Quantum self-learning Monte Carlo and quantum-inspired Fourier transform sampler

Katsuhiro Endo,[1] Taichi Nakamura,[2] Keisuke Fujii,[3,4] and Naoki Yamamoto[5]

[1]*Department of Mechanical Engineering, Keio University, Hiyoshi 3-14-1, Kohoku, Yokohama, Japan*
[2]*Department of System Design Engineering, Keio University, Hiyoshi 3-14-1, Kohoku, Yokohama, Japan*
[3]*Graduate School of Engineering Science, Osaka University, 1-3 Machikaneyama, Toyonaka, Osaka 560-8531, Japan*
[4]*Center for Quantum Information and Quantum Biology, Institute for Open and Transdisciplinary Research Initiatives, Osaka University, Osaka 560-8531, Japan*
[5]*Department of Applied Physics and Physico-Informatics, and Quantum Computing Center, Keio University, Hiyoshi 3-14-1, Kohoku, Yokohama, Japan*

(Received 2 June 2020; accepted 14 December 2020; published 31 December 2020)

The self-learning METROPOLIS-Hastings algorithm is a powerful Monte Carlo method that, with the help of machine learning, adaptively generates an easy-to-sample probability distribution for approximating a given hard-to-sample distribution. This paper provides a new self-learning Monte Carlo method that utilizes a quantum computer to output a proposal distribution. In particular, we show a novel subclass of this general scheme based on the quantum Fourier transform circuit; when the dimension of the input to QFT corresponding to the low-frequency components is not large, this sampler is classically simulable while having a certain advantage over conventional methods. The performance of this quantum-inspired algorithm is demonstrated by some numerical simulations.

DOI: 10.1103/PhysRevResearch.2.043442

I. INTRODUCTION

Monte Carlo (MC) simulation is a powerful statistical method that is generically applicable to compute statistical quantities of a given system by sampling random variables from its underlying probability distribution. A particularly efficient method is the METROPOLIS-Hastings (MH) algorithm [1]; this realizes a fast sampling from a target distribution via an appropriate acceptance/rejection filtering of the samples generated from an alternative proposal distribution which is easier to sample compared to the target one. Therefore the most important task in this algorithm is to specify an appropriate proposal distribution satisfying the following three conditions; (i) it must be easy to sample, (ii) the corresponding probability can be effectively computed for judging acceptance/rejection of the sample, and (iii) it is rich in representation, meaning that it may lie near the target distribution. This is a long-standing challenging problem, but the recent rapid progress of machine learning enables us to take a circumventing pathway for the issue, the self-learning MC [2–5], which introduces a parametrized proposal distribution and updates the parameters during the sampling so that it is going to mimic the target one. Despite of its potential power thanks to the aid of machine learning, this approach has been demonstrated only with a few physical models; for example in Ref. [2], for a target hard-to-sample Ising distribution, a parametrized proposal Ising distribution was applied to demonstrate the effectiveness of self-learning MC approach. To expand the scope of self-learning MC, we need a systematic method to design a parametrized proposal distribution that is generically applicable to a wide class of target distribution.

Now we turn our attention to quantum regime, with the hope that the quantum computing might provide us an effective means to attack the above-mentioned problem. In fact the so-called quantum supremacy holds for sampling problems; that is, a (nonuniversal) quantum computer can generate a probability distribution which is hard to sample via any classical (i.e., nonquantum) computer. Especially, the Boson sampling [6] and instantaneous quantum polynomial time computations [7,8] are well known, the former of which is now even within reach of experimental demonstration [9,10]. Moreover, a recent trend is to extend this idea to quantum learning supremacy [11], meaning that a quantum circuit is trained to learn a given target distribution faster than any classical computer.

With the above background in mind, in this paper we study a new type of self-learning MC that uses quantum computing to generate a proposal distribution. In fact this scheme satisfies the above-described three conditions. First, (i) is already fulfilled as an intrinsic nature of quantum computers. Second, it is well known that the task (ii) can be effectively executed using the amplitude estimation algorithm [12,13], which is in fact twice as fast as the classical correspondence. Lastly the above-described fact, the expressive power of quantum computers for generating complex probability distributions, might enable us to satisfy (iii) and mimic a target distribution which is essentially hard to sample via any classical means.

To realize the learning scheme in a quantum system, we take the variational method, meaning that a parametrized quantum circuit is trained so that its output probability distribution approaches to the target distribution. This schematic itself is also employed in the quantum generative modeling [14,15], but application to MH might be of more practical use for the following reason. That is, unlike the generative modeling problem, MH need not generate a proposal distribution that is very close to the target, but rather it requires only a relatively high acceptance ratio and accordingly less demanding quantum computers.

Of course the most difficult part is to design a parametrized quantum circuit which may successfully generate a suitable proposal distribution. Therefore, for the purpose of demonstrating the proof-of-concept, in this work, we consider a special type of quantum circuit composed of the quantum Fourier transform (QFT), where the parameters to be learned are assigned corresponding to some frequency components. In fact, thanks to the expressive power of the Fourier transform in representing or approximating various functions, the proposed QFT sampler is expected to satisfy the condition (iii), in addition to (i) and (ii). We also emphasize that this QFT sampler or its variant (e.g., with different parametrization to cover the high-frequencies components) is a new type of circuit ansatz in the quantum variational method and might be applicable to other problems such as the quantum generative modeling.

Now we state our bonus theorem; when the dimension of the input to QFT corresponding to the low-frequency components is not large, the proposed QFT sampler can be efficiently simulated with a classical means, using the adaptive measurement technique [16]. This is exactly the direction to explore a classical algorithm that fully makes use of quantum feature, i.e., a quantum-inspired algorithm such as in Ref. [17]. Actually we will show that this quantum-inspired sampler has a certain advantage over some conventional methods, in addition to the clear merit that the system with, e.g., hundreds of qubits is simulable.

The rest of the paper is organized as follows. First, in Sec. II, we describe the general idea of quantum self-learning Monte Carlo. Then Sec. III gives the scheme of QFT sampler and its classical implementation (i.e., the quantum-inspired algorithm based on QFT) as a demonstration of the proof-of-concept of the idea (Sec. III). Section IV is devoted to show detailed numerical simulations of the QFT algorithm for various target probability distributions. Lastly a summary and some discussion for future prospect are given in Sec. V.

Notations. For a complex column vector $\boldsymbol{a}$, the symbols $\boldsymbol{a}^\dagger$ and $\boldsymbol{a}^\top$ represent its complex conjugate transpose and transpose vectors, respectively. Also $\boldsymbol{a}^*$ denotes the elementwise complex column vector of $\boldsymbol{a}$. Hence $\boldsymbol{a}^\dagger = (\boldsymbol{a}^*)^\top$.

II. GENERAL SCHEME OF QUANTUM SELF-LEARNING MONTE CARLO METHOD

A. Classical MH algorithm

Let $p(\boldsymbol{x})$ and $q(\boldsymbol{x})$ be target and proposal probability distributions, respectively. To get a sample from $p(\boldsymbol{x})$, the MH algorithm instead samples from $q(\boldsymbol{x})$ and accepts the result with valid probability determined by the detailed balance conditions. More specifically, assume that we have last accepted a sample $\boldsymbol{r}$ and now obtain a sample $\tilde{\boldsymbol{r}}$ generated from the proposal distribution $q(\boldsymbol{x})$. Then, this sample $\tilde{\boldsymbol{r}}$ is accepted with probability

$$A(\boldsymbol{r}, \tilde{\boldsymbol{r}}) = \min\left\{1, \frac{p(\tilde{\boldsymbol{r}})q(\boldsymbol{r})}{p(\boldsymbol{r})q(\tilde{\boldsymbol{r}})}\right\}, \tag{1}$$

which is called the acceptance ratio. Note that the value $p(\boldsymbol{r})$ is assumed to be easily computable for a given $\boldsymbol{r}$, while its sampling is hard. This procedure is repeated until the number of samples becomes enough large; then these accepted samples are governed by the target distribution $p(\boldsymbol{x})$ due to the detailed balance conditions. Note that, if $q(\boldsymbol{x}) = p(\boldsymbol{x})$, then the acceptance ratio is always exactly 1, which is maximally efficient; but of course this does not happen because $p(\boldsymbol{x})$ is hard to sample while $q(\boldsymbol{x})$ is assumed to be relatively easy to sample.

In the context of self-learning MC, a parametric model of the proposal distribution $q(\boldsymbol{x};\boldsymbol{\theta})$ is considered, with $\boldsymbol{\theta}$ the vector of parameters. The self-learning MC aims to learn the parameters so that $q(\boldsymbol{x};\boldsymbol{\theta})$ moves toward the target $p(\boldsymbol{x})$.

