples are required to represent the posterior well; (ii) cost efficiency, which means that generating one sample requires as few posterior evaluations as possible.

Second paragraph: What are standard / state-of-the-art MCMC methods? Why might they not be appropriate for expensive posterior sampling problems?

- Metropolis-Hastings samplers
- Hamiltonian Monte Carlo
- MALA, Riemannian HMC, etc...
- Key limitations: (i) such methods are typically less efficient than Monte Carlo (unsurprising since we do not know normalizing constants). For complex problems, one often experiences high correlations in the sample chain, meaning many many samples are needed to achieve the desired precision (ii) for these methods, each sample requires at least one (and often more) evaluations of the posterior! so combined with (i), very expensive, (iii) tuning good parameters also requires many evaluations as well.
- (Simon: Irene can look over & add stuff after)

Markov chain Monte Carlo (MCMC) is a class of sampling methods that have been widely applied in Bayesian inference problems. The general idea is to construct a Markov chain that has the posterior distribution as its stationary distribution and samples are obtained from the successive states of the Markov chain. The starting state of the Markov chain is usually chosen at random. A transition kernel, or proposal distribution, is then used at each iteration to suggest a candidate for the next sample value given the previous sample. The posterior density of the candidate will be evaluated to compute a criterion that decides whether the

candidate should be accepted or rejected. The Metropolis-Hasting algorithm is a basic MCMC sampler from which many variations are developed, such as Metropolis-Adjusted Langevin Algorithm (MALA), Hamiltonian Monte Carlo (HMC), Riemann Manifold Hamiltonian Monte Carlo. These variations follow the aforementioned general framework, each with a different method to adaptively choose the proposal distribution so that samples converge to the posterior distribution in a faster rate.

However, several limitations make MCMC methods undesirable for expensive posterior sampling problems. (i) MCMC samples are typically less efficient than independent samples generated by Monte Carlo algorithms because they are positively correlated. In complex problems where sample chains are highly dependent, an enormous number of samples are required to achieve a reasonable approximation of the posterior. Moreover, since it takes some time for the Markov chain to move from the starting state to the stationary distribution, the first hundreds or thousands of samples, called "burn-in" period, are usually discarded. While the optimal acceptance ratio of an MCMC sample chain is still an open question, a rate between 20% to 40% is usually chosen in practice, which suggests that over 50% of posterior evaluations are wasted. (ii) For MCMC methods, each sample requires at least one evaluation of the posterior density. In algorithms like MALA and HMC where proposal distributions are adaptively tuned, much more evaluations are required at each iteration to find the right proposal. Combined with (i), generating MCMC samples that reasonably approximate an expensive posterior distribution can be computationally infeasible. (iii) Many MCMC algorithms require specification of some hyperparameters, such as the leapfrog step size in HMC. Tuning good hyperparameters for improved efficiency also demands additional posterior evaluations.

Quasi-Monte Carlo (QMC) is a deterministic sampling algorithm that can alleviate the issue of sample inefficiency in MCMC methods.

Third paragraph: Brief intro to Quasi Monte Carlo, recent work on integrating QMC

for posterior sampling. Why might these methods be not appropriate / suboptimal for our problem?

- In the case of Monte Carlo on the unit hypercube, one way to improve sample efficiency is QMC. In essence, this aims to place points as uniformly as possible over the sample space, which leads to improved efficiency for integration. See paragraph in PQMC paper.
- Recent interesting work on extending such methods for Bayesian sampling: Stein points, Stein variational gradient descent, MED, etc.
- What are limitations?

Fourth paragraph: what is our method? what is novel about it? how does this address the aforementioned limitations, and provide a satisfying solution for expensive posterior sampling? what do we present in the paper?

This paper is organized as follows. (Simon: fill in)

2 Background & Motivation

We first provide a brief overview of Minimum energy design (MED) and Stein points.

