Bayesian Analysis (2025) **TBA**, Number TBA, pp. 1–28

Quantum Speedups for Multiproposal MCMC

Chin-Yi Lin*, Kuo-Chin Chen†, Philippe Lemey§, Marc A. Suchard¶, Andrew J. Holbrook‖, and Min-Hsiu Hsieh‡

Abstract. Multiproposal Markov chain Monte Carlo (MCMC) algorithms choose from multiple proposals to generate their next chain step in order to sample from challenging target distributions more efficiently. However, on classical machines, these algorithms require $\mathcal{O}(P)$ target evaluations for each Markov chain step when choosing from P proposals. Recent work demonstrates the possibility of quadratic quantum speedups for one such multiproposal MCMC algorithm. After generating P proposals, this quantum parallel MCMC (`QPMCMC`) algorithm requires only $\mathcal{O}(\sqrt{P})$ target evaluations at each step, outperforming its classical counterpart. However, generating P proposals using classical computers still requires $\mathcal{O}(P)$ time complexity, resulting in the overall complexity of `QPMCMC` remaining $\mathcal{O}(P)$. Here, we present a new, faster quantum multiproposal MCMC strategy, `QPMCMC2`. With a specially designed Tjelmeland distribution that generates proposals close to the input state, `QPMCMC2` requires only $\mathcal{O}(1)$ target evaluations and $\mathcal{O}(\log P)$ qubits when computing over a large number of proposals P. Unlike its slower predecessor, the `QPMCMC2` Markov kernel (1) maintains detailed balance exactly and (2) is fully explicit for a large class of graphical models. We demonstrate this flexibility by applying `QPMCMC2` to novel Ising-type models built on bacterial evolutionary networks and obtain significant speedups for Bayesian ancestral trait reconstruction for 248 observed salmonella bacteria.

Keywords: Bayesian phylogenetics, MCMC, quantum algorithms, Ising models.

1 Introduction

In their many forms, multiproposal MCMC methods (Tjelmeland, 2004; Frenkel, 2004; Delmas and Jourdain, 2009; Neal, 2011; Calderhead, 2014; Luo and Tjelmeland, 2019) use multiple proposals to gain advantage over traditional MCMC algorithms (Metropolis et al., 1953; Hastings, 1970) that only generate a single proposal at each step. After generating a number of proposals, these methods randomly select the next Markov chain state from a set containing all P proposals and the current state with selection probabilities involving the target and proposal density (mass) functions. However, this claimed advantage has one shortcoming: calculation of proposal probabilities

arXiv: 2312.01402

*Department of Physics, National Taiwan University, Taipei, Taiwan, cylin1113@gmail.com
†Foxconn Research, Taipei, Taiwan, Jim.KC.Chen@foxconn.com
‡Foxconn Research, Taipei, Taiwan, min-hsiu.hsieh@foxconn.com
§Department of Microbiology, Immunology and Transplantation, Rega Institute, KU Leuven, Leuven, Belgium, philippe.lemey@kuleuven.be
¶Departments of Biostatistics, Biomathematics and Human Genetics, UCLA, Los Angeles, USA, msuchard@ucla.edu
‖Department of Biostatistics, UCLA, Los Angeles, USA, aholbroo@g.ucla.edu

© 2025 International Society for Bayesian Analysis https://doi.org/10.1214/25-BA1546

typically scales $\mathcal{O}(P^2)$ which outweighs the aforementioned advantages, ultimately resulting in degraded performance. Recent efforts (Glatt-Holtz et al., 2024a; Holbrook, 2023a) focus on efficient joint proposal structures that lead to computationally efficient $\mathcal{O}(P)$-time proposal selection probabilities. Even after incorporating efficient joint proposals such as the Tjelmeland correction (Section 2.2), selection probabilities still require evaluation of the function at each of the P proposals. Holbrook (2023b) uses the Gumbel-max trick to turn the proposal selection task into a discrete optimization procedure amenable to established quantum optimization techniques (Durr and Hoyer, 1996; Yoder et al., 2014). On the one hand, the resulting `QPMCMC` algorithm facilitates quadratic speedups, only requiring $\mathcal{O}(\sqrt{P})$ target evaluations. Although these quadratic speedups are significant, they are still not sufficient for `QPMCMC` to provide advantages when increasing the proposal number P. On the other hand, these target evaluations take the form of generic oracle calls embedded within successive Grover iterations (Grover, 1996), the circuit depth of which is not clear. Worse still, the fact that the optimization algorithms of Durr and Hoyer (1996); Yoder et al. (2014) sometimes fail to obtain the optimum means that the `QPMCMC` Markov kernel fails to maintain detailed balance with non-negligible probability. The relationship between the algorithm's stationary distribution (if it exists) and the target distribution is unclear as a result.

