Journal of Computational Physics 321 (2016) 1098–1109

Contents lists available at ScienceDirect

Journal of Computational Physics

www.elsevier.com/locate/jcp

A surrogate accelerated multicanonical Monte Carlo method for uncertainty quantification

Keyi Wu [a], Jinglai Li [b,*]

[a] Department of Mathematics, Zhiyuan College, Shanghai Jiao Tong University, Shanghai 200240, China
[b] Institute of Natural Sciences, Department of Mathematics, and MOE Key Laboratory of Scientific and Engineering Computing, Shanghai Jiao Tong University, Shanghai 200240, China

ARTICLE INFO

Article history:
Received 26 August 2015
Received in revised form 12 April 2016
Accepted 13 June 2016
Available online 17 June 2016

Keywords:
Gaussian processes
Multicanonical Monte Carlo
Uncertainty quantification

ABSTRACT

In this work we consider a class of uncertainty quantification problems where the system performance or reliability is characterized by a scalar parameter y. The performance parameter y is random due to the presence of various sources of uncertainty in the system, and our goal is to estimate the probability density function (PDF) of y. We propose to use the multicanonical Monte Carlo (MMC) method, a special type of adaptive importance sampling algorithms, to compute the PDF of interest. Moreover, we develop an adaptive algorithm to construct local Gaussian process surrogates to further accelerate the MMC iterations. With numerical examples we demonstrate that the proposed method can achieve several orders of magnitudes of speedup over the standard Monte Carlo methods.

1. Introduction

Uncertainty is an inevitable feature of real-world engineering systems. In those systems uncertainty can rise from various of sources: material properties, geometric parameters, boundary conditions, applied loadings and so on. In practice, it is essentially important to characterize and quantify the impact of the uncertainties on the system performances, which constitutes a central task of the newly emerging field of Uncertainty Quantification (UQ). To be specific, we consider the UQ problems in the following setting. We assume that the system is (formally) characterized by a performance function $y = g(\mathbf{x})$, where the input $\mathbf{x}$ is a random vector collecting all the uncertain factors in the system and y is a scalar indicating the system performance or reliability (in what follows, we will simply refer to y as the performance parameter). A typical example is the structural design problems, in which y can be the stress or the deformation. In this setting, the key task is to accurately assess and quantify the uncertainty in the performance parameter y. A challenge here is that real-world applications demand various statistical information of the performance y: for example, in robust design, the interests are mainly in the lower moments, especially the mean and the variance [12], in reliability analysis, it is mainly the tail probability [19], in risk management, one can be interested in the tail probability as well as some extreme quantiles [20], and in utility optimization, the complete distribution of the performance parameter is required [13]. To this end, a unified solution is to acquire the knowledge of the probability distribution of the performance parameter, which provides a complete characterization of the uncertainty in it. In theory, the distribution of y can be estimated by crucial Monte Carlo (MC) simulations, provided that a sufficient number of samples can be afforded. In reality, however, the function $g : \mathbf{x} \to y$ generally

[*] Corresponding author.
E-mail addresses: wukeyi@sjtu.edu.cn (K. Wu), jinglaili@sjtu.edu.cn (J. Li).

http://dx.doi.org/10.1016/j.jcp.2016.06.020

admits no analytical form, and evaluating function $g(\mathbf{x})$ must be done by performing computer simulation of the underlying system, which renders estimating the distribution of y with crucial MC impractical.

The main purpose of this work is to provide an efficient method to compute the full distribution of y. The proposed method has two major ingredients. First, we propose to sample the distribution of y with the multicanonical Monte Carlo (MMC) method, which can be regarded as a more efficient alternative to MC. The MMC method was initially developed by Berg and Neuhaus [5,6] to explore the energy landscape of a given physical system, and later it has been adopted to simulate rare events, such as transmission errors in optical communication systems [14,23], and the rare growth factors in random matrices [11]. Roughly speaking, the MMC method constructs an iterative procedure that generates samples forming a flat histogram in the space of the parameter of interest (i.e., the energy in the original problem setup). As will be shown in Section 2, the MMC method often requires to iterate many times and in each iteration it employs Markov chain Monte Carlo (MCMC) simulations to draw a rather large number of samples. As a result, the direct use of MMC to sample the distribution of the performance parameter can still be computationally demanding, especially for systems with computationally intensive models. To this end, the second major component of our method is to employ computationally inexpensive surrogates to further reduce the computational cost of MMC. In particular, building on the method developed in the work [10], we adaptively construct local Gaussian process (GP) surrogates in the MCMC iteration. We choose to use this method for the following reasons: first, the surrogate construction scheme is naturally incorporated in the MCMC iterations, which makes it convenient to use; secondly, unlike many other surrogate based algorithms which introduce errors in the equilibrium distribution, this method samples asymptotically from the exact distribution of interest [10].

It should be noted that the purpose of the MMC method differs from that of the advanced sampling techniques developed in the field of reliability analysis or rare event simulations, such as the cross entropy method [17], subset simulations [2], sequential Monte Carlo [8], etc. Namely, the purpose of those methods is to provide a variance-reduced estimator for a specific parameter associated with the distribution of y, while that of our method is to estimate the distribution of y itself. As will be shown in the next section, MMC is particularly useful for this purpose, which is our primary motivation to choose MMC over other advanced sampling schemes.

The rest of this paper is organized as the following. We first review the MMC method in Section 2, and then present our local GP construction algorithm in Section 3. Finally numerical examples are provided in Section 4 to demonstrate the performance of the proposed method.

2. The multicanonical Monte Carlo method

In this section we introduce the MMC algorithm, largely following the presentation of [7]. We start by summarizing the basic setup of our problem. Let $\mathbf{x}$ be a random vector taking values in the state space X, and $y = g(\mathbf{x})$ be a real scalar function of $\mathbf{x}$. For simplicity we assume that both $\mathbf{x}$ and y are continuous random variables whose probability density functions exist. We further assume that the PDF $p(\mathbf{x})$ of $\mathbf{x}$ is known, possibly up to an unknown normalization constant, and our goal is to determine the PDF $\pi(y)$ of y.