B. General form of the quantum self-learning MH algorithm

Here we describe the quantum sampler executing the self-learning MH algorithm, in the general setting. First, for the initial state $|g^N\rangle = |g\rangle^{\otimes N}$ with $|g\rangle = [1, 0]^\top$ a qubit state, we apply the parametric unitary gate $U(\boldsymbol{\theta})$:

$$|\Psi(\boldsymbol{\theta})\rangle = U(\boldsymbol{\theta})|g^N\rangle,$$

where $\boldsymbol{\theta} \in \mathbb{C}^m$ are the parameters to be tuned. The reason of taking the complex-valued parameters will be made clear in the next section when specializing to the QFT circuit. This state is measured in the computational basis, defining the probability distribution

$$q(\boldsymbol{x};\boldsymbol{\theta}) = |\langle \boldsymbol{x}|\Psi(\boldsymbol{\theta})\rangle|^2 = |\langle \boldsymbol{x}|U(\boldsymbol{\theta})|g^N\rangle|^2, \tag{2}$$

where $\boldsymbol{x}$ is the multidimensional random variable represented by binaries (an example is given in Appendix A). Equation (2) is the proposal distribution of our MH algorithm. Hence, our goal is to update $\boldsymbol{\theta}$ so that $q(\boldsymbol{x};\boldsymbol{\theta})$ is going to approximate the target distribution $p(\boldsymbol{x})$. The learning process for updating $\boldsymbol{\theta}$ is executed via the standard gradient descent method of a loss function, as described below.

First, for a fixed $\boldsymbol{\theta}$, we obtain samples $\boldsymbol{r}_1, \ldots, \boldsymbol{r}_B$ from $q(\boldsymbol{x};\boldsymbol{\theta})$ by the computational-basis measurement. These samples are filtered according to the acceptance probability (1), thus producing samples governed by $p(\boldsymbol{x})$. Note now that, for a given $\boldsymbol{r}$, the value $q(\boldsymbol{r}) = |\langle \boldsymbol{r}|\Psi(\boldsymbol{\theta})\rangle|^2$ must be effectively computed to do this filtering process (recall that the value of $p(\boldsymbol{r})$ is assumed to be easily obtained); this computability indeed depends on the structure of $U(\boldsymbol{\theta})$, but in general we could apply the quantum amplitude estimation algorithm [12,13] to reduce the cost for executing this task. The samples $\boldsymbol{r}_1, \ldots, \boldsymbol{r}_B$ are also used to calculate the gradient descent vector of the loss function $L(\boldsymbol{\theta})$ for updating $\boldsymbol{\theta}$. Noting that $L(\boldsymbol{\theta})$ is real while $\boldsymbol{\theta}$ is complex, its infinitesimal change with respect to $\boldsymbol{\theta}$ is given by

$$\delta L = \left(\frac{\partial L}{\partial \boldsymbol{\theta}}\right)^\top \delta\boldsymbol{\theta} + \left(\frac{\partial L}{\partial \boldsymbol{\theta}^*}\right)^\top \delta\boldsymbol{\theta}^* = \left(\frac{\partial L}{\partial \boldsymbol{\theta}}\right)^\top \delta\boldsymbol{\theta} + \left(\frac{\partial L}{\partial \boldsymbol{\theta}}\right)^\dagger \delta\boldsymbol{\theta}^*.$$

The gradient descent vector for updating the parameter from $\boldsymbol{\theta}$ to $\boldsymbol{\theta}' = \boldsymbol{\theta} + \delta\boldsymbol{\theta}$ is thus given by

$$\boldsymbol{\theta}' = \boldsymbol{\theta} - \alpha\left(\frac{\partial L(\boldsymbol{\theta})}{\partial \boldsymbol{\theta}}\right)^*, \tag{3}$$

where $\alpha > 0$ is the learning coefficient; in fact then $\delta L = -2\alpha\|\partial L/\partial\boldsymbol{\theta}\|^2 \leqslant 0$. In this work, the loss function is set to the following cross entropy between $q(\boldsymbol{x};\boldsymbol{\theta})$ and $p(\boldsymbol{x})$:

$$L(\boldsymbol{\theta}) = -\sum_{\boldsymbol{x}} p(\boldsymbol{x}) \log q(\boldsymbol{x};\boldsymbol{\theta}), \tag{4}$$

which is a standard measure for quantify the similarity of two distributions. The gradient vector of $L(\boldsymbol{\theta})$ can be computed as follows:

$$\begin{aligned}\frac{\partial L(\boldsymbol{\theta})}{\partial \boldsymbol{\theta}} &= -\sum_{\boldsymbol{x}} \frac{p(\boldsymbol{x})}{q(\boldsymbol{x};\boldsymbol{\theta})}\frac{\partial q(\boldsymbol{x};\boldsymbol{\theta})}{\partial \boldsymbol{\theta}} \\ &= -\sum_{\boldsymbol{x}} q(\boldsymbol{x};\boldsymbol{\theta})\frac{p(\boldsymbol{x})}{q(\boldsymbol{x};\boldsymbol{\theta})^2}\frac{\partial q(\boldsymbol{x};\boldsymbol{\theta})}{\partial \boldsymbol{\theta}} \\ &= \lim_{n\to\infty} -\frac{1}{n}\sum_{i=1}^{n} \frac{p(\boldsymbol{r}_i)}{q(\boldsymbol{r}_i;\boldsymbol{\theta})^2}\frac{\partial q(\boldsymbol{r}_i;\boldsymbol{\theta})}{\partial \boldsymbol{\theta}} \\ &\simeq -\frac{1}{B}\sum_{i=1}^{B} \frac{p(\boldsymbol{r}_i)}{q(\boldsymbol{r}_i;\boldsymbol{\theta})^2}\frac{\partial q(\boldsymbol{r}_i;\boldsymbol{\theta})}{\partial \boldsymbol{\theta}}.\end{aligned} \tag{5}$$

Note that

$$\frac{\partial q(\boldsymbol{r};\boldsymbol{\theta})}{\partial \boldsymbol{\theta}} = \left[|\langle \boldsymbol{r}|\frac{\partial U(\boldsymbol{\theta})}{\partial \theta_1}|g^N\rangle|^2, \ldots, |\langle \boldsymbol{r}|\frac{\partial U(\boldsymbol{\theta})}{\partial \theta_m}|g^N\rangle|^2\right]$$

can be directly estimated when $U(\boldsymbol{\theta})$ is composed of Pauli operators with the parameters $\{\theta_i\}$ corresponding to the rotation angles [18].

Here we discuss the notable feature and possible quantum advantage of the quantum sampler for the MH algorithm, by referring to the three conditions mentioned in Sec. I. First, the condition (i) is indeed satisfied because now the sampler is a quantum device that physically produces each measurement result $\boldsymbol{r}$ according to the proposal probability distribution $q(\boldsymbol{x};\boldsymbol{\theta})$, only in a few micro second in the case of superconducting devices. As for the condition (ii), it is in principle possible to effectively compute the probability $q(\boldsymbol{r};\boldsymbol{\theta})$ for a given $\boldsymbol{r}$, as discussed above in this subsection. Lastly for the condition (iii), $q(\boldsymbol{x};\boldsymbol{\theta})$ might be able to represent a wide class of probability distribution, which is even hard to sample via any classical means as mentioned in Sec. I. Realization of this possible quantum advantage of course needs a clever designing of the ansatz $U(\boldsymbol{\theta})$.

III. THE QUANTUM FOURIER TRANSFORM SAMPLER

To show the proof-of-concept of the quantum sampler for self-learning MH algorithm, here we consider a special class of circuit composed of QFT, called the QFT sampler. Importantly, as will be shown, the QFT sampler characterized by a low dimensional parameter vector is classically simulable, while it has an advantage over classical algorithms.

Before describing the scheme, we remark that, in the machine learning community, there have been proposed several Fourier-based neural networks (or neural networks with periodic activation functions) [19–22]; these are in fact motivating studies for why we take the QFT ansatz in this work.

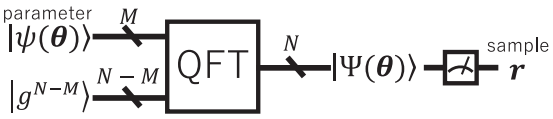

FIG. 1. Schematic illustration of the quantum Fourier transform sampler.

A. One-dimensional QFT sampler

We begin with the one-dimensional QFT sampler; extension to the multidimensional case is discussed in Sec. III C. As illustrated in Fig. 1, this sampler is composed of the QFT operation applied to a N-qubits input state $|\text{in}\rangle = |\psi(\boldsymbol{\theta})\rangle \otimes |g^{N-M}\rangle$, where

$$|\psi(\boldsymbol{\theta})\rangle = \theta_0|0\rangle + \theta_1|1\rangle + \cdots + \theta_{2^M-1}|2^M-1\rangle. \tag{6}$$

$\{|0\rangle, |1\rangle, \ldots, |2^M-1\rangle\}$ is the set of computational basis states in $(\mathbb{C}^2)^{\otimes M} = \mathbb{C}^{2^M}$, e.g., $|0\rangle = |g^M\rangle$. That is, the first M-qubits state contains the parameters $\boldsymbol{\theta} = [\theta_0, \cdots, \theta_{2^M-1}]^\top \in \mathbb{C}^{2^M}$, while the residual $N-M$ qubits are set to $|g\rangle$ states. Note that, if $\boldsymbol{\theta} \in \mathbb{R}^{2^M}$, then the proposal distribution (7) below is limited to an even function, and thus $\boldsymbol{\theta}$ must take complex numbers. Also, the state (6) can be replaced by a parametrized quantum state such as the output of the so-called hardware efficient anzatz [23]. In this case, the state preparation is much easier than to generate (6). Also, we find some studies that demonstrated generating relatively large entangled state [24]. A concern on the use of hardware efficient anzatz is that the representative power of this ansatz is less than (6); but, in the MH setting, the proposal distribution is not required to be very close to the target, implying that a less-expressive or even a mixed input state might be acceptable.