Our `QPMCMC2` algorithm (Section 3.2) combines multiproposal MCMC with quantum computing but improves upon `QPMCMC` in multiple ways. First, the `QPMCMC2` circuit depth is $\mathcal{O}(1)$, i.e., it does not grow with the number of proposals P. Second, the `QPMCMC2` Markov kernel maintains detailed balance exactly, so the algorithm obtains ergodicity and provides asymptotically exact estimators with the usual guarantees (Tierney, 1994). Third, the `QPMCMC2` circuit is fully explicit for a large class of graphical models, making it possible to quantify circuit depth and the $\mathcal{O}(\log P)$ circuit width. Our algorithm uses the same efficient multiproposal structures as `QPMCMC` to simplify selection probabilities, but this is where similarities cease. Instead of indirectly choosing the next Markov chain state via quantum optimization, we directly obtain selection probabilities as quantum probability amplitudes that provide weights for superposed proposal states. Collapsing the quantum state results in easy proposal selection.

Beyond `QPMCMC`, other quantum-accelerated MCMC algorithms, such as quantum simulated annealing (QSA) (Somma et al., 2008) and the quantum Metropolis solver (QMS) (Montanaro, 2015; Campos et al., 2023), have been proposed. While primarily designed for optimization tasks, these algorithms can also perform sampling. Theoretically, they achieve significant speedups during the convergence process by leveraging quantum phase estimation (QPE) (Dorner et al., 2009) and Szegedy's quantum walk (Szegedy, 2004). However, these methods execute all iterations within a single quantum circuit before measurement, drawing only one sample from the target distribution. This approach has two notable limitations: (1) the reliance on very deep quantum circuits, which are susceptible to substantial errors, and (2) the inability to retain the samples generated across iterations as classical data.

Although making a direct comparison between `QPMCMC2` and quantum MCMC algorithms like QSA or QMS is challenging, `QPMCMC2` offers distinct advantages. Unlike QSA

and QMS, which rely on a single quantum circuit to process all iterations, `QPMCMC2` employs a separate quantum circuit for each iteration. This significantly reduces the circuit depth and, consequently, the error rates, making `QPMCMC2` more practical for implementation on noisy quantum devices in the near term. Furthermore, the iterative structure of `QPMCMC2` enables the storage of all samples classically after each iteration, offering greater flexibility and usability compared to the designs of QSA and QMS, where only a single sample is generated at the end of the process.

We apply `QPMCMC2` to ancestral trait reconstruction on bacterial evolutionary networks, the irregularity of which serves as a naturally arising test of the algorithm's flexibility. Phylogenetic comparative methods (Felsenstein, 1985) investigate the shared evolution of biological traits and their mutual associations within or across species. Recent statistical efforts in comparative phylogenetics emphasize big data scalability and the application of increasingly complex models that condition on—or jointly infer—phylogenetic trees describing shared evolutionary histories between observed taxa (Hassler et al., 2023). For example, Zhang et al. (2021, 2023) develop a statistical computing framework for learning dependencies between high-dimensional discrete traits and apply their methods to the Bayesian analysis of, e.g., nearly 1,000 H1N1 influenza viruses. Unfortunately, these methods are ill-suited for bacterial ancestral trait reconstruction. First, their dynamic programming routines for fast likelihood and log-likelihood gradient calculations rely on the tree structure that directly characterizes the shared evolutionary history of the observed specimens, and the phylogenetic tree fails to capture the reticulate evolution that arises from the exchange of genetic material between microbes. Second, the methods of Zhang et al. (2021, 2023) rely on Gaussianity assumptions in order to efficiently integrate over unobserved ancestral traits and obtain a reduced likelihood describing only the traits of observed specimens.