2.1. Flat histogram importance sampling

A popular strategy to estimate the PDF of a continuous random variable y with simulation, is to approximate the PDF with histograms, like a special case of the kernel density estimation. Suppose we are interested in the PDF of y in a given closed interval R_y, and we first equally decompose R_y into M bins of width Δ centered at the discrete values $\{b_1, ..., b_M\}$. We define the i-th bin as the interval $B_i = [b_i - \frac{\Delta}{2}, b_i + \frac{\Delta}{2}]$ and the probability for y to be in B_i is $P_i = \mathbb{P}\{y \in B_i\}$. The PDF of y at point y_i then can be approximated by

$$\pi(y_i) \approx P_i/\Delta,$$

if Δ is sufficiently small. This binning implicitly defines a partition of the input space X into M domains $\{D_i\}_{i=1}^{M}$, where

$$D_i = \{\mathbf{x} \in X : g(\mathbf{x}) \in B_i\}$$

is the domain in X that maps into the i-th bin B_i. See Fig. 1 for an illustration. Note that, while B_i are simple intervals, the domains D_i are multidimensional regions with possibly tortuous topologies. As a result, the probability P_i can be re-written as an integral in the input space:

$$P_i = \int_{D_i} p(\mathbf{x})dx = \int I_{D_i}(\mathbf{x})p(\mathbf{x})dx = \mathbb{E}[I_{D_i}(\mathbf{x})], \tag{2.1}$$

where $I_{D_i}(\mathbf{x})$ is an indicator function defined as,

$$I_{D_i}(\mathbf{x}) = \begin{cases} 1 & \mathbf{x} \in D_i; \\ 0 & \text{otherwise.} \end{cases}$$

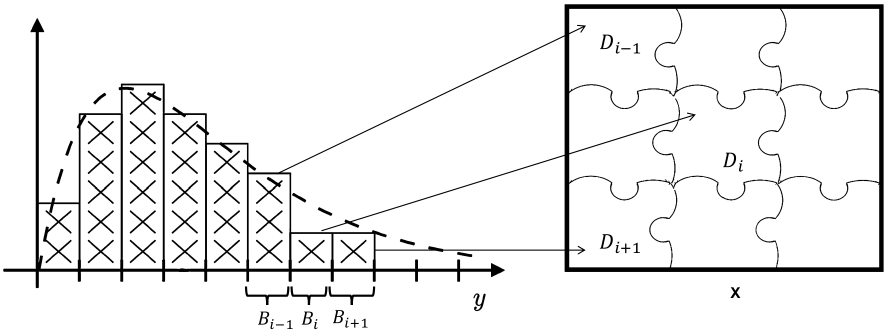

Fig. 1. Schematic illustration of the connection between B_i and D_i.

Now suppose that N samples $\{\mathbf{x}_1, \ldots, \mathbf{x}_N\}$ are drawn from the distribution $p(\mathbf{x})$, possibly with MCMC, P_i can be evaluated with the MC estimator:

$$\hat{P}_i^{MC} = \frac{1}{N}\sum_{j=1}^{N} I_{D_i}(\mathbf{x}_j) = \frac{N_i}{N}, \tag{2.2}$$

where N_i is the number of samples that fall in bin B_i.

As is well known, standard MC simulations have difficulty in reliably estimating the probabilities in the tail bins. The technique of importance sampling (IS) can be used to address the issue. Namely we choose a biasing distribution $q(\mathbf{x})$ and re-write (2.1) as

$$P_i = \int I_{D_i}(\mathbf{x})[\frac{p(\mathbf{x})}{q(\mathbf{x})}]q(\mathbf{x})d\mathbf{x} = \mathbb{E}^*[I_{D_i}(X)w(X)] \tag{2.3}$$

where $w(\mathbf{x}) = p(\mathbf{x})/q(\mathbf{x})$ is the IS weight, and $\mathbb{E}^*$ indicates expectation with respect to the biasing distribution $q(\mathbf{x})$. It follows that the IS estimator of P_i becomes

$$\hat{P}_i^{IS} = \left(\frac{N_i^*}{N}\right)\left[\frac{1}{N_i^*}\sum_{j=1}^{N} I_{D_i}(\mathbf{x}_j)w(\mathbf{x}_j)\right] \tag{2.4}$$

where the samples $\{\mathbf{x}_1, \ldots, \mathbf{x}_N\}$ are now drawn from the biasing distribution $q(\mathbf{x})$, and N_i^* is the number of samples falling in region D_i. For conciseness, we let $\hat{H}_i^* = \frac{N_i^*}{N}$. The intuition behind IS is that, the biasing distribution should assign higher probability in the region of interest than the original one, and thus it can draw more samples in that region.

The key of IS is to choose an appropriate biasing distribution $q(\mathbf{x})$ that can help to achieve the objective of the simulation. Unlike regular IS methods which usually employ biasing distributions that are easy to sample from, the MMC method chooses a biasing distribution $q(\mathbf{x})$ in the form of:

$$q(\mathbf{x}) = \begin{cases} \dfrac{p(\mathbf{x})}{\Theta(\mathbf{x})} & \mathbf{x} \in D; \\ 0 & \mathbf{x} \notin D, \end{cases} \tag{2.5}$$

where $\Theta(\mathbf{x}) = \Theta_i$. For $q(\mathbf{x})$ to be a well-defined distribution, we must have $\sum_{i=1}^{M} P_i/\Theta_i = 1$. It is easy to see that the distribution given in Eq. (2.5) assigns a constant weight to all $\mathbf{x} \in D_i$: $w(\mathbf{x}) = w_i$ for $\mathbf{x} \in D_i$ where $w_i = \Theta_i$, which is referred to be as uniform-weight (UW). In particular, if we let $\Theta_i = MP_i$ for all $x \in D_i, i = 1, ..., M$, the biasing distribution in Eq. (2.5) assigns equal probability to each bin and zero probability for any region outside $D = \cup_{i=1}^{M} D_i$, namely,

$$P_1^* = P_2^* = ...P_M^* = 1/M, \quad \text{where} \quad P_i^* = \int I_{D_i}(\mathbf{x})q(\mathbf{x})d\mathbf{x}. \tag{2.6}$$

We say such a biasing distribution as to be flat-histogram (FH). FH is an important feature for our purpose which is to have a good estimate of P_i for all $i = 1 \ldots M$.

2.2. Multicanonical Monte Carlo

It is easy to see, however, that the actual UW-FH distribution presented in Section 2.1 can not be used directly, as Θ_i depends on the sought after unknown P_i. The MMC method addresses the issue in an incremental manner. Simply speaking MMC iteratively constructs a sequence of distributions

$$q_k(\mathbf{x}) = \begin{cases} \dfrac{p(\mathbf{x})}{\Theta_k(\mathbf{x})}, & \mathbf{x} \in D; \\ 0 & \mathbf{x} \notin D, \end{cases} \tag{2.7}$$

where $\Theta_k(\mathbf{x}) = \Theta_{k,i}$ for $\mathbf{x} \in D_i$, converging to the actual UW-FH distribution. Specifically the sequence usually starts with $q_0(\mathbf{x})$ where $\Theta_{0,i} = \rho$ for all $i = 1, \ldots, M$ and $\rho = \sum_{i=1}^{M} P_i \leq 1$ is the probability that y falls in the region of interest.[1] The iteration is then guided by the following equation:

$$P_i^* = \int_{D_i} q(\mathbf{x}) d\mathbf{x} = \frac{\int_{D_i} p(\mathbf{x}) d\mathbf{x}}{c_\Theta \Theta_i} = \frac{P_i}{c_\Theta \Theta_i}, \tag{2.8}$$

or equivalently $P_i = P_i^* \Theta_i$. Namely, in the k-th iteration, one first draws N samples $\{\mathbf{x}_j\}_{j=1}^{N}$ from the current distribution $q_k(\mathbf{x})$, and then updates $\{\Theta_{k+1,i}\}_{i=1}^{M}$ using the following formulas, which are derived from Eq. (2.8),

$$\hat{H}_{k,i} = \frac{N_{k,i}^*}{N}, \tag{2.9a}$$

$$P_{k,i} = \hat{H}_{k,i} * \Theta_{k,j}, \tag{2.9b}$$

$$\Theta_{k+1,i} = P_{k,i}, \tag{2.9c}$$

where $N_{k,i}^*$ is the number of samples falling into region D_i in the k-th iteration. We reinstate that, unlike a usual IS method, which often chooses a biasing distribution easy to sample from, the biasing distribution of the MMC method Eq. (2.7) is not a standard distribution, and thus directly sampling from the distribution is rather difficult. To this end, MMC usually employs MCMC algorithm to draw samples from $q_k(\mathbf{x})$. Formal convergence analysis, as well as possible improvements of the method are not discussed in this work, and interested readers may consult, e.g., [3,4,15,16], and the references therein.