The output of QFT is given by $|\Psi(\boldsymbol{\theta})\rangle = U_{\text{QFT}}|\text{in}\rangle$, where U_{QFT} is the QFT unitary operator whose matrix representation is given by

$$\langle k|U_{\text{QFT}}|j\rangle = \frac{1}{\sqrt{2^N}} e^{i2\pi kj/2^N}.$$

Now the measurement on $|\Psi(\boldsymbol{\theta})\rangle$ in the computational basis yields the probability distribution

$$q_{\text{QFT}}(x;\boldsymbol{\theta}) = |\langle x|\Psi(\boldsymbol{\theta})\rangle|^2 = |\langle x|U_{\text{QFT}}|\text{in}\rangle|^2, \tag{7}$$

where the random variable x is represented with binaries, i.e., $x \in \{0, 1, \cdots, 2^N-1\}$.

Again, the task of self-learning MH is to update $\boldsymbol{\theta}$ so that the proposal distribution $q_{\text{QFT}}(x;\boldsymbol{\theta})$ may approach to the target distribution $p(x)$. To compute the gradient vector (5), let us express $\langle x|U_{\text{QFT}}|\text{in}\rangle$ as

$$\langle x|U_{\text{QFT}}|\text{in}\rangle = \boldsymbol{u}_x^\top \boldsymbol{\theta}, \tag{8}$$

where $\boldsymbol{u}_x$ is the vector composed of the first 2^M elements of the x-th row vector of U_{QFT}; hence the jth element of $\boldsymbol{u}_x$ is

given by

$$(\boldsymbol{u}_x)_j = \frac{1}{\sqrt{2^N}} e^{i2\pi xj/2^N} (j = 0, 1, \cdots, 2^M - 1).$$

Then the partial derivative of $q_{\mathrm{QFT}}(x;\boldsymbol{\theta}) = (\boldsymbol{u}_x^\top \boldsymbol{\theta})^* (\boldsymbol{u}_x^\top \boldsymbol{\theta}) = (\boldsymbol{u}_x^\dagger \boldsymbol{\theta}^*)(\boldsymbol{u}_x^\top \boldsymbol{\theta})$ with respect to $\boldsymbol{\theta}$ is given by

$$\frac{\partial q_{\mathrm{QFT}}(x;\boldsymbol{\theta})}{\partial \boldsymbol{\theta}} = (\boldsymbol{u}_x^\dagger \boldsymbol{\theta}^*) \boldsymbol{u}_x.$$

Hence, from Eq. (5), the gradient vector of the loss function $L(\boldsymbol{\theta})$ can be calculated as follows:

$$\left(\frac{\partial L(\boldsymbol{\theta})}{\partial \boldsymbol{\theta}}\right)^* \simeq -\frac{1}{B} \sum_{i=1}^{B} \frac{p(r_i)}{q_{\mathrm{QFT}}(r_i;\boldsymbol{\theta})^2} (\boldsymbol{u}_{r_i}^\top \boldsymbol{\theta}) \boldsymbol{u}_{r_i}^*, \tag{9}$$

where $\{r_1, r_2, \cdots, r_B\}$ are samples taken from the proposal distribution $q_{\mathrm{QFT}}(x;\boldsymbol{\theta})$. Recall that we filter these samples via the rule (1) and thereby obtain samples that are subjected to the target distribution $p(x)$. In this work, instead of the standard gradient update (3), we take the following momentum gradient descent [25];

$$\boldsymbol{\theta}' = \boldsymbol{\theta} - \alpha \boldsymbol{m}', \quad \boldsymbol{m}' = \mu \boldsymbol{m} + (1-\mu)\left(\frac{\partial L(\boldsymbol{\theta})}{\partial \boldsymbol{\theta}}\right)^*, \tag{10}$$

where α and μ are the learning coefficients; in general, the momentum method offers a more efficient convergence than the standard gradient method, thanks to the momentum effect via $\boldsymbol{m}$, which is the exponential moving average of the stochastic gradient $\partial L/\partial \boldsymbol{\theta}$. The updated vector $\boldsymbol{\theta}'$ is normalized and substituted into Eq. (6) for the next learning stage. Note that Eq. (10) is identical to the standard gradient descent when $\mu = 0$.

Here let us discuss how the basic conditions (i)–(iii) are reasonably fulfilled by the QFT sampler. First, as mentioned before, the QFT sampler enables us to obtain samples enough fast, thanks to the feature of quantum devices, and thus it satisfies the condition (i). Also the probability $q_{\mathrm{QFT}}(r;\boldsymbol{\theta})$ with given r is obtained via estimating the inner product (8), and thereby the condition (ii) is satisfied. As for the condition (iii), in view of the fact that several Fourier-based neural networks are employed in machine learning [19–22], we expect that the QFT sampler may also be able to well approximate the target $p(x)$, hopefully better than any classical means due to the sampling supremacy property [6–10].

B. The quantum-inspired Fourier sampler

Now, one might think that the QFT sampler is still out of reach, even in the case of medium-size quantum devices with, e.g., $N = 100$ qubits. However, remarkably, the QFT sampler can be realized in a classical digital computer as long as N^2 and 2^M are not too large (for example, 2^M is polynomial size of N); that is, in this regime, this is a quantum-inspired algorithm that can deal with even a random variable on $2^N = 2^{100}$ discrete elements. The trick relies on the use of the adaptive measurement technique [16], which enables us to sample only by applying $O(2^M + N)$ operations on a classical computer; see Appendix B for a detailed explanation. This means that, therefore, the QFT sampler fulfills the condition (i), even as a classical computer. Also we can immediately compute the probability $q_{\mathrm{QFT}}(r;\boldsymbol{\theta})$ with given r by just calculating the inner product (8) which costs of the order $O(2^M)$, and thereby the condition (ii) is satisfied as long as 2^M is not too large.

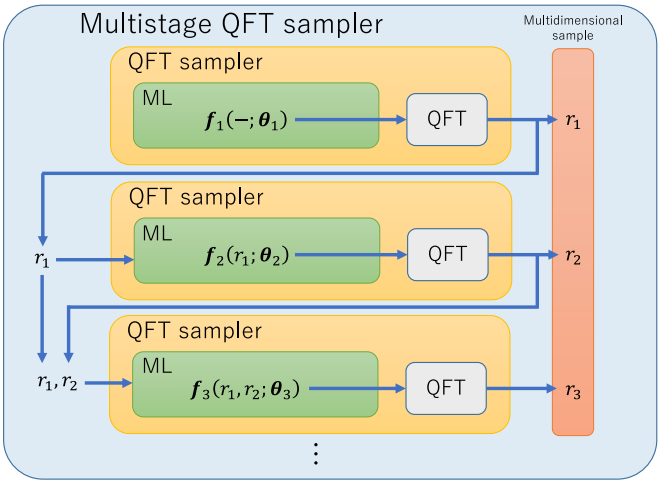

FIG. 2. Schematic illustration of the multistage QFT sampler, where ML means machine learning.

Finally, we discuss a possible advantage of our QFT sampler, within a regime of classical sampling method. As a classical Fourier-based proposal distribution, one might think to employ the fast Fourier transform (FFT). However, to deal with a variable with 2^N discretized elements, FFT needs $\mathcal{O}(N2^N)$ operations, while QFT can realize the same operation only with $\mathcal{O}(N^2)$ gates. As is well known, this does not mean a quantum advantage in the typical application scene such as signal processing, because all the amplitude of the QFT-transformed state cannot be effectively determined [26]. On the other hand, the presented scheme only requires sampling and thus determining $\{q_{\mathrm{QFT}}(r_i;\boldsymbol{\theta})\}_{i=1,\ldots,B}$ rather than all the elements $\{q_{\mathrm{QFT}}(x;\boldsymbol{\theta})\}_{x=0,\ldots,2^N-1}$. Hence we could say that the developed quantum-inspired algorithm has a solid computational advantage over the known classical algorithm, in the problem of determining a Fourier-based proposal distribution for the MH algorithm.

C. Multistage QFT sampler for multidimensional distributions

The QFT sampler discussed above is able to sample only from a one-dimensional distribution. To deal with the multidimensional distributions over D random variables $x_1, \ldots, x_D$, a straightforward way is to employ the D-dimensional QFT on ND-qubits. In this case, the input state to the QFT part is the product of MD-qubits state $|\psi(\boldsymbol{\theta})\rangle$ and the trivial $|g^{D(N-M)}\rangle$. Then the adaptive measurement scheme is implemented by repeatedly operating 2^{MD}-dimensional matrices on the input vector, which immediately becomes intractable by classical computing even for small M and D such as $(M, D) = (4, 10)$. In other words, this is the situation where the quantum implementation of QFT circuit might serve as a genuine efficient sampler.