Given these shortcomings, we instead define novel Ising-type models on Neighbor-Net phylogenetic networks (Bryant and Moulton, 2004) that directly account for bacterial reticulate evolution. Within these models, exterior nodes represent observed bacteria, internal nodes represent unobserved ancestors, and spins, the discrete binary variables associated with each node, represent biological traits. Bayesian ancestral trait reconstruction then amounts to sampling interior spins while keeping exterior spins fixed. We apply our `QPMCMC2` to this sampling task for single- and multi-trait Ising models that arise from a Neighbor-Net network characterizing the evolutionary history shared by 248 salmonella bacteria. Notably, this same microbial collection features prominently in high-impact studies (Mather et al., 2013; Cybis et al., 2015) of the evolution and development of antibiotic resistances in salmonella bacteria, a matter of pressing societal concern.

2 Preliminaries

We present limited introductions to the methods and ideas that are central to our development and exposition of `QPMCMC2`, including MCMC and multiproposal MCMC. See the Supplement (Lin et al., 2025) for a brief introduction to quantum computing.

2.1 MCMC and Barker's Algorithm

Markov Chain Monte Carlo (MCMC) constitutes a class of algorithms that are useful for sampling from probability distributions in situations where direct sampling is otherwise untenable. Key applications of MCMC include inference of high-dimensional model parameters within Bayesian inference (Gelman et al., 1995) and simulation of physical many-body systems (Metropolis et al., 1953; Duane et al., 1987). In the following, we consider the application of MCMC to discrete-valued models, but the framework applies equally to both discrete and continuous contexts. Letting $\mathcal{A}$ denote some finite or countably-infinite index set, we consider the discrete set $\{\boldsymbol{\theta}_\alpha\}_{\alpha\in\mathcal{A}}$. We identify our target distribution π with a probability mass function $\pi(\cdot)$ defined with respect to the counting measure on the power set $2^{\mathcal{A}}$. The probability measure π may be, e.g., a posterior distribution in Bayesian inference or a Boltzmann distribution in statistical mechanics. However, the probability mass function $\pi(\cdot)$ cannot be accessed in most practical scenarios. Instead, an unnormalized function $\pi^*(\cdot) \propto \pi(\cdot)$ is accessible.

In this context, Monte Carlo methods generate (pseudo) random samples in order to obtain estimates of expectations $E_\pi(f) < \infty$ for arbitrary bounded functions f defined on the set $\{\boldsymbol{\theta}_\alpha\}_{\alpha\in\mathcal{A}}$. Whereas classical Monte Carlo techniques such as rejection sampling tend to break down in high dimensions, MCMC effectively generates samples from high-dimensional distributions by constructing a Markov chain with transition kernel $Q(\cdot,\cdot)$ that maintains the target distribution π as a stationary distribution, i.e.,

$$\pi(\alpha) = \sum_{\alpha'} \pi(\alpha')Q(\alpha',\alpha), \quad \forall\alpha \in \mathcal{A}\,. \tag{1}$$

When designing such Markov kernels Q, it is helpful to note that the detailed balance condition

$$\pi(\alpha')Q(\alpha',\alpha) = \pi(\alpha)Q(\alpha,\alpha'), \quad \forall\alpha,\alpha' \in \mathcal{A} \tag{2}$$

guarantees the kernel Q's satisfaction of (1), while at the same time verifying more easily than (1). The Metropolis-Hastings kernel (Metropolis et al., 1953) maintains detailed balance using two steps: first, it generates a random proposal $\boldsymbol{\theta}_1 \sim q(\boldsymbol{\theta}_0,\boldsymbol{\theta}_1)$, where $\boldsymbol{\theta}_0 := \boldsymbol{\theta}^{(s-1)}$ is the current state of the Markov chain; second, it accepts the proposal with probability $a_{MH}(\boldsymbol{\theta}_0,\boldsymbol{\theta}_1)$ or remains in the current state for one more iteration.

In fact, other acceptance probabilities besides a_{MH} also maintain detailed balance when coupled with proposals of the form $q(\boldsymbol{\theta}_0,\boldsymbol{\theta}_1)$. We are particularly interested in the Barker (Barker, 1965) acceptance probability

$$a_B := \frac{\pi(\boldsymbol{\theta}_p)q(\boldsymbol{\theta}_p,\boldsymbol{\theta}_{|p-1|})}{\sum_{p'=0}^{1}\pi(\boldsymbol{\theta}_{p'})q(\boldsymbol{\theta}_{p'},\boldsymbol{\theta}_{|p'-1|})}, \quad p\in\{0,1\}. \tag{3}$$