3. Accelerating MMC with local GP surrogates

In the MMC iteration, the main computational cost arises from performing the MCMC iteration to draw samples from each $q_k(\mathbf{x})$, for each sample requires a full-scale simulation of the underlying system. Thus, the MMC efficiency can be significantly improved by using computationally inexpensive surrogates in the MCMC scheme. As is mentioned in Section 1, here we adopt the adaptive surrogate construction scheme developed in [10]. In the work [10], the authors presented their method with two different surrogate models: the quadratic regression and the GP model, and their numerical results suggest that the GP model has better performance. We thus choose to use the GP model, while noting that other types of surrogates can also be used. In this section, we first briefly introduce the GP surrogate and then present the adaptive surrogate construction scheme modified for our specific use in MMC.

3.1. Gaussian process regression

The GP surrogates, which are also known as kriging, have been widely used in many practical problems (see e.g., [21]). The GP surrogate constructs the approximation of $g(\mathbf{x})$ in a nonparametric Bayesian regression framework [18,21]. Specifically the target function $g(\mathbf{x})$ is cast as

$$g(\mathbf{x}) = \mu_0(\mathbf{x}) + \eta(\mathbf{x}) \tag{3.1}$$

where $\mu_0(\mathbf{x})$ is a real-valued function and $\epsilon(\mathbf{x})$ is a zero mean Gaussian process whose covariance is specified by a kernel $K(\mathbf{x}, \mathbf{x}')$, namely,

$$\mathrm{COV}[\eta(\mathbf{x}), \eta(\mathbf{x}')] = K(\mathbf{x}, \mathbf{x}').$$

In practice, $\mu_0(\mathbf{x})$ can be represented as a linear or a quadratic polynomial whose coefficients are determined by simple regression. In this work, we assume it is a quadratic polynomial. The kernel $K(\mathbf{x}, \mathbf{x}')$ is positive semidefinite and bounded. Popular choices of the covariance functions include squared exponential, exponential, and Matern. The hyper-parameters inside the covariance functions can be prescribed or determined by maximizing the marginal likelihood function. Suppose that N computer simulations of the function $g(\mathbf{x})$ are performed at parameter values $\mathbf{X}^* := [\mathbf{x}_1^*, \ldots \mathbf{x}_n^*]$, yielding function evaluations $\mathbf{y}^* := [y_1^*, \ldots y_n^*]$, where

$$y_i^* = g(\mathbf{x}_i) \quad \text{for} \quad i = 1, \ldots, n.$$

Suppose we want to predict the function values at a given point $\mathbf{x}$, i.e., $y = g(\mathbf{x})$, the posterior of which is Gaussian:

[1] In practice, it is often convenient to assume that $\rho \approx 1$ and in this case we have $q_0(\mathbf{x}) \approx p(\mathbf{x})$.

$$y \mid \mathbf{x}, \mathbf{X}^*, \mathbf{y}^* \sim \mathcal{N}(\mu(\mathbf{x}), \sigma^2(\mathbf{x})). \tag{3.2}$$

The posterior mean of y is

$$\mu(\mathbf{x}) = \mu_0(\mathbf{x}) + K(\mathbf{x}, \mathbf{X}^*)^T K(\mathbf{X}^*, \mathbf{X}^*)^{-1}(\mathbf{y}^* - \mu_0(\mathbf{X}^*)), \tag{3.3a}$$

and the posterior variance is

$$\sigma^2(\mathbf{x}) = K(\mathbf{x}, \mathbf{x}) - K(\mathbf{x}, \mathbf{X}^*)^T\, K(\mathbf{X}^*, \mathbf{X}^*)^{-1}\, K(\mathbf{X}^*, \mathbf{x}), \tag{3.3b}$$

where the notation $K(\mathbf{A}, \mathbf{B})$ to denote the matrix of the covariance evaluated at all pairs of points in set $\mathbf{A}$ and in set $\mathbf{B}$ [21]. Eq. (3.3a) can be used as the surrogate to predict the function values at points of interest, and Eq. (3.3b) provides a measure of confidence in the predicted values.

3.2. Local GP construction

In the standard GP methods, the surrogates are constructed with all the data points. Constructing the GP surrogate this way can be very costly when the data set becomes large, as it involves inverting a large covariance matrix. On the other hand, it has been well noted that data points far from the point of interest have little influence on the prediction (assuming the usual choices of covariance). Thus, a natural choice is to construct GP only with the data points near the point of interest. The resulting surrogate is thus *local*, in the sense that it is only intended to be accurate at the point of interest. Next we discuss in detail how to construct a local GP surrogate at point $\mathbf{x}$ given a collection of model evaluations: $\mathbf{S} := \{(\mathbf{x}_i, y_i)\}_{i=1}^{n_S}$ where $y_i = g(\mathbf{x}_i)$ for $i = 1...n_S$.

First we need to determine how many data points we want to use in the surrogate construction. Following the suggestion of [10], we choose the number of data points n as

$$n = \sqrt{d_x}(d_x + 1)(d_x + 2)/2,$$

where d_x is the dimensionality of $\mathbf{x}$. This choice allows us to have sufficient data points to perform a quadratic regression for $\mu_0(\mathbf{x})$. The specific points used to build the surrogate are chosen with the nearest neighbor (NN) method: namely, we use the n points closest to $\mathbf{x}$ to construct the GP surrogate. It has been pointed out that the NN method only provides a suboptimal point selection, and better selection strategy can be obtained by solving an optimization problem. However, in our problem, the GP construction must be done repeatedly in the MCMC scheme, and as a result even very fast optimization may significantly increase the total computational cost. In this respect, we nevertheless adopt the NN method for the sake of computational simplicity. In what follows, we refer to a local GP surrogate constructed with the prescribed procedure, as $\widetilde{g}(\mathbf{x}|\mathbf{S})$.