2.1 Minimum energy design

According to Joseph et al. (2015), a set of deterministic points $S = \{x_i\}_{i=1}^n$ from a posterior distribution $P(x)$, where each $x_i \in \mathbb{R}^d$, is called a minimum energy design (MED) if it minimizes the total potential energy $E(S)$ given by

$$E(S) = \sum_{i \neq j} \frac{q(x_i)q(x_j)}{d(x_i, x_j)},$$

where $q(x)$ is called a charge function and $d(x_i, x_j)$ is the Euclidean distance between points x_i and x_j. Joseph et al. (2015) also proposed a generalized version of MED

$$\min_S \mathrm{GE}_k = \left\{ \sum_{i \neq j} \left(\frac{q(x_i)q(x_j)}{d(x_i, x_j)} \right)^k \right\}^{1/k}$$

for $k \in [1, \infty]$ and showed that as $k \to \infty$, this criterion converges to

$$\max_S \min_{i,j} \frac{d(x_i, x_j)}{q(x_i)q(x_j)}.$$

Since the limiting distribution of MED is given by $1/q^{2d}(x)$, Joseph et al. (2015) chose the charge function to be $q(x) = \frac{1}{\{P(x)\}^{1/(2d)}}$ in order to obtain a deterministic sample from the desired posterior density $P(x)$. Thus, given a posterior distribution $P(x)$, the MED criterion can be written as

$$\max_S \min_{i \neq j} P^{1/(2d)}(x_i) P^{1/(2d)}(x_j) d(x_i, x_j). \tag{1}$$

Note that we only need to know $P(x)$ up to a constant because the constant does not affect the optimization. Joseph et al. (2015) proposed a one-point-at-a-time greedy algorithm to generate MED, which requires extensive evaluations of the density function $P(x)$. In Joseph et al. (2019), a surrogate-based optimization approach reduces the number of density function evaluations to Kn, where K is the number of steps to anneal $P(x)$ and n is the number of generated samples. This improvement makes MED more suitable for expensive posterior problems. (why need to introduce SP, how they are related..?)

(2 paragraphs – discuss the optimization of 2019 paper, more details on the algorithm – how they make it cost-efficient)

(limitation of MED)

2.2 Stein points

Similar to MED where the optimization of criterion (1) is used to construct an optimal sample set, Stein points define the optimality of a sample set for a posterior distribution $P(x)$ through kernel Stein discrepancy (KSD). Given a sample set $S = \{x_i\}_{i=1}^n$ from a posterior distribution $P(x)$, a discrepancy measures how well S approximates $P(x)$. A popular discrepancy called integral probability metric (IPM) (Muller, 1997) is defined as

$$D_{\mathcal{F},P}(\{x\}_{i=1}^n) := \sup_{f\in\mathcal{F}} |\frac{1}{n}\sum_{i=1}^n f(x_i) - \int_{\mathcal{X}} f dP| \tag{2}$$

where $\mathcal{F}$ is a set of measure-determining functions on a measurable space $\mathcal{X}$ such that $S \subset \mathcal{X}$. The problem with this discrepancy is that $\int_{\mathcal{X}} f dP$ requires an exact integration with respect to $P(x)$, but $P(x)$ is usually known up to a constant in most Bayesian context. To construct a computationally tractable form of IPM, Gorham and Mackey (2015) proposed Stein discrepancy based on Stein's method (Stein, 1972), which consists of finding a function class $\mathcal{G}$ and an operator $\mathcal{A}_P$, which depends on $P(x)$ and acts on functions $\mathcal{G}$ such that $\int_{\mathcal{X}} \mathcal{A}_P g dP = 0$ for all $g \in \mathcal{G}$. Taking $\mathcal{F} = \mathcal{A}_P\mathcal{G}$ in (2), Gorham and Mackey (2015) defined the Stein discrepancy as

$$D_{\mathcal{A}_P\mathcal{G},P}(\{x_i\}_{i=1}^n) := \sup_{g\in\mathcal{G}} |\frac{1}{n}\sum_{i=1}^n \mathcal{A}_P g(x_i)|. \tag{3}$$