On the other hand, for the proof-of-concept, in this paper we develop a multistage QFT sampler composed of single QFT samplers illustrated in Fig. 2. This scheme can be implemented in a classical way as in the one-dimensional case, based on the adaptive measurement technique shown

in Sec. III B together with the classical hierarchical post-processing (the machine learning part) to make correlations between the variables. To see the idea, let us begin with the case where the D random variables $x_1, \ldots, x_D$ are independent with each other and subjected to the D-dimensional independent target distribution $p(\boldsymbol{x})$. In this case, the following proposal distribution might work:

$$q(\boldsymbol{x};\boldsymbol{\theta}) = \prod_{k=1}^{D} q_{\mathrm{QFT}}(x_k;\boldsymbol{\theta}_k), \tag{11}$$

where $\boldsymbol{x} = [x_1, \cdots, x_D]^\top$ is the D-dimensional random variable represented by binaries $x_k \in \{0, 1, \ldots, 2^N - 1\}$. Here $q_{\mathrm{QFT}}(x_k;\boldsymbol{\theta}_k)$ is a one-dimensional QFT sampler parametrized by the 2^M-dimensional complex vector $\boldsymbol{\theta}_k$; we summarize these vectors to $\boldsymbol{\theta} = [\boldsymbol{\theta}_1^\top, \cdots, \boldsymbol{\theta}_D^\top]^\top$.

Now based on Eq. (11) we construct the multistage QFT sampler for dealing with a nonindependent multidimensional proposal distribution. The point is that, as illustrated in Fig. 2, the parameter vector $\boldsymbol{\theta}_k$ specifying the kth one-dimensional QFT sampler in Eq. (11) is replaced by a vector of parametrized functions of random variables up to the $(k-1)$th stage, i.e., $\boldsymbol{f}_k(x_1, \cdots, x_{k-1};\boldsymbol{\theta}_k)$, whose control parameter $\boldsymbol{\theta}_k$ is to be repeatedly modified through the learning process. Hence the proposal distribution is given by

$$q(\boldsymbol{x};\boldsymbol{\theta}) = \prod_{k=1}^{D} q_{\mathrm{QFT}}(x_k;\boldsymbol{f}_k(x_1, \cdots, x_{k-1};\boldsymbol{\theta}_k)). \tag{12}$$

Note that this is simply a representation of the joint probability distribution over the multidimensional random variables $\boldsymbol{x} = [x_1, \cdots, x_D]^\top$ via the series of conditional probabilities. The gradient of the cross entropy (4) is derived in the same way as the one-dimensional case;

$$\left(\frac{\partial L(\boldsymbol{\theta})}{\partial \boldsymbol{\theta}_k}\right)^* \simeq -\frac{1}{B}\sum_{i=1}^{B} \frac{p(\boldsymbol{r}_i)}{q(\boldsymbol{r}_i;\boldsymbol{\theta})} \frac{(\boldsymbol{u}_{r_k}^\top \boldsymbol{f}_k)\boldsymbol{u}_{r_k}^*}{q_{\mathrm{QFT}}(r_k;\boldsymbol{f}_k)} \frac{\partial \boldsymbol{f}_k}{\partial \boldsymbol{\theta}_k}, \tag{13}$$

where $\{\boldsymbol{r}_1, \cdots, \boldsymbol{r}_B\}$ are samples produced from the proposal distribution $q(\boldsymbol{x};\boldsymbol{\theta})$. Note that each sample $\boldsymbol{r} = [r_1, \cdots, r_D]^\top$ is formed from r_k produced from the kth one-dimensional QFT sampler, as shown in Fig. 2. Also, as in the one-dimensional case (8), the vector $\boldsymbol{u}_{x_k}$ is defined through $\langle x_k|U_{\mathrm{QFT}}|\mathrm{in}\rangle = \boldsymbol{u}_{x_k}^\top \boldsymbol{f}_k(x_1, \cdots, x_{k-1};\boldsymbol{\theta}_k)$. This gradient vector (13) is used to update each parameter vector $\boldsymbol{\theta}_k$ using the momentum gradient descent (10), which is now of the form

$$\boldsymbol{\theta}_k' = \boldsymbol{\theta}_k - \alpha \boldsymbol{m}_k', \quad \boldsymbol{m}_k' = \mu \boldsymbol{m}_k + (1-\mu)\left(\frac{\partial L(\boldsymbol{\theta})}{\partial \boldsymbol{\theta}_k}\right)^*,$$

where the learning coefficients (α, μ) do not depend on k for simplicity. This eventually constitutes the total gradient descent vector minimizing the loss function (4) and accordingly move the proposal distribution (12) toward the target $p(\boldsymbol{x})$. Also recall that we use Eq. (1) to filter $\{\boldsymbol{r}_1, \cdots, \boldsymbol{r}_B\}$ to obtain samples that are subjected to $p(\boldsymbol{x})$.

In this work, we examine the following four models of the parameterized function $\boldsymbol{f}_k(x_1, \cdots, x_{k-1};\boldsymbol{\theta}_k)$.

(1) Identity (Id) model:

$$\boldsymbol{f}_k(x_1, \cdots, x_{k-1};\boldsymbol{\theta}_k) = \mathrm{Norm}(\boldsymbol{\theta}_k) = \frac{\boldsymbol{\theta}_k}{\|\boldsymbol{\theta}_k\|},$$

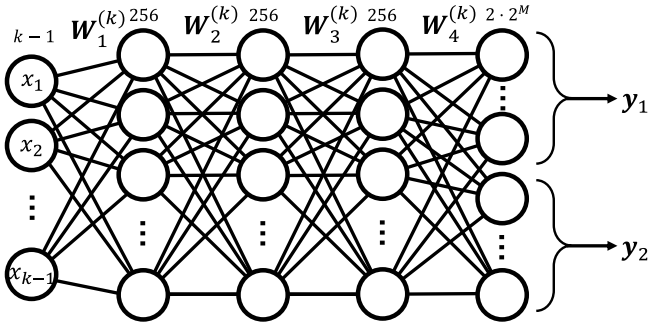

FIG. 3. Fully connected neural network.

where $\|\boldsymbol{\theta}\| = \sqrt{\boldsymbol{\theta}^\dagger \boldsymbol{\theta}}$. The Norm function ensures the normalization of $|\boldsymbol{\psi}(\boldsymbol{\theta})\rangle$. This leads to the independent proposal distribution (11).

(2) Linear basis linear regression (LBLR) model:

$$\boldsymbol{f}_k(x_1, \cdots, x_{k-1};\boldsymbol{\theta}_k) = \mathrm{Norm}\big(\boldsymbol{w}_1^{(k)} x_1 + \cdots + \boldsymbol{w}_{k-1}^{(k)} x_{k-1} + \boldsymbol{b}^{(k)}\big),$$

where in this case the parameters to be learned are the collection of 2^M-dimensional complex vectors, $\boldsymbol{\theta}_k = \{\boldsymbol{w}_1^{(k)}, \boldsymbol{w}_2^{(k)}, \cdots, \boldsymbol{w}_{k-1}^{(k)}, \boldsymbol{b}^{(k)}\}$. This model outputs a linear combination of their input argument.

(3) Nonlinear basis linear regression (NBLR) model:

$$\boldsymbol{f}_k(x_1, \cdots, x_{k-1};\boldsymbol{\theta}_k) = \mathrm{Norm}\left(\sum_{j=1}^{J} \boldsymbol{w}_j^{(k)} \phi_j^{(k)}(x_1, \cdots, x_{k-1}) + \boldsymbol{b}^{(k)}\right),$$

where $\boldsymbol{\theta}_k = \{\boldsymbol{w}_1^{(k)}, \boldsymbol{w}_2^{(k)}, \cdots, \boldsymbol{w}_J^{(k)}, \boldsymbol{b}^{(k)}\}$ are the collection of 2^M-dimensional complex vectors to be learned, and $\{\phi_j^{(k)}(\cdot)\}$ are fixed nonlinear basis functions. Different set of nonlinear basis functions should be chosen for a different target distribution.

(4) Neural network (NN) model:

$$\boldsymbol{f}_k(x_1, \cdots, x_{k-1};\boldsymbol{\theta}_k) = \mathrm{Norm}(h(\boldsymbol{y}^{(k)})),$$
$$\boldsymbol{y}^{(k)} = g_4^{(k)} \circ \mathrm{Sg} \circ g_3^{(k)} \circ \mathrm{Sg} \circ g_2^{(k)} \circ \mathrm{Sg} \circ g_1^{(k)}(\boldsymbol{x}),$$

where $\circ$ denotes $g_1 \circ g_2(\boldsymbol{x}) = g_1(g_2(\boldsymbol{x}))$. This is a fully connected four-layer neural network with input $\boldsymbol{x} = (x_1, \cdots, x_{k-1})$, shown in Fig. 3. Here $g_j^{(k)}(\boldsymbol{x}) = \boldsymbol{W}_j^{(k)}\boldsymbol{x} + \boldsymbol{b}_j^{(k)}$ is a linear transformation between the layers; $(\boldsymbol{W}_2^{(k)}, \boldsymbol{W}_3^{(k)})$ are 256×256 real parameter matrices, and $(\boldsymbol{W}_1^{(k)}, \boldsymbol{W}_4^{(k)})$ are $256 \times (k-1)$ and $2^{M+1} \times 256$ real parameter matrices, respectively; also $(\boldsymbol{b}_1^{(k)}, \boldsymbol{b}_2^{(k)}, \boldsymbol{b}_3^{(k)})$ are 256 dimensional real parameter vectors, and $\boldsymbol{b}_4^{(k)}$ is a 2^{M+1} dimensional real parameter vector. Hence $\boldsymbol{\theta}_k = (\boldsymbol{W}_1^{(k)}, \cdots, \boldsymbol{W}_4^{(k)}, \boldsymbol{b}_1^{(k)}, \cdots, \boldsymbol{b}_4^{(k)})$ are the parameters to be optimized. $\mathrm{Sg}(\boldsymbol{x})$ is the sigmoid function whose ℓth component is defined as $1/(1+e^{-x_\ell})$. Finally, for the output vector $\boldsymbol{y} = [\boldsymbol{y}_1^\top, \boldsymbol{y}_2^\top]^\top$, where $\boldsymbol{y}_1$ and $\boldsymbol{y}_2$ are the 2^M real vectors, the function h acts on it and produces a complex vector $h(\boldsymbol{y}) = \boldsymbol{y}_1 + i\boldsymbol{y}_2$.