3.3. MCMC with local GP surrogates

In this section, we present a modified version of the local surrogate accelerated MCMC scheme developed in [10]. The method embeds an adaptive surrogate construction in the MCMC iteration: in each iteration the method constructs a local surrogate using data set $\mathbf{S}$, for the proposed point and the current point, and decides whether it needs model refinement; when refinement is needed, the algorithm then refines the surrogate by evaluating more points near the proposed point or the current one depending on where the refinement is triggered; all the evaluated points are included in the data set $\mathbf{S}$ which will be used for constructing surrogates in the next step. In [10], refinement is triggered by either of two criteria. The first is random: with probability γ_t, the model refined at either the current point or the proposed point. The second criterion used in [10], intended to make the algorithm efficient in practice, is based on an error indicator of the acceptance probability. In this work, we follow the random criteria and choose γ_t to be a constant for simplicity. We use, however, a different practical criterion, taking advantage of the special structure of the target distribution $q_k(\mathbf{x})$ in Eq. (2.7). Namely, it is easy to see that, for $q_k(\mathbf{x})$ in Eq. (2.7), an error in the surrogate does not cause an error in the acceptance probability unless the surrogate assigns the sample into a wrong bin, assuming a symmetric proposal distribution. Specifically, suppose the current sample is $\mathbf{x}^-$ and the proposed sample is $\mathbf{x}^+$, and the posterior mean and variance of the GP at $\mathbf{x}^+$ are y^+ and ϵ^2 respectively. Suppose it is assigned to bin $B_i = [b_i - \Delta/2,\, b_i + \Delta/2]$ based on the predicted value y^+, and the probability that the assignment of $\mathbf{x}_i$ is incorrect can be computed as

$$\begin{aligned}\beta(\mathbf{x}^+) &:= \mathbb{P}[g(\mathbf{x}^+) < b_i - \Delta/2 \,\mathrm{or}\, g(\mathbf{x}^+) > b_i + \Delta/2] \\ &= \Phi(b_i - \Delta/2, y^+, \sigma^+) - \Phi(b_i + \Delta/2, y^+, \sigma^+) + 1,\end{aligned} \tag{3.4}$$

where $\Phi(\cdot, y^+, \epsilon)$ is the cumulative density function (CDF) of the normal distribution with mean y^+ and standard deviation $\sigma^+ = \sigma(\mathbf{x}^+)$. Thus we can define the refinement criteria as that the misassignment probability β is smaller than a threshold value: $\beta < \beta_{\max}$. Since the refinement criteria is applied to each iteration, the probability that the acceptance probability computed with the surrogate is erroneous is bounded by $2\beta_{\max}$, in any iteration. As a result, to achieve this probability boundedness, we only need to check if $\mathbf{x}^+$ satisfies the quality condition: $\beta(\mathbf{x}^+) < \beta_{\max}$, as $\mathbf{x}^-$ has been verified

Algorithm 1 Metropolis–Hastings with local GP surrogates.

```
for $t = 1, ..., T$ do
    $(\mathbf{x}_{t+1}, y_{t+1}, \mathbf{S}_{t+1}) \leftarrow K_t(\mathbf{x}_t, y_t, \mathbf{S}_t, q(\cdot; y_t), \gamma_t, \beta_{\max})$
end for

procedure $K_t(\mathbf{x}^-, y^-, \mathbf{S}, q(\cdot; y^-), \gamma, \beta_{\max})$
    Draw proposal $\mathbf{x}^+ \sim \Pi(\mathbf{x}^-, \cdot)$
    $(y^+, \epsilon^+) \leftarrow \widetilde{g}(\mathbf{x}^+, \mathbf{S})$
    if $u \sim \text{Uniform}(0, 1) < \gamma$ then
        $y^+ = g(\mathbf{x}^+)$
        $\mathbf{S} \leftarrow \mathbf{S} \cup \{(\mathbf{x}^+, y^+)\}$
    else
        $\beta \leftarrow 1 + \Phi(b_i - \Delta/2, y^+, \sigma^+) - \Phi(b_i + \Delta/2, y^+, \sigma^+)$
        if $\beta > \beta_{\max}$ then
            $y^+ = g(\mathbf{x}^+)$
            $\mathbf{S} \leftarrow \mathbf{S} \cup \{(\mathbf{x}^+, y^+)\}$
        end if
    end if
    $\alpha \leftarrow q(\mathbf{x}^+; y^-)/q(\mathbf{x}^-; y^-)$
    if $u \sim \text{Uniform}(0, 1) < \alpha$ then
        return $(\mathbf{x}^+, y^+, \mathbf{S})$
    else
        return $(\mathbf{x}^-, y^-, \mathbf{S})$
    end if
end procedure
```

in the previous iteration. We outline our algorithm in Algorithm 1, where the surrogate construction is integrated into a standard Metropolis–Hastings (MH) MCMC scheme.

We have the following remarks regarding the proposed algorithm, highlighting its differences from that given in [10] in addition to the refinement criteria.

- As a pre-processing of the first MMC iteration, we choose n_o points, and use them as the initial data set $\mathbf{S}$. These points can be chosen in many different ways: sampling according to $p(\mathbf{x})$, Latin hypercube, or experimental design methods. For the succeeding MMC iterations, the data set $\mathbf{S}$ is simply taken to be that obtained in the previous round.
- Unlike regular MCMC methods, in each iteration our algorithm returns the sample $\mathbf{x}_t$ as well as the function value y_t for the sample. Note that, the function values are needed in Eqs. (2.9), and thus by recording the function values, we can compute Eq. (2.9) without evaluating the function again.
- As has been discussed in the beginning of Section 3.3, in each iteration we only need to consider the quality of the surrogate at the proposed point $\mathbf{x}^+$ thanks to the special structure of $q_k(\mathbf{x})$, while in the original algorithm, both $\mathbf{x}^+$ and $\mathbf{x}^-$ need to be examined.
- In our algorithm, when model refinement is needed, we simply evaluate the current point $\mathbf{x}^+$. It has been suggested that this strategy may lead to poor conditioned regression in particular when polynomial surrogates are used, and as an alternative a space filling approach is used in [10]. However, we have found it is not a very serious issue for the GP surrogates in our numerical tests, and, considering that the space filling method requires an extra optimization step, we choose to directly evaluate $\mathbf{x}^+$ for simplicity's sake.

Finally we note that it is a very interesting problem to analyze the convergence property of the algorithm. To this end, the convergence analysis in [10] can provide certain useful results of the MCMC iterations. However, since the algorithm is a combination of the two methods, a formal convergence analysis can be very challenging, and so is not pursued in this work.

4. Numerical examples

We use three numerical examples to demonstrate the performance of the proposed GP accelerated MMC (GP-MMC) method. Before proceeding to the examples, we describe the specific GP surrogate used in all the three examples. First in all the examples we use an anisotropic covariance function in the form of:

$$K(\mathbf{x}, \mathbf{x}') = a \exp\left[-\sum_{i=1}^{d_x} \frac{|x_i - x_i'|^p}{l_i}\right], \tag{4.1}$$

where p is a prescribed positive integer which usually takes values of 1 (the exponential kernel) or 2 (the squared exponential kernel), the coefficient a is determined with empirical Bayes in the iteration, and the correlation length $\mathbf{l} = (l_1, ..., l_{d_x})$ is determined from the initial data set and is not adjusted in the iteration. Note that, the correlation length $\mathbf{l}$ can also be determined with empirical Bayes in the iteration if desired, but we choose not to do so here for simplicity's sake, as it requires to numerically solving an optimization problem.