Kernel Stein discrepancy is a special form of (3) that can be easily computed. Let $\mathcal{H}(k)$ denote a reproducing kernel Hilbert space (RKHS) on a set $\mathcal{X}$ with a kernel function $k : \mathcal{X}\times\mathcal{X} \to \mathbb{R}$ such that $\forall x \in \mathcal{X}$, we have $k(x,\cdot) \in \mathcal{H}(k)$ and $\forall x \in \mathcal{X}$ and $\forall h \in \mathcal{H}(k)$, we have $h(x) = \langle h, k(\cdot, x)\rangle_{\mathcal{H}(k)}$. Liu et al. (2016); Chwialkowski et al. (2016); Gorham and Mackey (2017) showed that if we take $\mathcal{A}_P$ to be the Langevin operator $\mathcal{A}_P g := \nabla_x(g(x)P(x))/P(x)$ and the class $\mathcal{G} := \{g \in \mathcal{H}(k) : ||g||_{\mathcal{H}(k)} \leq 1\}$ to be the unit ball in the RKHS $\mathcal{H}(k)$, then

$\mathcal{A}_P\mathcal{G}$ is the unit ball of another RKHS $\mathcal{H}(k_0)$ with kernel k_0 given by

$$\begin{aligned} k_0(x, x') = &\nabla_x \cdot \nabla_{x'} k(x, x') + \nabla_x k(x, x') + \nabla_{x'} \log P(x') \\ &+ \nabla_{x'} k(x, x') \cdot \nabla_x \log P(x) + k(x, x') \nabla_x \log P(x) \nabla_{x'} \log P(x'). \end{aligned} \tag{4}$$

With this choice of $\mathcal{A}_P$ and $\mathcal{G}$, it follows that (3) can be viewed as a maximum mean discrepancy (Gretton et al., 2006) in RKHS $\mathcal{H}(k_0)$ and can be explicitly computed by

$$D_{k_0,P}(\{x_i\}_{i=1}^n) := \sqrt{\frac{1}{n^2} \sum_{i,j=1}^{n} k_0(x_i, x_j)}. \tag{5}$$

Stein points (Chen et al., 2018) select a sample set $\{x_i\}_{i=1}^n$ from a posterior distribution $P(x)$ through sequential minimization of kernel Stein discrepancy defined in (5). Different sequential strategies were considered, but since the greedy algorithm performs the best, we mainly focus on the greedy Stein points in this paper. The algorithm is as follows: select the first point x_1 to be the global maximum of $P(x)$; at iteration $n > 1$, x_n is selected such that

$$x_n \in \operatorname*{argmin}_{x \in \mathcal{X}} \frac{k_0(x, x)}{2} + \sum_{i=1}^{n-1} k_0(x_i, x). \tag{6}$$

Chen et al. (2018) proposed three choices of base kernel k used in (4) and three numerical optimization techniques to find x_n in (6). Since the inverse multiquadric (IMQ) kernel $k(x, x') = (\alpha + ||x = x'||_2^2)^\beta$ and the Monte Carlo optimization method were observed to result in the best sample set as quantified by the Wasserstein distance between $\{x_i\}_{i=1}^n$ and $P(x)$ given fewest density evaluations, we choose them in the Stein points implementation for comparison with ESP.

One limitation of Stein points is that .... (emphasize that proposal points are random in optimization so many are wasted) (need to show SP MC optimization algorithm)

(MED uses a different approach to tackle the cost-efficiency problem – introduce MED,

limitation)