IV. SIMULATION RESULTS

In this section, we numerically examine the performance of the QFT sampler for several target distributions. For this purpose, the following criteria are used for the evaluation.

First, the cross entropy (4) is employed to see if the proposal distribution approaches toward the target. Note that in general the cross entropy does not decrease to zero even in the case when the proposal distribution coincides with the target. Hence, to evaluate the distance between the two probability distributions, we use the following Wasserstein distance:

$$\mathrm{W}(P, Q) = \sup_{\|f\|_L \leqslant 1} \{\mathbf{E}_{x\sim P}[f(x)] - \mathbf{E}_{x\sim Q}[f(x)]\},$$

where the supremum is taken over f contained in the set of functions satisfying the one-Lipschitz constraint. Note that here the Kullback Leibler divergence is not used, because it is applicable only when the supports of the two distributions coincide.

We also use the average acceptance ratio to evaluate the efficiency of the sampler. Mathematically it is defined as the expectation value of Eq. (1), but in the simulation we simply take the ratio of the number of accepted samples to the number of all samples generated.

The source code for the simulation is available in the github repository [27].

A. One-dimensional case

First we discuss the performance of the one-dimensional QFT sampler. The QFT sampler is composed of $N = 10$ qubits wherein $M = 4$ qubits are used for parametrization; see Fig. 1. The total learning step is 40 000, where in each step $B = 32$ samples are used to compute the gradient vector (9). The learning coefficients in the momentum update rule (10) are chosen as $\alpha = 0.01$ and $\mu = 0.9$. In the upper panels of Fig. 4, five types of target distributions (red broken lines) and snapshots of proposal distributions in each 1000 steps (black solid lines) as well as the convergent distributions (blue solid lines) are plotted. In the lower three panels in each subfigures, CE (Cross entropy), WA (Wasserstein distance), and AC (Average acceptance ratio) are plotted, demonstrating that in each case the proposal distribution $q_{\mathrm{QFT}}(x;\boldsymbol{\theta})$ is approaching to the target $p(x)$, and accordingly AC increases. Note that in the case D, a wavelike structure still remains in the convergent proposal distribution, because of the absence of qubits corresponding to the high-frequency components. However, as mentioned in Sec. I, this is not a serious issue in the framework of MH, which does not require a precise approximation of the target distribution via the proposal one but only needs a proposal distribution realizing relatively high acceptance ratio. In fact the lower panel of D in Fig. 4 shows about 30% improvement in the acceptance ratio.

We now add a discussion on the advantage of our self-learning MC compared to two conventional MC methods, in terms of the acceptance ratio (1) and the decay ratio of autocorrelation function. The first conventional method is the case where the proposal distribution is a uniform distribution over the entire space; this realizes a trivial global update in the sense that a sample $\tilde{\boldsymbol{r}}$ is taken without respect to the last accepted sample $\boldsymbol{r}$. Hence, the autocorrelation between the samples becomes almost zero only at one algorithm step in all experiments, and this is the same for our self-learning MC. However, the trivial global update strategy shows the acceptance ratio with only 0.72 (a), 0.14 (b), 0.52 (c), 0.50 (d), and 0.53 (e), which is significantly lower than those of our method where the acceptance ratio is bigger than 0.9 in most experiments. Hence, our method outperforms this trivial global update under the criterion of acceptance ratio. The second conventional method is the case where the proposal distribution is a Gaussian distribution centered at the last accepted sample $\boldsymbol{r}$; in this case, clearly, a new sample $\tilde{\boldsymbol{r}}$ appears around $\boldsymbol{r}$, and hence this is a local update strategy of proposal distribution. In this case, there is a trade-off between the acceptance ratio and the decay rate of autocorrelation per acceptance step; that is, if the standard deviation of the Gaussian distribution, σ, is small, then $\tilde{\boldsymbol{r}}$ and $\boldsymbol{r}$ are close, meaning that the autocorrelation does not change a lot whereas the acceptance ratio is large; also the opposite effect is observed when σ is large. In this work, σ was chosen so that the acceptance ratio is 0.9 in each experiment; then the numbers of adoptions such that the decay ratio of autocorrelation function is less than e^{-1}, are 20 (a), 20 (b), 100 (c), 10 (d), and 300 (e), meaning that the convergence of the proposal distribution to the target is much slower than our method (where the autocorrelation is almost zero). Therefore our method is superior to this standard local update strategy, under the criterion of convergence speed.

B. Two-dimensional case

Next we study several two-dimensional (i.e., $D = 2$) target distributions, which are shown in the top right panels from A to G in Fig. 5. In this case, the proposal distribution (12) is of the form

$$q(\boldsymbol{x};\boldsymbol{\theta}) = q_{\mathrm{QFT}}(x_2, \boldsymbol{f}_2(x_1;\boldsymbol{\theta}_2))q_{\mathrm{QFT}}(x_1, \boldsymbol{f}_1(\boldsymbol{\theta}_1)), \tag{14}$$

where $\boldsymbol{f}_1(\boldsymbol{\theta}_1)$ is set to be $\boldsymbol{f}_1(\boldsymbol{\theta}_1) = \mathrm{Norm}(\boldsymbol{\theta}_1)$. As for $\boldsymbol{f}_2(x_1;\boldsymbol{\theta}_2)$, we study the four models described in Sec. III C; i.e., Id, LBLR, NBLR, and NN models. The learning coefficients of the momentum gradient method are set to $\alpha = 0.01$ and $\mu = 0.9$ for all the cases, except that $\alpha = 0.001$ is chosen in the NN model for the cases from (a) to (f). Also the two one-dimensional QFT samplers in Eq. (14) are both composed of $N = 10$ and $M = 4$ qubits, for the cases from (a) to (f), while $N = 10$ and $M = 5$ for the case (g). The basis functions in the NBLR model $\boldsymbol{f}_2(x_1;\boldsymbol{\theta}_2)$ are chosen as follows:

$$\phi_1(x) = \bar{x},\ \phi_2(x) = \bar{x}^2,\ \phi_3(x) = \bar{x}^3,\ \phi_4(x) = \sqrt{|\bar{x}|},$$

where $\bar{x}$ is defined as $\bar{x} = 2x/(2^N - 1) - 1$. The total learning step is 40 000 for the cases from (a) to (f) and 400 000 for the case (g). Finally, for all cases, $B = 32$ samples are taken to compute the gradient in each step. With this setting, the proposal distributions at the final learning step corresponding to the four models (Id, LBLR, NBLR, and NN models from the left to right) are shown in the top panel of Fig. 5, and the change of the figure of merits (CE, WA, and AC) are also provided in the bottom panel.

First, as expected, the Id model can only approximate the distribution having no correlation in the space dimension, i.e., the case of (c). On the other hand, the LBLR model acquires a distribution close to the target, for the cases (b), (d), (e), (f),