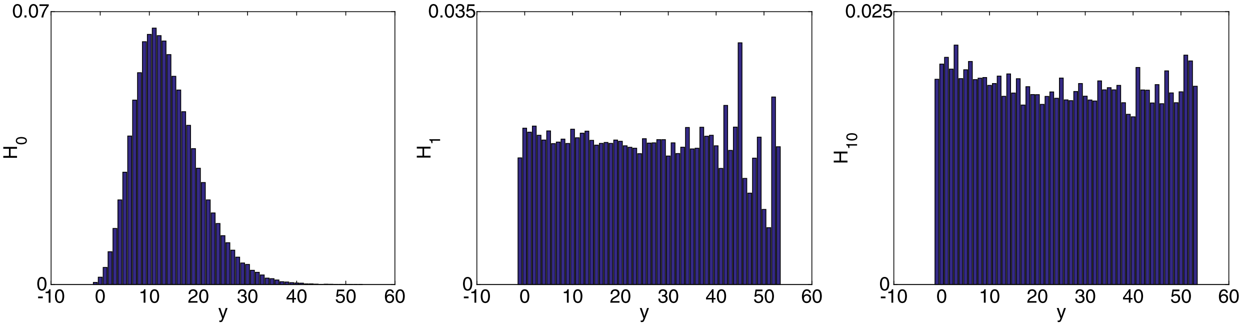

Fig. 2. The histograms of the first two steps and the 10th iteration of MMC.

Table 1
(Example 1) The performance results of GP-MMC with various values of $\beta_{\max}$.

$\beta_{\max}$	0.92	0.76	0.32	0.05	0.003	Plain MMC
True model evals	796	810	809	926	1089	10^6
Maximum RelErr	0.1775	0.148	0.1058	0.1321	0.1217	0.0921
Average RelErr	0.0419	0.039	0.0327	0.0333	0.0345	0.0225

4.1. A multi-dimensional analytical example

Our first example is a multi-dimensional problem where the performance function is

$$g(\mathbf{x}) = \min\{g_1(\mathbf{x}),\ g_2(\mathbf{x})\} - 1,$$

with

$$g_1(\mathbf{x}) = \|\mathbf{x} - \mathbf{x}_1\|, \quad \text{and} \quad g_2(\mathbf{x}) = \|\mathbf{x} - \mathbf{x}_2\|.$$

The input $\mathbf{x}$ are multidimensional independently distributed standard normal random variables and $\mathbf{x}_1, \mathbf{x}_2$ are two fixed points. It is obvious that each D_i has two possibly disjoint sections: $\{\mathbf{x} \,|\, g_1(\mathbf{x}) \in B_i\}$ and $\{\mathbf{x} \,|\, g_2(\mathbf{x}) \in B_i\}$, which makes the problem challenging for many variance-reducing sampling techniques.

We first test our method for the two dimensional case and choose $\mathbf{x}_1 = (3,3)$ and $\mathbf{x}_2 = (3,-3)$ respectively. We run standard MC simulations with 10^7 samples, and use its results as the "truth" to validate the estimates of the MMC methods. In the first numerical experiment, we perform MMC simulations without using surrogates, where 10 iterations are used with 10^5 samples in each iteration, resulting in a total computational cost of 10^6 full-model simulations. When constructing the PDF, we use $R_y = [-1, 54]$ which is divided into 55 bins. In Fig. 2 we show the histograms obtained in the 1st, 2nd and the final MMC iteration, from which one can see that the histograms tend to become flat as the iterations proceed.

Our second numerical experiment is to run MMC with the assistance of the GP surrogates, and, as is in the first experiment, we again use 10 iterations with 10^5 samples in each. In the GP-MMC computation, we construct the GP surrogates as is described in the beginning of the section, where the kernel is given by Eq. (4.1) with $p = 1$. The initial data set contains 50 samples randomly drawn from the distribution of $\mathbf{x}$, and we choose the random model refinement probability $\gamma_t = 10^{-4}$. The key parameter in the algorithm is the maximum misassignment probability $\beta_{\max}$, and to examine the robustness of our method against the choices of $\beta_{\max}$, we implement our method with various values of $\beta_{\max}$ and show the results in Table 1. In particular, for the results of each value of $\beta_{\max}$, we show the number of true model evaluations, the maximum and the average relative errors (compared to the MC results) of all the bins.

One can see that, the method performs well even for very large misassignment probabilities, and the results are rather robust for different values of $\beta_{\max}$ except that the number of true model evaluations grows as $\beta_{\max}$ becomes smaller.

To further compare the results, we plot the PDF obtained by MC, MMC and GP-MMC with $\beta_{\max} = 0.05$, in Fig. 3 (Top), and one can see that the results of the three methods agree very well with each other. To have a quantitative assessment of the performance, we compute the relative error of the MMC and the GP-MMC estimates, against the results of plain MC:

$$\mathrm{RelErr}_{\mathrm{MMC}} = \frac{|\hat{p}_{\mathrm{MMC}} - \hat{p}_{\mathrm{MC}}|}{\hat{p}_{\mathrm{MC}}}, \quad \mathrm{RelErr}_{\mathrm{GPMMC}} = \frac{|\hat{p}_{\mathrm{GPMMC}} - \hat{p}_{\mathrm{MC}}|}{\hat{p}_{\mathrm{MC}}}, \tag{4.2}$$

and show the results in Fig. 3 (Bottom).

We see that, the relative errors in both MMC and GP-MMC are around 0.1, indicating that both MMC and GP-MMC produce reliable estimates of the PDF of y. To further compare the performance, we computed the mean, the variance, the 3rd, the 4th and the 5th central moments of y using the samples obtained by the three methods shown in Table 2, which shows that the results obtained by the three methods agree well with each other. Regarding the computational cost, the MMC method uses 10^6 full model evaluations while our GP-MMC method only uses less than a thousand full-model evaluations.

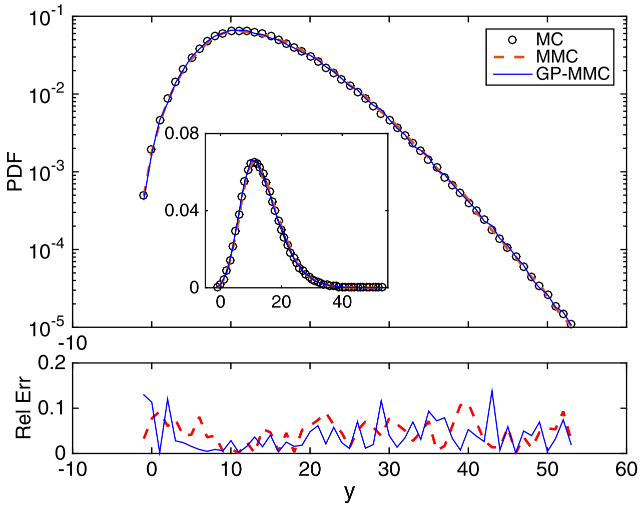

Fig. 3. (Example 1) Top: the PDF of y obtained by MC (circles), MMC (dashed line) and GP-MMC (solid line) on a logarithmic scale; inset is the same plots on a linear scale. Bottom: the relative error in the PDF obtained by MMC (dashed) and GP-MMC (solid).