When $q(\cdot,\cdot)$ is symmetric in its two arguments, (3) takes the salient form

$$\frac{\pi(\boldsymbol{\theta}_p)}{\sum_{p'=0}^{1}\pi(\boldsymbol{\theta}_{p'})} =: \pi_p, \quad p\in\{0,1\}, \tag{4}$$

Algorithm 1 MCMC with Barker Acceptances and Symmetric Proposals

Input: An initial Markov chain state $\boldsymbol{\theta}^{(0)}$; a routine for evaluating a function $\pi^*(\cdot) \propto \pi(\cdot)$, where $\pi(\cdot)$ is our target distribution's probability mass function; a routine for sampling $\boldsymbol{\theta}'$ from a proposal distribution $q(\boldsymbol{\theta}, \boldsymbol{\theta}')$ symmetric in $\boldsymbol{\theta}$ and $\boldsymbol{\theta}'$; a routine for sampling from a discrete distribution Discrete$(\cdot)$ parameterized by an arbitrary probability vector; the number of samples to generate S.

1: **for** $s \in \{1, \ldots, S\}$ **do**
2: $\quad \boldsymbol{\theta_0} \leftarrow \boldsymbol{\theta}^{(s-1)}$; $\boldsymbol{\theta}_1 \sim q(\boldsymbol{\theta}_0, \cdot)$;
3: $\quad \boldsymbol{\pi^*} = (\pi_0^*, \pi_1^*)^T$ where $\pi_0^* \leftarrow \pi^*(\boldsymbol{\theta_0})$ and $\pi_1^* \leftarrow \pi^*(\boldsymbol{\theta_1})$;
4: $\quad \hat{p} \sim$ Discrete$(\boldsymbol{\pi}^*/\boldsymbol{\pi}^{*T}\mathbf{1})$; $\boldsymbol{\theta^{(s)}} \leftarrow \boldsymbol{\theta}_{\hat{\boldsymbol{p}}}$;
5: **end for**
6: **return** $\boldsymbol{\theta^{(1)}}, \ldots, \boldsymbol{\theta^{(S)}}$.

leading to Algorithm 1. The notation of (3) and (4) extends to the multiple proposal case. Here, the development of symmetric joint proposals and simplified acceptances π_p is not straightforward, but leads to significant computational efficiencies. It is worth noting that the simplified acceptances π_p can be obtained by replacing π with π^*:

$$\frac{\pi^*(\boldsymbol{\theta}_p)}{\sum_{p'=0}^{1} \pi^*(\boldsymbol{\theta}_{p'})} = \frac{\pi(\boldsymbol{\theta}_p)}{\sum_{p'=0}^{1} \pi(\boldsymbol{\theta}_{p'})} = \pi_p, \quad p \in \{0, 1\}. \tag{5}$$

2.2 Multiproposal MCMC and the Tjelmeland Correction

Multiproposal MCMC algorithms use multiple proposals at each iteration to explore target distributions more efficiently. Recently, Glatt-Holtz et al. (2024a) present general measure theoretic foundations for the many different multiproposal MCMC algorithms that already exist. Among many other important contributions, this abstract multiproposal MCMC framework incorporates: both Metropolis-Hastings-type and Barker-type multiproposal MCMC acceptance criteria; and efficient joint proposal structures (Tjelmeland, 2004; Holbrook, 2023a) called Tjelmeland corrections. We follow Holbrook (2023a,b) and consider a multiproposal MCMC algorithm that combines Barker-type acceptance criteria with the Tjelmeland correction.

Again letting $\boldsymbol{\theta}_0 := \boldsymbol{\theta}^{(s-1)}$ denote the current state of the Markov chain, one version of multiproposal MCMC proceeds by generating P proposals $(\boldsymbol{\theta}_1, \ldots, \boldsymbol{\theta}_P) =: \boldsymbol{\Theta}_{-0}$ from some joint distribution with probability mass function $q(\boldsymbol{\theta}_0, \boldsymbol{\Theta}_{-0})$ and randomly selecting the next Markov chain state from among the current and proposed states with probabilities