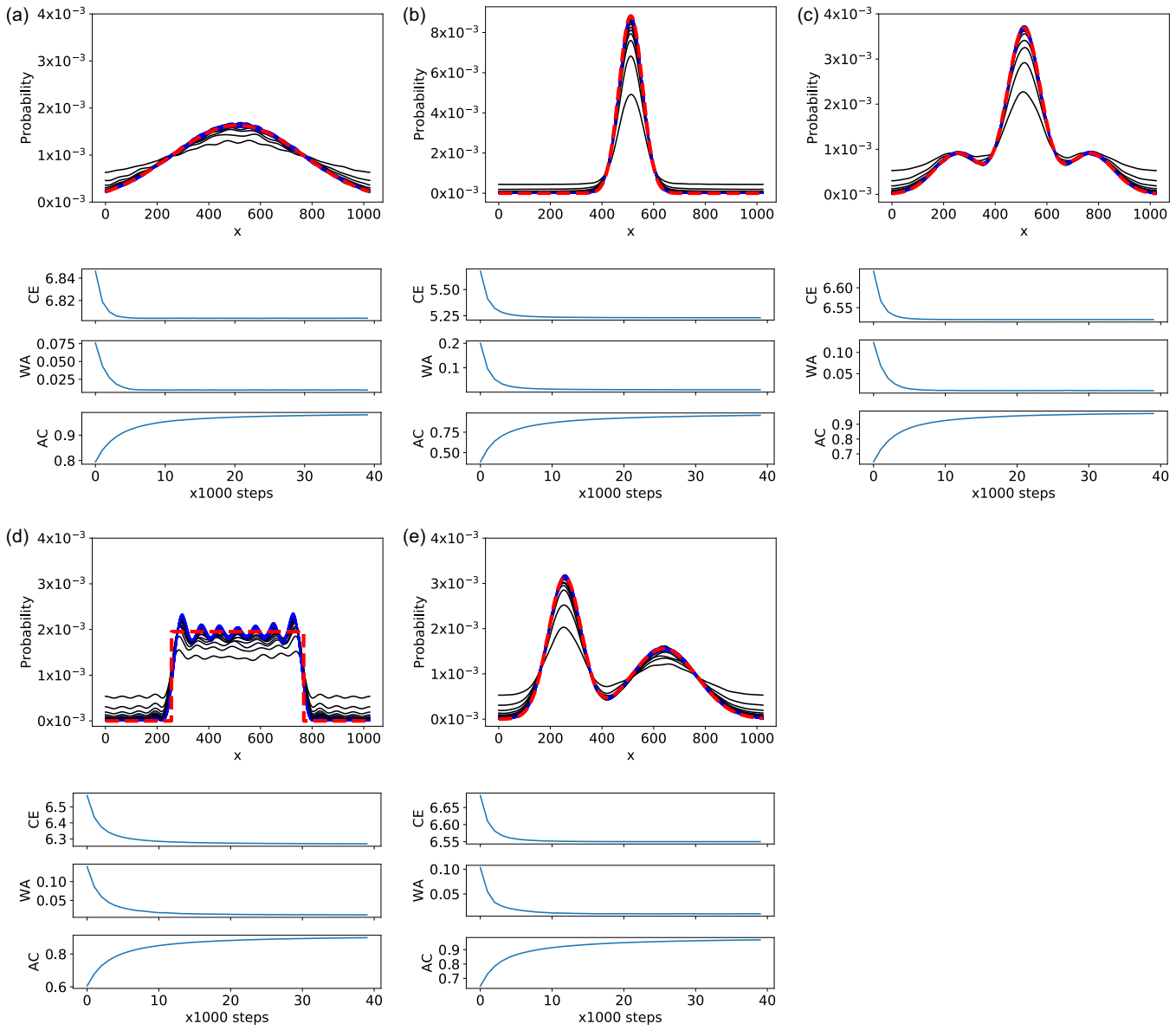

FIG. 4. Performance of the one-dimensional QFT samplers, for the five types of target distributions shown with the red dotted line in the upper panel in each subfigures from (a) to (g). In the upper panel in each subfigure, some snapshots of the proposal distribution generated from the QFT sampler are shown with solid lines. In the bottom three panels in each subfigure, the convergence trend of the three types of figure of merits are shown.

in addition to (c). The figure of merits, CE, WA, and AC, also reflect this fact. [The blue and orange lines in the cases (a) and (c) almost coincide.] It is notable that the simple LBLR model greatly improves the performance obtained with the Id model.

To further approach to the cases of (a) and (g), some nonlinearities need to be introduced, as demonstrated by the NBLR and NN models. In particular, for all the cases from A to (g), the NBLR model shows almost the same level of performance as the NN model, which is also supported by the figure of merits. Considering the fact that there are some jumps in WA and CE in the NN model, and the fact that the NN model costs a lot in the learning process, our conclusion is that the NBLR is the most efficient model in our case-study.

Lastly, we do the same discussion as that given at the end of Sec. IV A, to compare our method to the trivial global update strategy using the uniform proposal distribution and the standard local one using the Gaussian proposal distribution. Again, the acceptance ratio and the number of acceptances with autocorrelation functions below e^{-1}, are computed. Then, for the case of uniform distribution, although the autocorrelation is less than e^{-1} at one acceptance step for all simulations, the acceptance rates are 0.31 (a), 0.36 (b), 0.60 (c), 0.34 (d), 0.34 (e), 0.46 (f), and 0.23 (g), which are significantly less than the value of 0.9 achieved with our method. Also, for the case of Gaussian distribution, the number of acceptances with autocorrelation below e^{-1} is about 400 (a), 30 (b), 40 (c), 8000 (d), 50 (e), 10 (f), and 300 (g), when the Gaussian shape is chosen so that the acceptance ratio is 0.9; on the other hand, the autocorrelation of the proposed method is less than e^{-1} at one acceptance step while the acceptance

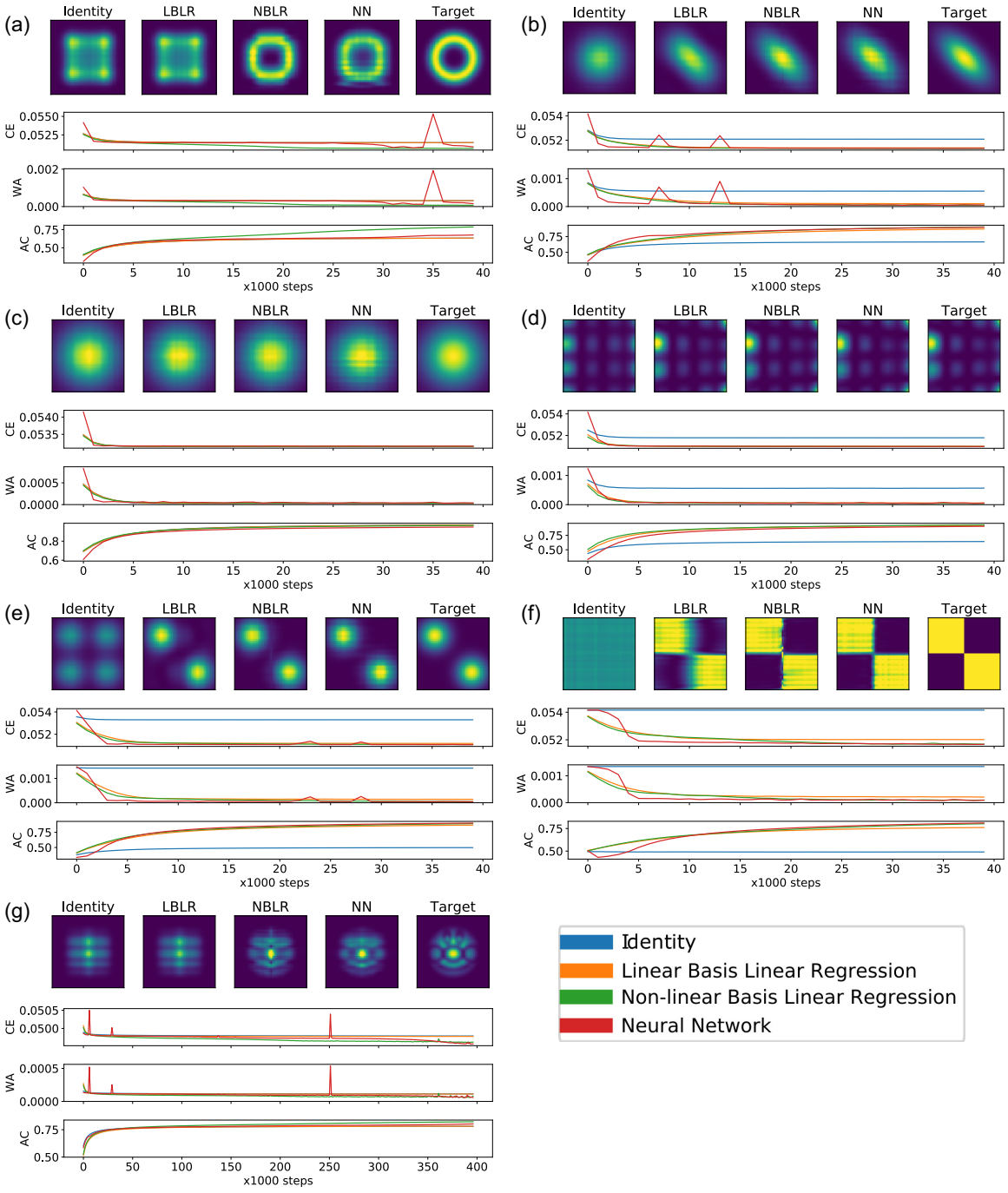

FIG. 5. Performance of the two-dimensional QFT samplers, for the five types of target distributions shown in the upper right panel in each subfigures from (a) to (g). In the upper left four panels in each subfigure, the four proposal distributions generated from the QFT sampler at the final step are shown. In the bottom three panels in each subfigure, the convergence trend of the three types of figure of merits are shown.

ratio is about 0.9 on average. To conclude, hence, the proposed method is superior to those two conventional methods in terms of the acceptance ratio and the convergence speed.

C. Application to a molecular simulation

The last case-study is focused on the stochastic dynamics of two atoms obeying the Lennard-Jones (LJ) potential field, which is often employed in the field of molecular simulation. This problem requires sampling from the Boltzmann distribution

$$p(\boldsymbol{r}_1, \boldsymbol{r}_2) \propto \exp\{-\beta \mathrm{LJ}(\|\boldsymbol{r}_1 - \boldsymbol{r}_2\|)\},$$

where $\mathrm{LJ}(a) = a^{-12} - a^{-6}$ and $\beta = 0.1$ is the inverse temperature. $\boldsymbol{r}_1 = [r_{1x}, r_{1y}, r_{1z}]^\top$ and $\boldsymbol{r}_2 = [r_{2x}, r_{2y}, r_{2z}]^\top$ are the vectors of position variables of each atom. We apply the six-stage QFT sampler to generate a proposal distribution for approximating this target distribution.