Table 2
(Example 1) The mean, variance, and 3rd–5th central moments of y, estimated by MC, MMC and GP-MMC.

Moment	Mean	Var	3rd	4th	5th
MC	14.21	43.58	217.42	7340.55	108583.52
MMC	14.43	44.04	241.11	7505.02	113171.57
GP-MMC	14.28	44.04	230.10	7456.33	111877.63

Table 3
(Example 1) The performance of MMC and GP-MMC with respect to various sample sizes.

	Sample size	1e+4	1e+5	1e+6
MMC	maximum RelErr	0.3097	0.1858	0.0466
	average RelErr	0.0791	0.0387	0.0120
GP-MMC	true model evals	891	1855	2033
	maximum RelErr	0.2593	0.0906	0.0576
	average RelErr	0.087	0.0328	0.0147

Table 4
(Example 1) The performance of MMC and GP-MMC with respect to various numbers of dimensions.

	Dimension	2	8	16
MMC	maximum RelErr	0.1173	0.1168	0.1531
	average RelErr	0.0225	0.0370	0.0566
GP-MMC	true model evals	891	3226	16886
	maximum RelErr	0.1497	0.1521	0.1692
	average RelErr	0.0414	0.0422	0.0665

We also consider the performance of the proposed method with respect to different sample sizes and dimensionality. To this end, we first perform the GP-MMC method as well as standard MMC with different number of samples in each iteration, in which $\beta_{\max}$ is taken to be 0.075. The results are shown in Table 3, and as expected, with more samples in each iteration, the results become more accurate at the price of more true model evaluations. Next, we consider the example with different number of dimensions. In this case we let $\mathbf{x}_1 = (1, ..., 1)^d$ and $\mathbf{x}_2 = (-1, ..., -1)^d$ for $d = 2, 8, 16$. We perform both MMC and GP-MMC in each case, and in the GP-MMC we take $\beta_{\max} = 0.075$ and the number of sample size in each iteration to be 5×10^4. The performance comparison is shown in Table 4. We can see from the results that as the dimensionality increases, the GP-MMC method requires more true model evaluations, but the computational cost saving compared to standard MMC is still significant even for the case of 16 dimensions. Overall we have found that the performance of the GP-MMC method is rather robust with respect to the sample size and the dimensionality.

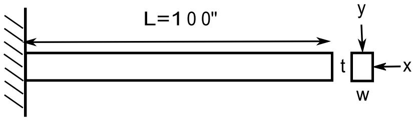

Fig. 4. (Example 2) Schematic illustration of a cantilever beam subject to horizontal and vertical loads.

Table 5
(Example 2) The mean and variance of the random parameters in the cantilever beam model.

Parameter	w	t	X	Y	E
Mean	4	4	500	1000	2.9×10^6
Variance	0.001	0.0001	100	100	1.45×10^6

Table 6
(Example 2) The performance results of GP-MMC with various values of $\beta_{\max}$.

$\beta_{\max}$	0.92	0.76	0.32	0.05	0.003
True model evals	894	2523	4775	7456	7589
Maximum RelErr	0.116	0.143	0.102	0.099	0.089
Average RelErr	0.034	0.042	0.038	0.038	0.037

Table 7
(Example 2) The mean, variance, and 3rd–5th central moments of y, estimated by MC, MMC and GP-MMC.

Moment	Mean	Var	3rd	4th	5th
MC	0.6024	8.99e−5	6.28e−8	2.43e−8	4.96e−11
MMC	0.6024	8.97e−5	7.04e−8	2.43e−8	5.32e−11
GP-MMC	0.6025	9.04e−5	7.55e−8	2.46e−8	5.54e−11

4.2. Cantilever beam

We now consider a cantilever beam problem [17,22] as illustrated in Fig. 4, with width w, height t, length L, and subject to transverse load Y and horizontal load X. This is a popular benchmark problem in the reliability analysis literature, where the performance function is

$$y = \frac{4L^3}{Ewt}\sqrt{\left(\frac{Y}{t^2}\right)^2 + \left(\frac{X}{w^2}\right)^2},$$

which represents the deflection of the beam. In this example, we assume that the beam length is fixed $L = 100$, and the beam width w, the height x, the applied loads X and Y, as well as the elastic module E of the material, are random parameters. We further assume that these random parameters are all independently distributed, with each following a normal distribution. The means and the variances of the parameters are summarized in Table 5.

In this example, we also compute the PDF of y with three methods: plain MC, MMC and GP-MMC. In the MC simulations, we use 10^9 full model evaluations. In both MMC and GP-MMC, we use 10 iterations where 10^5 samples in each iteration. In the GP-MMC computation, the number of initial data and the values of γ_t are the same as those used in the first example. The GP kernel is also given by Eq. (4.1) with $p = 1$. Also, we test the GP-MMC method with various values of $\beta_{\max}$ and show the results in Table 6. In this example, we use $R_y = [0.56, 0.66]$ which is divided into 40 bins. To compare the results, we plot the PDF obtained by MC, MMC and GP-MMC with $\beta_{\max} = 0.32$ which requires 4775 true model evaluations, as well as the relative errors of MMC and GP-MMC, in Figs. 5. We also show the same moment plots as is in the first example in Table 7. All the figures indicate that our GP-MMC method yields very reliable estimates of the PDF of y, while its computational cost is significantly lower than both MC and standard MMC.

4.3. Random PDE example

Finally we consider a random partial differential equation (PDE) example: a two-dimensional Poisson equation on region $\Gamma = [0, 1] \times [0, 1]$:

$$\nabla(a(\mathbf{x})\nabla u(\mathbf{x})) = f(\mathbf{x}), \tag{4.3a}$$

$$u = 0 \quad \text{on} \quad \partial\Gamma, \tag{4.3b}$$

where $a(\mathbf{x})$ is a random field and $\partial\Gamma$ is the boundary of Γ. We want to compute the statistical distribution of the value of u at location $\mathbf{x}_* \in \Gamma$. A physical interpretation of the problem is the following: we consider a steady flow in an isotropic

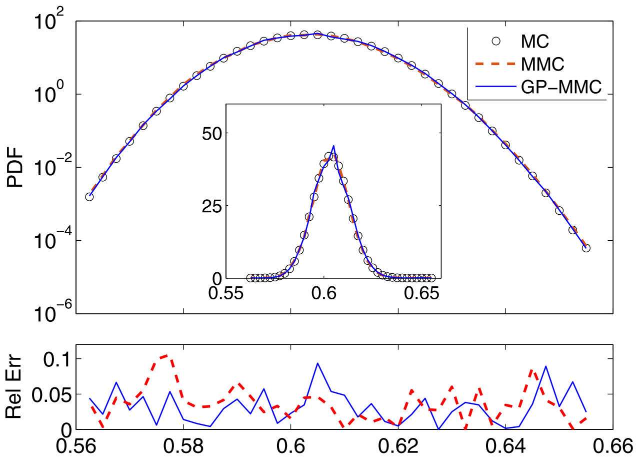

Fig. 5. (Example 2) Top: the PDF of y obtained by MC (circles), MMC (dashed line) and GP-MMC (solid line) on a logarithmic scale; inset is the same plots on a linear scale. Bottom: the relative error in the PDF obtained by MMC (dashed) and GP-MMC (solid).