$$\pi_p := \frac{\pi(\boldsymbol{\theta}_p) q(\boldsymbol{\theta}_p, \boldsymbol{\Theta}_{-p})}{\sum_{p'=0}^{P} \pi(\boldsymbol{\theta}_{p'}) q(\boldsymbol{\theta}_{p'}, \boldsymbol{\Theta}_{-p'})} = \frac{\pi^*(\boldsymbol{\theta}_p) q(\boldsymbol{\theta}_p, \boldsymbol{\Theta}_{-p})}{\sum_{p'=0}^{P} \pi^*(\boldsymbol{\theta}_{p'}) q(\boldsymbol{\theta}_{p'}, \boldsymbol{\Theta}_{-p'})}, \quad p \in \{0, \ldots, P\}, \tag{6}$$

where $\boldsymbol{\Theta}_{-p}$ is the P-columned matrix that results when one extracts the vector $\boldsymbol{\theta}_p$ from the matrix $(\boldsymbol{\theta}_0, \boldsymbol{\theta}_1, \ldots, \boldsymbol{\theta}_P)$. Given the burdensome $\mathcal{O}(P^2)$ floating-point operations required to evaluate all $P + 1$ joint mass functions $q(\boldsymbol{\theta}_p, \boldsymbol{\Theta}_{-p})$, Holbrook (2023a)

Algorithm 2 Multiproposal MCMC with Barker Acceptances and the Tjelmeland Correction

Input: An initial Markov chain state $\boldsymbol{\theta}^{(0)}$; a routine for evaluating a function $\pi^*(\cdot) \propto \pi(\cdot)$, where $\pi(\cdot)$ is our target distribution's probability mass function; a routine for sampling $\boldsymbol{\theta}'$ from a Tjelmeland distribution $\bar{q}(\boldsymbol{\theta}, \boldsymbol{\theta}')$ symmetric in $\boldsymbol{\theta}$ and $\boldsymbol{\theta}'$; a routine for sampling from a discrete distribution Discrete$(\cdot)$ parameterized by an arbitrary probability vector; the number of samples to generate S; the number of proposals P.

1: **for** $s \in \{1, \ldots, S\}$ **do**
2: $\quad \boldsymbol{\theta_0} \leftarrow \boldsymbol{\theta}^{(s-1)}$;
3: $\quad \boldsymbol{\theta}^{(s)} \leftarrow$ Multiproposal MCMC iteration (Algorithm 3)
4: **end for**
5: **return** $\boldsymbol{\theta}^{(1)}, \ldots, \boldsymbol{\theta}^{(S)}$.

recommends using joint proposal strategies that enforce the higher-order symmetry relation

$$q(\boldsymbol{\theta}_0, \boldsymbol{\Theta}_{-0}) = q(\boldsymbol{\theta}_1, \boldsymbol{\Theta}_{-1}) = \cdots = q(\boldsymbol{\theta}_P, \boldsymbol{\Theta}_{-P}) \tag{7}$$

and lead to simplified acceptance probabilities

$$\pi_p = \frac{\pi^*(\boldsymbol{\theta}_p)}{\sum_{p'=0}^{P} \pi^*(\boldsymbol{\theta}_{p'})}, \quad p \in \{0, 1, \ldots, P\}. \tag{8}$$

To this end, Holbrook (2023a) shows that an elegant joint proposal structure put forth by Tjelmeland (2004) leads to (7). This Tjelmeland correction uses a symmetric probability distribution with mass function satisfying $\bar{q}(\boldsymbol{\theta}, \boldsymbol{\theta}') = \bar{q}(\boldsymbol{\theta}', \boldsymbol{\theta})$ to first generate a random offset $\bar{\boldsymbol{\theta}} \sim \bar{q}(\boldsymbol{\theta}_0, \cdot)$ and then generate P proposals $\boldsymbol{\theta}_1, \ldots, \boldsymbol{\theta}_P \overset{iid}{\sim} \bar{q}(\bar{\boldsymbol{\theta}}, \cdot)$. Because

$$q(\boldsymbol{\theta}_0, \boldsymbol{\Theta}_{-0}) = \sum_{\bar{\boldsymbol{\theta}}} \bar{q}(\boldsymbol{\theta}_0, \bar{\boldsymbol{\theta}}) \prod_{p' \neq 0} \bar{q}(\bar{\boldsymbol{\theta}}, \boldsymbol{\theta}_{p'}) = \sum_{\bar{\boldsymbol{\theta}}} \bar{q}(\boldsymbol{\theta}_p, \bar{\boldsymbol{\theta}}) \prod_{p' \neq p} \bar{q}(\bar{\boldsymbol{\theta}}, \boldsymbol{\theta}_{p'}) = q(\boldsymbol{\theta}_p, \boldsymbol{\Theta}_{-p}),$$