2.3 Motivating Example

3 Methodology

3.1 Gaussian process & Bayesian optimization

Let us first have a brief overview of Gaussian process model (GP) and how it is utilized in Bayesian optimization technique. Let $X_n = \{x_i\}_{i=1}^n$ be the input points with corresponding output values $Y_n = \{y_i\}_{i=1}^n$. Let $f : \mathcal{X} \to \mathcal{Y}$ denote an unknown function that maps an input x to an output y. A Gaussian process model assumes the following prior on the function $f(x)$:

$$f(x) \sim \mathcal{GP}(\mu(x), \Sigma(x, x')) \tag{7}$$

where $\mu(x)$ is the mean function that reflects the expected function value at x and $\Sigma(x, x')$ is the covariance function that models the dependence of function values at different input points x and x'. Without prior knowledge, the prior mean function $\mu(x)$ is often set to a constant μ and the covariance function is chosen to be the squared exponential kernel or the Matérn kernel. With observed training data $D_n = (X_n, Y_n)$, the predictive distribution at a new input location x_{new} is given by

$$f(x_{new})|D_n \sim \mathcal{N}(\tilde{\mu}(x_{new}), \tilde{\sigma}^2(x_{new})) \tag{8}$$

$$\tilde{\mu}(x_{new}) = \mu(x_{new}) + \Sigma(x_{new}, X_n)^T \Sigma(X_n, X_n)^{-1}(Y_n - \mu(X_n)\mathbf{1}) \tag{9}$$

$$\tilde{\sigma}^2(x_{new}) = \Sigma(x_{new}, x_{new}) - \Sigma(x_{new}, X_n)^T \Sigma(X_n, X_n)^{-1}\Sigma(x_{new}, X_n) \tag{10}$$

One advantage of GP is that the closed-form predictive distribution not only provides an estimated function value $\tilde{\mu}(x_{new})$, but also quantifies the uncertainty of the prediction

through $\tilde{\sigma}^2(x_{new})$, which is highest at locations furthest away from the training points X_n.

GP models have been widely used in Bayesian optimization (BO), an efficient method to optimize an expensive blackbox objective function globally. Suppose we want to find an input x^* the minimizes the objective function $F(x)$ in a bounded domain $x \in \mathcal{X} \subset \mathbb{R}^d$:

$$x^* = \underset{x \in \mathcal{X}}{\operatorname{argmin}} F(x).$$

Popular algorithms such as gradient descent and BFGS rely on exact expression or approximation to the gradient or Hessian matrix of $F(x)$. They require a large number of function evaluations to converge and can be computationally prohibitive when $F(x)$ is expensive. Compared to these methods, BO usually requires fewer function evaluations to achieve comparable performance. The idea is simple: given a set of input locations $X_n = \{x_i\}_{i=1}^n$ and the corresponding objective values $Y_n = \{y_i = F(x_i)\}_{i=1}^n$, (1) BO fits a GP predictive model $f(x)$ on $D_n = (X_n, Y_n)$; (2) the next point x_{new} where $F(x)$ should be evaluated is selected by optimizing an acquisition function $\mathcal{A}(x; f)$, that is, $x_{new} = \operatorname{argmax}_{x \in \mathcal{X}} \mathcal{A}(x; f)$; (3) BO updates $f(x)$ with x_{new} and $y_{new} = F(x_{new})$. This process repeats until convergence or a fixed budget of function evaluations.

(need to compared with SP, indicate that proposal points are guided by previous knowledge)

The acquisition function $\mathcal{A} : \mathcal{X} \to \mathbb{R}$ maps a new input location x_{new} to a value describing how much contribution sampling at x_{new} will make to finding the true minimum. A meaningful acquisition function should balance between exploitation, which is to favor x_{new} with low prediction $f(x_{new})$, and exploration, which is to favor x_{new} with high predictive uncertainty. Due to this property, the GP model is the most common predictive model in BO as it provides closed-form prediction and uncertainty quantification. There are a wide range of acquisition functions, the choice of which can greatly affect the performance of

BO. Several popular choices include expected improvement (EI), Thompson sampling (TS), entropy search (ES), and upper confidence bounds (UCB). Though EI is more greedy than other methods and tends to over-exploit near the input point that gives the current minimum, it is more computationally tractable than TS and ES, which can only be approximated when the domain space $\mathcal{X}$ is continuous, and it does not require specification of additional hyperparameters as UCB does. Therefore, we will use EI as the acquisition function in the ESP algorithm.