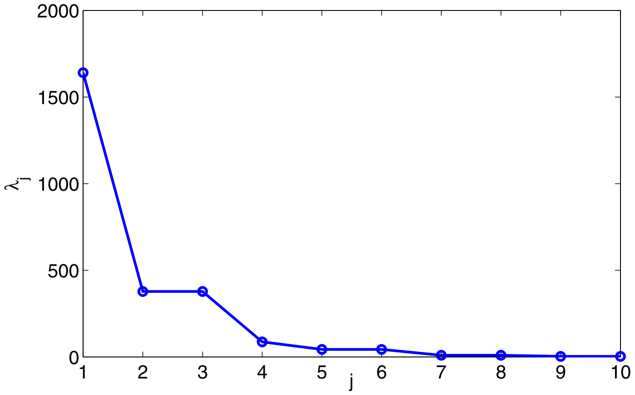

Fig. 6. (Example 3) The eigenvalues of the KL expansion plotted in a descending order.

aquifer subject to random permeability [1], and we are interested in the statistical information of the hydraulic head at a particular location $\mathbf{x}_*$.

We further assume the permeability is a log-normal random field, namely, $a(\mathbf{x}) = a_o \exp(z(\mathbf{x}))$ where $z(\mathbf{x})$ is a Gaussian random field with zero mean and covariance kernel,

$$\Sigma(\mathbf{x}_1, \mathbf{x}_2) = \exp(-\frac{\|\mathbf{x}_1 - \mathbf{x}_2\|^2}{\Delta}). \tag{4.4}$$

In this example we take $a_0 = 1$ and $\Delta = 0.6$. In practice, the random field $z(\mathbf{x})$ in the PDE is often represented with a truncated Karhunen–Loève (K–L) expansion. Namely, let $\{\lambda_j, \xi_j(\mathbf{x})\}_{j=1}^{\infty}$ be the eigenvalue–eigenfunction pairs of the covariance kernel $\Sigma(\cdot,\cdot)$ such that $\lambda_j > \lambda_{j+1}$ for all $j = 1 \ldots \infty$, and we can approximate $z(\mathbf{x})$ with

$$z(\mathbf{x}) = \sum_{j=1}^{J} c_j \sqrt{\lambda_j} \xi_j(\mathbf{x}), \tag{4.5}$$

where $\mathbf{c} = (c_1, ..., c_J)$ follows a standard isotropic normal distribution. Thus the dimensionality of the problem is reduced to J and in this example we choose $J = 10$. We plot the eigenvalues associated with the 10 KL modes in a descending order in Fig. 6, which suggests that 10 KL-modes can sufficiently represent the Gaussian field $z(\mathbf{x})$ in this problem. Moreover, in the numerical simulations, we take $f(\mathbf{x}) = 1$ and $\mathbf{x}^* = (0.5, 0.5)$. A sample coefficient $a(\mathbf{x})$ and the associated solution $u(\mathbf{x})$ is shown in Fig. 7.

As the computational cost for solving Eq. (4.3) is rather high, which renders standard MC unfeasible, we choose to only perform MMC and GP-MMC simulations in this problem. In both cases, we use 10 iterations with 20000 samples in each iteration. In GP-MMC, we use the covariance function (4.1) with $p = 2$. The number of initial samples is 400, $\gamma_t = 10^{-4}$ and $\beta_{\max} = 0.05$. As a result the total number of true model evaluations is 4885. When constructing the PDF, we use $R_y = [-2, 0]$ divided into 20 bins. We plot the PDF computed with MMC and GP-MMC as well as the relative error in the

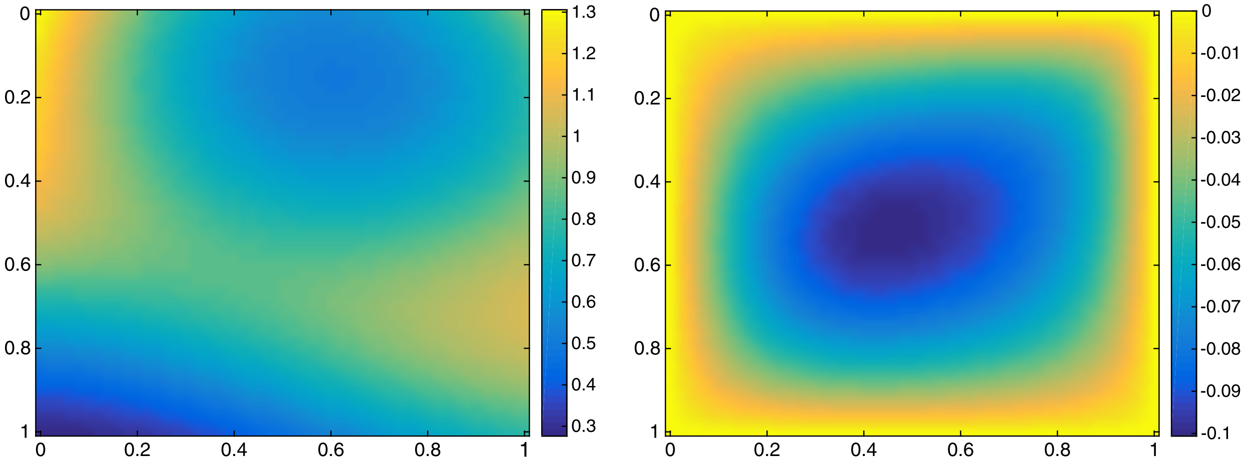

Fig. 7. Left: a randomly drawn coefficient sample $a(\mathbf{x})$. Right: the solution of Eq. (4.3) associated with $a(\mathbf{x})$.