this joint proposal strategy satisfies (7) and leads to the simple multiproposal MCMC routine shown in Algorithm 3. In the following, we refer to $\bar{q}(\cdot, \cdot)$ as a Tjelmeland kernel and $\bar{q}(\boldsymbol{\theta}, \cdot)$ as a Tjelmeland distribution. According to Theorem 2.11 and Corollary 2.12 in Glatt-Holtz et al. (2024b), Algorithm 2 is guaranteed to maintain detail balance and leaves the target distribution $\pi^*(\cdot)$ invariant.

3 Main Result: Fast Quantum Parallel MCMC

In this section, we introduce our main result for `QPMCMC2` (Algorithm 4), which presents a novel quantum sampling algorithm that surpasses the classical multiproposal MCMC (Algorithm 2). Subsequently, we delineate the technical contribution, and illustrate its significance in addressing the bottleneck identified in Algorithm 2. Finally, a comprehensive elucidation of our algorithm will be presented in detail.

Algorithm 3 Multiproposal MCMC iteration with Barker Acceptances and the Tjelmeland Correction

Input: An input Markov chain state $\boldsymbol{\theta}_0$; a routine for evaluating a function $\pi^*(\cdot) \propto \pi(\cdot)$, where $\pi(\cdot)$ is our target distribution's probability mass function; a routine for sampling $\boldsymbol{\theta}'$ from a Tjelmeland distribution $\bar{q}(\boldsymbol{\theta}, \boldsymbol{\theta}')$ symmetric in $\boldsymbol{\theta}$ and $\boldsymbol{\theta}'$; a routine for sampling from a discrete distribution Discrete$(\cdot)$ parameterized by an arbitrary probability vector; the number of proposals P.

1: $\bar{\boldsymbol{\theta}} \sim \bar{q}(\boldsymbol{\theta}_0, \cdot)$; $\boldsymbol{\theta}_1, \ldots, \boldsymbol{\theta}_P \overset{iid}{\sim} \bar{q}(\bar{\boldsymbol{\theta}}, \cdot)$;
2: $\boldsymbol{\pi^*} = (\pi_0^*, \pi_1^*, \ldots, \pi_P^*)^T$ where $\pi_p^* \leftarrow \pi^*(\boldsymbol{\theta_p})$, $p \in \{0, 1, \ldots, P\}$;
3: $\hat{p} \sim$ Discrete$(\boldsymbol{\pi^*}/\boldsymbol{\pi^*}^T\mathbf{1})$;
4: **return** $\boldsymbol{\theta}_{\hat{\boldsymbol{p}}}$.

`QPMCMC2` (Algorithm 4) is a quantum sampling algorithm that serves as the quantum counterpart to the classical multiproposal MCMC (Algorithm 2), offering improved time complexity under specific conditions. This acceleration is achieved by substituting the classical iteration step (Algorithm 3) with its quantum-enhanced version (Algorithm 5).

Theorem 3.1. *Algorithm 5 is a quantum algorithm that is equivalent to Algorithm 3, with the additional input $\mathcal{L}$ that is a constant larger than $\max_{\boldsymbol{\theta} \sim \bar{q}(\boldsymbol{\theta}', \cdot); \boldsymbol{\theta}' \in \mathcal{A}} \left[\frac{\pi(\boldsymbol{\theta})}{\pi(\boldsymbol{\theta}')}\right]$. This quantum algorithm has a success probability given by:*

$$R = \sum_{p \in \{0, \ldots, P\}} \frac{\pi^*_{\bar{\boldsymbol{\theta}}}(\boldsymbol{\theta}_p)}{P+1},$$

where $\bar{\boldsymbol{\theta}} \sim \bar{q}(\boldsymbol{\theta}_0, \cdot)$ is the intermediate state generated from input state $\boldsymbol{\theta}_0$, and $\pi^*_{\bar{\boldsymbol{\theta}}} := \frac{\pi}{\pi(\bar{\boldsymbol{\theta}})\mathcal{L}}$ is an unnormalized probability mass function corresponding to π. The time complexity of this quantum algorithm is expressed as:

$$2\mathbb{T}(O_{\bar{q}}) + \mathbb{T}(O_{\pi^*_{\bar{\boldsymbol{\theta}}}}) + \mathcal{O}(1), \tag{9}$$