The QFT sampler is configured by conditioning the samples in the order $r_{1x} \to r_{2x} \to r_{1y} \to r_{2y} \to r_{1z} \to r_{2z}$; for in-

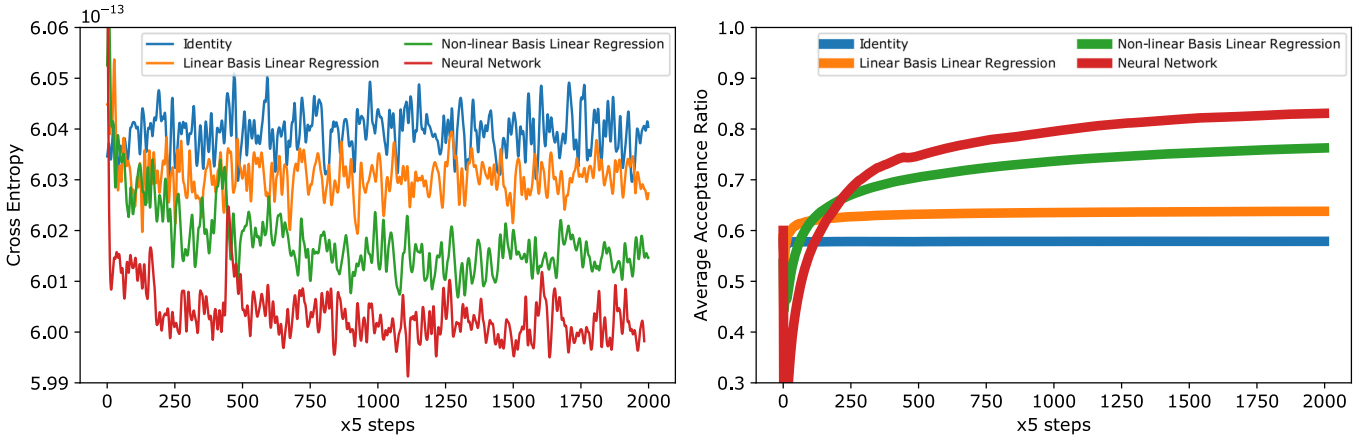

FIG. 6. Convergence trend of the cross entropy (4) between the proposal distribution generated from the QFT sampler to the target six-dimensional Boltzmann distribution (left) and the average acceptance ratio (1) (right).

stance, $q_{\mathrm{QFT}}(r_{1y}, \boldsymbol{f}_{1y}(r_{1x}, r_{2x}; \boldsymbol{\theta}_{1y}))$ is conditioned on (r_{1x}, r_{2x}). The output of the QFT sampler, $r_i \in \{0, 1, \cdots, 2^N - 1\}$, is rescaled to $r_i/0.7(2^N - 1)$. We again employ the four models (Id, LBLR, NBLR, and NN models) as the conditioning functions, where $\boldsymbol{f}_1(\boldsymbol{\theta}_1)$ is set to be $\boldsymbol{f}_1(\boldsymbol{\theta}_1) = \mathrm{Norm}(\boldsymbol{\theta}_1)$. Each QFT sampler is composed of $N = 10$ and $M = 4$ qubits, and the total learning step is 10 000, whereas $B = 1024$ samples are used to compute the gradient in each step. The learning coefficient is $\alpha = 10$ for the Id, LBLR, and NBLR models, while $\alpha = 0.001$ for the NN case; in each case, the momentum method with $\mu = 0.9$ is employed to update the parameters. The basis functions $\{\phi_j^{(k)}(x_1, x_2, \ldots, x_{k-1})\}$ for constructing the NBLR model are set to all the coefficients (except for the constant) of the third-order polynomial function $(1 + \bar{x}_1 + \bar{x}_2 + \cdots + \bar{x}_{k-1})^3$ with $\bar{x} = 2x/(2^N - 1) - 1$.

Figure 6 shows the change of CE and AC; WA was not calculated due to its heavy computational cost. We then find that, similar to the two-dimensional case, the NBLR and NN models succeed in improving both CE and AC, although more learning steps are necessary compared to the previous cases. Note also that the Id and LBLR models are almost not updated, implying the validity to introduce nonlinearities in the model of QFT sampler.

V. CONCLUSION

This paper provided a new self-learning METROPOLIS-Hastings algorithm based on quantum computing and an important subclass of this sampler that uses the QFT. This QFT sampler is shown to be classically simulable when the dimension of the input to QFT corresponding to the low-frequency components is not large. The effectiveness of this quantum inspired method is supported by several numerical simulations. There are a lot of rooms to be investigated more, such as the choice of the optimizer and the conditioning function for constructing the multistage QFT sampler. In particular, in extending to the case of multidimensional distribution, a completely different schematic than the proposed multistage QFT sampler may be necessary.

Although the QFT sampler offers a certain advantage over some classical sampling schemes as discussed at the end of Sec. III A, of course, this quantum inspired algorithm is not what fully makes use of the true power of quantum computation. The real goal is to establish a genuine quantum sampler, which may provide a faster sampling with higher acceptance ratio than any conventional classical method. Such a direction is in fact found in the literature [28,29], which are though far beyond the reach of current available devices.

In our scenario, the D-dimensional QFT sampler on ND-qubits mentioned at the beginning of Sec. III C might be a candidate for such a genuine quantum sampler. As discussed there, the input state to the QFT part, $|\psi(\boldsymbol{\theta})\rangle$, is generated on MD-qubits, and then the classical simulation on 2^{MD}-dimensional matrices via the adaptive measurement technique immediately becomes intractable. On the other hand, the recent rapid advancement of quantum devices suggests that a relatively large quantum device with, e.g., $N \times D = 10 \times 100$ qubits might be hopefully within reach within the next decade. Note that this qubit size is much less demanding than the case of Shor's algorithm, which may need 20 million qubits for the execution [30]. Of course the coherence time is a serious issue in such large-size quantum devices. However, as mentioned in Sec. I, the point of our quantum Metropolis Hastings algorithm is that it does not need to generate a proposal distribution very close to the target, but rather it requires only a relatively high acceptance ratio; hence the algorithm with a mixed state may still work, which will be investigated as a future work. We here add a remark that the input state $|\psi(\boldsymbol{\theta})\rangle$ can be realized as the output of a hardware efficient ansatz over MD-qubits, meaning that the number of parameters of $|\psi(\boldsymbol{\theta})\rangle$ is the polynomial order of MD [usually $\mathcal{O}(MD)$ or $\mathcal{O}(M^2D^2)$]. Also in this framework the learning process does not involve complex conditioning functions depicted in Fig. 2, but it can be simply performed by, e.g., the gradient descent of the cost. For these reasons, we consider that the proposed quantum self-learning Monte Carlo method would be scalable in dealing with multidimensional problems and thereby serve as a useful tool for practical application in the future when a large-scale real quantum device is available.

ACKNOWLEDGMENTS

This work was supported by IPA MITOU Target Program. K.F. and N.Y. are supported by the MEXT Quantum

Leap Flagship Program Grant No. JPMXS0118067394 and JPMXS0118067285, respectively.

APPENDIX A: QUANTUM CIRCUIT PRODUCING A MULTIDIMENSIONAL PROBABILITY DISTRIBUTION

The multidimensional probability distribution of random variables $\boldsymbol{x} = (x_1, \cdots, x_n)$ is simply obtained by measuring a state $|\Psi\rangle \in \otimes_{i=1}^{n} \mathcal{H}_i$ in the computational basis $|\boldsymbol{x}\rangle = \otimes_{i=1}^{n} |x_i\rangle$, i.e., $p(\boldsymbol{x}) = |\langle \boldsymbol{x}|\Psi\rangle|^2$. The following is an example of two-dimensional probability distribution such that the joint probability is explicitly represented via the conditional probability. Let $|\phi_1\rangle \otimes |\phi_2\rangle$ be an initial state on $\mathcal{H}_1 \otimes \mathcal{H}_2$. We consider a quantum circuit of the form $\boldsymbol{U} = \sum_x |x\rangle\langle x| \otimes U_x$, where $|x\rangle$ is a computational basis state of $\mathcal{H}_1$, and U_x is a unitary operator conditioned on the value x; that is, $\boldsymbol{U}$ is a sum of controlled unitaries. Then the output probability distribution of the measurement result on the state $|\Psi\rangle = \boldsymbol{U}|\phi_1\rangle \otimes |\phi_2\rangle$, in the computational basis $|\boldsymbol{x}\rangle = |x_1\rangle|x_2\rangle$, is given by

$$p(\boldsymbol{x}) = |\langle x_1|\langle x_2|\boldsymbol{U}|\phi_1\rangle|\phi_2\rangle|^2 = |\langle x_1|\phi_1\rangle|^2 |\langle x_2|U_{x_1}|\phi_1\rangle|^2.$$

Thus, in this case, the joint probability $p(\boldsymbol{x}) = p(x_1, x_2)$ is explicitly given as a product of the conditional probability $p(x_2 \,|\, x_1) = |\langle x_2|U_{x_1}|\phi_1\rangle|^2$ and $p(x_1) = |\langle x_1|\phi_1\rangle|^2$.