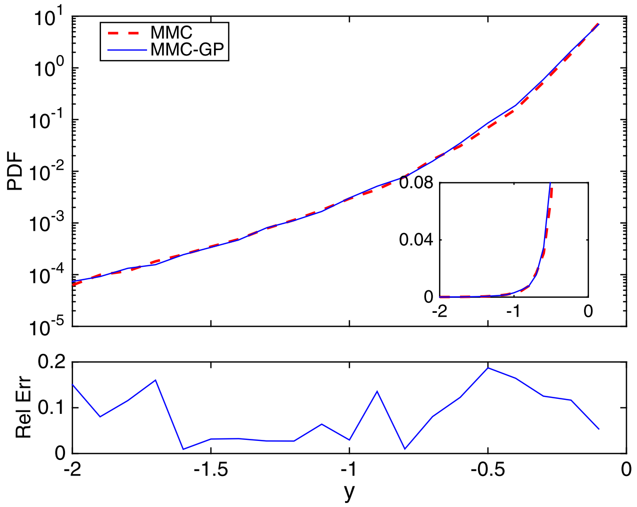

Fig. 8. (Example 3) Top: the PDF of y obtained by MMC (dashed line) and GP-MMC (solid line) on a logarithmic scale; inset is the same plots on a linear scale. Bottom: the relative error in the PDF.

two results in Fig. 8. One can see from the figures that the results of GP-MMC agree very well with those of plain MMC, while it only uses around one fortieth true model evaluations of the plain MMC.

5. Conclusions

We consider a special type of UQ problems where the system performance is characterized by a scalar parameter. We use a MMC based method to compute the distribution of the performance parameter, and we also propose to use a local GP surrogate to accelerate the MMC simulations. Based on the work [10], we design an adaptive algorithm that can effectively refine the GP surrogate in the MMC iterations. With numerical examples, we demonstrate that the proposed GP-MMC method can efficiently and accurately compute the distribution of the performance parameter. We expect the proposed method can be useful in various fields of applications, such as reliability analysis, risk management, and utility optimizations.

There are a number of possible improvements and extensions of the proposed method that we plan to investigate in the future. First there are some well-known open issues with GP: most notably, how to choose the best covariance functions, and such a choice may certainly affect the performance of our MMC-GP method. To this end, we hope to develop approaches that can effectively choose the covariance functions for our MMC method. Second, as has been mentioned in Section 3, we are not able to provide a convergence analysis of the proposed method in this paper and we hope to address the issue in a future work. Third we are also interested in more general uncertainty propagation problems where the output is a multidimensional vector rather than a scalar. In this case, the standard MMC scheme does not apply directly, due to the multi-dimensionality of the output. We plan to tackle such problems with modified MMC algorithms. Finally, we note that the Wang–Landau algorithms, which can be regarded as a variant of MMC, have been applied to the Bayesian inference problems (e.g. [9]), and we hope that our GP-MMC method can be applied to such problems as well. In this case, we expect that our method can further improve the computational efficiency of the Bayesian inferences, thanks to the use of surrogates.

Acknowledgements

We want to thank the anonymous reviewer for his/her very constructive comments. The work was partially supported by the National Natural Science Foundation of China under grant number 11301337.

References

[1] Mary P. Anderson, William W. Woessner, Applied Groundwater Modeling: Simulation of Flow and Advective Transport, vol. 4, Gulf Professional Publishing, 1992.
[2] S.K. Au, J. Beck, A new adaptive importance sampling scheme for reliability calculations, Struct. Saf. 21 (2) (1999) 135–158.
[3] Bernd A. Berg, Introduction to multicanonical Monte Carlo simulations, Fields Inst. Commun. 26 (1) (2000) 1–24.
[4] Bernd A. Berg, Markov Chain Monte Carlo Simulations and Their Statistical Analysis: With Web-Based Fortran Code, World Scientific, 2004.
[5] Bernd A. Berg, Thomas Neuhaus, Multicanonical algorithms for first order phase transitions, Phys. Lett. B 267 (2) (1991) 249–253.
[6] Bernd A. Berg, Thomas Neuhaus, Multicanonical ensemble: a new approach to simulate first-order phase transitions, Phys. Rev. Lett. 68 (1) (1992) 9.
[7] Alberto Bononi, Leslie Rusch, Amirhossein Ghazisaeidi, Francesco Vacondio, Nicola Rossi, et al., A fresh look at multicanonical Monte Carlo from a telecom perspective, in: IEEE Global Telecommunications Conference, GLOBECOM 2009, IEEE, 2009, pp. 1–8.
[8] Frederic Cerou, Pierre Del Moral, Teddy Furon, Arnaud Guyader, Sequential Monte Carlo for rare event estimation, Stat. Comput. 22 (3) (2012) 795–808.
[9] Nicolas Chopin, Tony Lelièvre, Gabriel Stoltz, Free energy methods for Bayesian inference: efficient exploration of univariate Gaussian mixture posteriors, Stat. Comput. 22 (4) (2012) 897–916.
[10] Patrick R. Conrad, Youssef M. Marzouk, Natesh S. Pillai, Aaron Smith, Accelerating asymptotically exact MCMC for computationally intensive models via local approximations, arXiv preprint, arXiv:1402.1694, 2014.
[11] Tobin A. Driscoll, Kara L. Maki, Searching for rare growth factors using multicanonical Monte Carlo methods, SIAM Rev. 49 (4) (2007) 673–692.
[12] Xiaoping Du, Wei Chen, Sequential optimization and reliability assessment method for efficient probabilistic design, J. Mech. Des. 126 (2) (2004) 225–233.
[13] George A. Hazelrigg, A framework for decision-based engineering design, J. Mech. Des. 120 (4) (1998) 653–658.
[14] Ronald Holzlööhner, Curtis R. Menyuk, Use of multicanonical Monte Carlo simulations to obtain accurate bit error rates in optical communications systems, Opt. Lett. 28 (20) (2003) 1894–1896.
[15] Yukito Iba, Nen Saito, Akimasa Kitajima, Multicanonical mcmc for sampling rare events: an illustrative review, Ann. Inst. Stat. Math. 66 (3) (2014) 611–645.
[16] David P. Landau, Kurt Binder, A Guide to Monte Carlo Simulations in Statistical Physics, Cambridge University Press, 2014.
[17] Jing Li, Jinglai Li, Dongbin Xiu, An efficient surrogate-based method for computing rare failure probability, J. Comput. Phys. 230 (24) (2011) 8683–8697.
[18] Anthony O'Hagan, J.F.C. Kingman, Curve fitting and optimal design for prediction, J. R. Stat. Soc. B (1978) 1–42.
[19] Rüdiger Rackwitz, Struct. Saf. 23 (4) (2001) 365–395.
[20] R. Tyrrell Rockafellar, Stanislav Uryasev, Optimization of conditional value-at-risk, J. Risk 2 (2000) 21–42.
[21] Christopher K.I. Williams, Carl Edward Rasmussen, Gaussian Processes for Machine Learning, The MIT Press, 2006.
[22] Y-T. Wu, H.R. Millwater, T.A. Cruse, Advanced probabilistic structural analysis method for implicit performance functions, AIAA J. 28 (9) (1990) 1663–1669.
[23] David Yevick, Multicanonical communication system modeling-application to PMD statistics, IEEE Photonics Technol. Lett. 14 (11) (2002) 1512–1514.