The expected improvement at a new input location x_{new} can be computed as follows. Let $y^n_{\min} = \min\{y_i\}_{i=1}^n$ denote the current minimum of objective values at known input locations $\{x_i\}_{i=1}^n$. The potential for improvement over $y^n_{\min}$ at x_{new} is defined by

$$I(x_{new}) = \max(0, y^n_{\min} - f(x_{new})).$$

Thus, $I(x_{new})$ measures how much the objective value at x_{new} could be below the current minimum. Here, $f(x_{new})$ is shorthand for $f(x_{new})|D_n$, the predictive distribution as defined in (8). Since $f(x_{new})$ is a random variable, $I(x_{new})$ is also a random variable. The expected improvement at x_{new} is thus given by $EI(x_{new}) = \mathbb{E}[\max(0, y^n_{\min} - f(x_{new}))]$. When $f(x_{new})$ is Gaussian, $EI(x_{new})$ can be shown to have the following analytic form:

$$EI(x_{new}) := (y^n_{\min} - \tilde{\mu}(x_{new}))\Phi\left(\frac{y^n_{\min} - \hat{\mu}(x_{new})}{\tilde{\sigma}(x_{new})}\right) + \tilde{\sigma}(x_{new})\phi\left(\frac{y^n_{\min} - \hat{\mu}(x_{new})}{\tilde{\sigma}(x_{new})}\right) \quad (11)$$

where Φ is a standard Gaussian CDF and ϕ is a standard Gaussian PDF. The first term in $EI(x_{new})$ represents the goal of exploitation: it is large when $\tilde{\mu}(x_new)$ is much below $y^n_{\min}$. The second term embodies the goal of exploration: it is large when the predictive uncertainty $\tilde{\sigma}(x_{new})$ is high.

3.2 Cost-efficient Stein points

We now integrate the aforementioned Bayesian optimization technique into the framework of Stein points. Let $P(x)$ denote the target distribution that is known up to a constant. Suppose we have taken $n-1$ samples from $P(x)$. At iteration n, the objective function is given by

$$F_{KSD}(\cdot) = \frac{k_0(\cdot,\cdot)}{2} + \sum_{i=1}^{n-1} k_0(x_i,\cdot)$$

with $k_0(\cdot,\cdot)$ defined in (4). Let t denote the maximal number of objective function evaluations we can afford at iteration n, $X_{prev} = (x_j)_{j=1}^m$ denote all the proposal points in previous iterations that are not selected as samples, and $S_{prev} = (\nabla_x P(x_j))_{j=1}^m$ denote the score values of these proposal points.

Algorithm 1 BO Algorithm

Input: KSD function $KSD(x)$, desired sample size N,
Require: $n \geq 0$
Ensure: $y = x^n$
  $y \leftarrow 1$
  $X \leftarrow x$
  $N \leftarrow n$
  while $N \neq 0$ **do**
    if N is even **then**
      $X \leftarrow X \times X$
      $N \leftarrow \frac{N}{2}$ ▷ This is a comment
    else if N is odd **then**
      $y \leftarrow y \times X$
      $N \leftarrow N - 1$

4 Numerical Experiments

4.1 Illustrative example

toy 1d example – ci, ess (integrate 1d results to background + motivation / compare sp-mcmc / sp sample wasserstein distances to normal, tangible example to show the problem of current method)

4.2 Mixture normal distribution

4.3 Banana distribution

numerical: illustrative 1d + 2d 2d gaussian mixtures organize the results – ci, ess,

harder distribution: higher dim gaussian mixtures (5, 18, 50) other distribution – banana distribution?

section 2: subsection stein points, sp-mcmc, the problem/limitation of inference (why need some things better than sp)

2d: ess, wasserstein distance, test functions from r2 to r1 and compute the ci for this function

5 Applications

5.1 Wiffle ball drop

See `https://bookdown.org/rbg/surrogates/chap8.html`. We can add air resistance for more realistic calibration

5.2 CFD application

Xingjian: feel free to add thoughts here.

6 Conclusion

References