APPENDIX B: CLASSICAL SIMULATION OF THE QFT SAMPLER VIA ADAPTIVE MEASUREMENT

It is known that the a variety class of quantum circuits composed of the QFT can be simulated on a classical computer [16]. This fact is indeed applied to our case, as described here. The point is twofold; one is that the input state to QFT is now of the form

$$|\mathrm{in}\rangle = |\psi(\boldsymbol{\theta})\rangle \otimes |g^{N-M}\rangle = [\theta_0, \ldots, \theta_{2^M-1}, 0, \ldots, 0]^\top,$$

where $|\psi(\boldsymbol{\theta})\rangle = [\theta_0, \ldots, \theta_{2^M-1}]^\top$ is a relatively small quantum state whose entries can be efficiently determined. The other is that the circuit is terminated with the QFT part, meaning that the QFT is immediately followed by the measurement process.

Here we explain a detailed classical algorithm for simulating our circuit in the case of $N = 4$ and $M = 2$; hence $|\mathrm{in}\rangle = |\psi(\boldsymbol{\theta})\rangle \otimes |g\rangle \otimes |g\rangle$ with $|\psi(\boldsymbol{\theta})\rangle$ a two-qubits state. In this case, the original quantum circuit to implement the QFT sampler is shown in Fig. 7(a). The main body of this circuit is QFT, which consists of the Hadamard gates denoted by H and the controlled-R_n gates which rotate the target qubit by acting $R_n = \exp[-i2\pi\sigma_z 2^{-(n+1)}]$ on it iff the control qubit is $|e\rangle = [0, 1]^\top$. The output of QFT is measured in the computational basis, producing the binary output sequence $b_1 b_2 b_3 b_4$

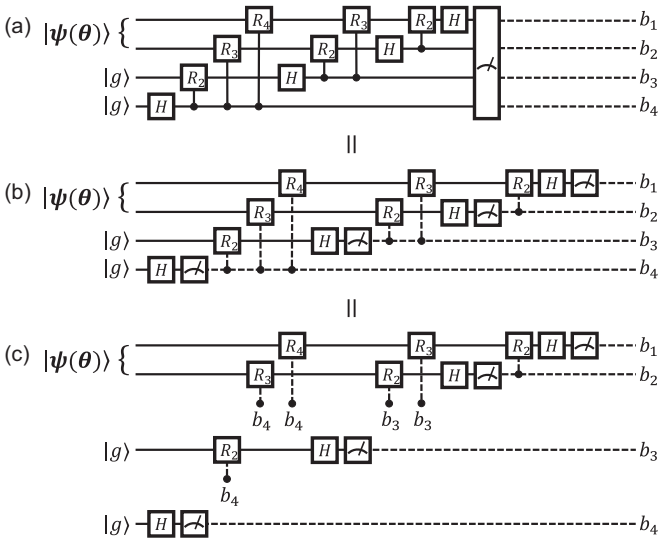

FIG. 7. (a) Original circuit implementation of the QFT sampler on a quantum computer. (b) Equivalent circuit implementation of the QFT sampler, in which all the controlled rotation gates are replaced with the feedforward operations. The solid and dashed lines denote quantum and classical operations, respectively. (c) Equivalent circuit representation of the circuit (b), emphasizing that the change of the first two-qubit (possibly entangled) state $|\psi(\boldsymbol{\theta})\rangle$ and the bottom two $|g\rangle$ states can be separately and adaptively tracked.

with $b_i \in \{0, 1\}$. To classically simulate this circuit, we utilize the fact that, if the controlled rotation gate is immediately followed by a measurement on a control qubit, this process is replaced with the following feedforward type operation; that is, the control qubit is first measured, and, if the measurement result is "1" the rotating operation acts on the target qubit. Repeating this replacement one by one from right to left in the circuit shown in Fig. 7(a), we end up Fig. 7(b). That is, all controlled-rotation gates are replaced with the one-qubit rotation gate which depends on the antecedent measurement results.

Using this classically controlled implementation, the QFT sampler can be simulated on a classical computer as described below. In the example considered here, the input $|\mathrm{in}\rangle$ is divided into at least three unentangled states $\{|\psi(\boldsymbol{\theta})\rangle, |g\rangle, |g\rangle\}$. Hence, these three parts can be tracked separately and adaptively, as shown in Fig. 7(c). In general, the change of states of the QFT sampler can be computed by repeatedly multiplicating 2^M-dimensional matrices on $|\psi(\boldsymbol{\theta})\rangle$ and two-dimensional matrices on $|g\rangle$ in the adaptive manner, leading that the total computational cost is of the order $\mathcal{O}(2^M + N)$. Therefore the QFT sampler can be classically simulated as long as 2^M is not too large.

[1] W. K. Hastings, Monte Carlo sampling methods using Markov chains and their applications, Biometrika **57**, 97 (1970).
[2] J. Liu, Y. Qi, Z. Yang Meng, and L. Fu, Self-learning Monte Carlo method, Phys. Rev. B **95**, 041101(R) (2017).
[3] H. Li and L. Wang, Accelerated Monte Carlo simulations with restricted Boltzmann machines, Phys. Rev. B **95**, 035105 (2017).
[4] M. S. Albergo, G. Kanwar, and P. E. Shanahan, Flow-based generative models for Markov chain Monte Carlo in lattice field theory, Phys. Rev. D **100**, 034515 (2019).
[5] H. Shen, J. Liu, and L. Fu, Self-learning Monte Carlo with deep neural networks, Phys. Rev. B **97**, 205140 (2018).
[6] S. Aaronson and A. Arkhipov, The computational complexity of linear optics, Theory of Computing **9**, 143 (2013).

[7] D. Shepherd and M. J Bremner, Temporally unstructured quantum computation, Proc. R. Soc. A **465**, 1413 (2009).

[8] M. J Bremner, R. Joza, and D. Shepherd, Classical simulation of commuting quantum computations implies collapse of the polynomial hierarchy, Proc. R. Soc. A **467**, 459 (2011).

[9] J. B. Spring *et al.*, Boson sampling on a photonic chip, Science **339**, 798 (2013).

[10] M. Tillmann *et al.*, Experimental boson sampling, Nat. Photonics **7**, 540 (2013).

[11] B. Coyle, D. Mills, V. Danos, and E. Kashefi, The Born supremacy: Quantum advantage and training of an Ising Born machine, npj Quantum Inf. **6**, 60 (2020).

[12] G. Brassard, P. Hoyer, M. Mosca, and A. Tapp, Quantum amplitude amplification and estimation, AMS Contemp. Math. Ser. **305**, 53 (2002).

[13] Y. Suzuki, S. Uno, R. Raymond, T. Tanaka, T. Onodera, and N. Yamamoto, Amplitude estimation without phase estimation, Quantum Info. Process. **19**, 75 (2020).

[14] J.-G. Liu and L. Wang, Differentiable learning of quantum circuit Born machines, Phys. Rev. A **98**, 062324 (2018).

[15] Z.-Yu. Han, J. Wang, H. Fan, L. Wang, and P. Zhang, Unsupervised Generative Modeling Using Matrix Product States, Phys. Rev. X **8**, 031012 (2018).

[16] D. E. Browne, Efficient classical simulation of the quantum Fourier transform, New J. Phys. **9**, 146 (2007).

[17] E. Tang, A quantum-inspired classical algorithm for recommendation systems, *Proceedings of the 51st Annual ACM SIGACT Symposium on Theory of Computing* (ACM Press, New York, 2019), pp. 217–228.

[18] K. Mitarai, M. Negoro, M. Kitagawa, and K. Fujii, Quantum circuit learning, Phys. Rev. A **98**, 032309 (2018).

[19] V. Sitzmann, J. N. P. Martel, A. W. Bergman, D. B. Lindell, and G. Wetzstein, Implicit neural representations with periodic activation functions, arXiv:2006.09661.

[20] M. Uteuliyeva, A. Zhumekenov, R. Takhanov, Z. Assylbekov, A. J. Castro, and O. Kabdolov, Fourier neural networks: A comparative study, Intelligent Data Analysis **24**, 1107 (2020).

[21] J. M. Sopena, E. Romero, and R. Alquezar, Neural networks with periodic and monotonic activation functions: a comparative study in classification problems, *Proceedings of Ninth International Conference on Artificial Neural Networks* (IET, 1999), pp. 323–328.

[22] R. Koplon and E. D. Sontag, Using Fourier-neural recurrent networks to fit sequential input/output data, Neurocomputing **15**, 225 (1997).

[23] A. Kandala, A. Mezzacapo, K. Temme, M. Takita, M. Brink, J. M. Chow, and J. M. Gambetta, Hardware-efficient variational quantum eigensolver for small molecules and quantum magnets, Nature (London) **549**, 242 (2017).

[24] Y. Wang, Y. Li, Z. Yin, and B. Zeng, 16-qubit IBM universal quantum computer can be fully entangled, npj Quantum Inf. **4**, 46 (2018).

[25] N. Qian, On the momentum term in gradient descent learning algorithms, Neural Networks **12**, 145 (1999).

[26] M. A. Nielsen and I. L. Chuang, *Quantum Computation and Quantum Information* (Cambridge University Press, Cambridge, 2000).

[27] https://github.com/IntenF/QFTSampler.

[28] K. Temme, T. J. Osborne, K. G. Vollbrecht, D. Poulin, and F. Verstraete, Quantum Metropolis sampling, Nature (London) **471**, 87 (2011).

[29] J. E. Moussa, Measurement-based Quantum Metropolis algorithm, arXiv:1903.01451.

[30] C. Gidney and M. Ekera, How to factor 2048 bit RSA integers in 8 hours using 20 million noisy qubits, arXiv:1905.09749.