where $O_{\bar{q}}$ and $O_{\pi^*_{\bar{\boldsymbol{\theta}}}}$ represent the quantum operations corresponding to $\bar{q}(\boldsymbol{\theta}, \boldsymbol{\theta}')$ and $\pi^*_{\bar{\boldsymbol{\theta}}}(\cdot)$, respectively.[1] Furthermore, the success probability R can be lower bounded by a quantity that only depends on $\bar{q}$ and π as follow:

$$R \geq \frac{\mathcal{M}}{\mathcal{L}}, \tag{10}$$

where $\mathcal{M} = \min_{\boldsymbol{\theta} \sim \bar{q}(\boldsymbol{\theta}', \cdot); \boldsymbol{\theta}' \in \mathcal{A}} \left[\frac{\pi(\boldsymbol{\theta})}{\pi(\boldsymbol{\theta}')}\right]$.

Algorithm 5 is the sub-algorithm used in each iteration of Algorithm 4, representing a quantum accelerated version of Algorithm 3, which is the sub-algorithm of a single iteration of the classical multiproposal MCMC (Algorithm 2). We note that Algorithm 3 requires evaluating π^* and $\bar{q}$ a total of P times, leading to $\mathcal{O}(P)$ implementations of

[1] See Supplement for a detailed description.

these functions. In contrast, Algorithm 5 requires only $\mathcal{O}(1)$ implementations, making it independent of the choice of P. Although Algorithm 5 may fail with a probability $1 - R$, it is lower bounded in Equation (10) which is independent of P. With this quantum-enhanced MCMC iteration (Algorithm 5), the time complexity of `QPMCMC2` (Algorithm 4) remains independent of P.

Notice that in Algorithm 5, the lower bound of the success rate R given in Equation (10) can generally be small while remaining independent of P. Here, we demonstrate that for a target distribution $\pi(\cdot)$, if $\log \pi(\cdot)$ satisfies the Lipschitz continuity property (Definition 3.2), it is possible to calculate higher lower bounds for R for certain specifically designed Tjelmeland distributions $\bar{q}(\cdot, \cdot)$.

Definition 3.2. Let $K \in \mathbb{R}^+$ be a positive constant, and let $(\mathcal{X}, d_{\mathcal{X}})$ and $(\mathcal{Y}, d_{\mathcal{Y}})$ be two metric spaces, where $d_{\mathcal{X}}$ and $d_{\mathcal{Y}}$ are the metrics on the sets $\mathcal{X}$ and $\mathcal{Y}$, respectively. A function $f : \mathcal{X} \to \mathcal{Y}$ is said to be K-Lipschitz continuous if:

$$d_{\mathcal{Y}}(f(x_1), f(x_2)) \leq K\, d_{\mathcal{X}}(x_1, x_2), \quad \forall x_1, x_2 \in \mathcal{X}.$$

For a target distribution $\pi : \mathcal{A} \to [0, 1]$ such that $\log \pi(\boldsymbol{\theta})$ is K-Lipschitz, the following property holds:

$$\left| \log \left(\frac{\pi(\boldsymbol{\theta}_1)}{\pi(\boldsymbol{\theta}_2)} \right) \right| \leq K d_{\mathcal{A}}(\boldsymbol{\theta}_1, \boldsymbol{\theta}_2), \tag{11}$$

where $\boldsymbol{\theta}_1, \boldsymbol{\theta}_2 \in \mathcal{A}$.

To optimize the lower bound in Equation (10), we design the Tjelmeland distribution $\bar{q}(\cdot, \cdot)$ such that it only assigns non-zero probabilities to pairs of states $\boldsymbol{\theta}_1, \boldsymbol{\theta}_2$ that are sufficiently close to each other, with a given threshold $\mathcal{D} \in \mathbb{R}^+$:

$$\bar{q}(\boldsymbol{\theta}_1, \boldsymbol{\theta}_2) = 0, \quad \forall d_{\mathcal{A}}(\boldsymbol{\theta}_1, \boldsymbol{\theta}_2) > \mathcal{D}. \tag{12}$$

By combining Equation (11) and Equation (12), for all $\boldsymbol{\theta}' \in \mathcal{A}$ and $\boldsymbol{\theta} \sim \bar{q}(\boldsymbol{\theta}', \cdot)$, the following inequality holds:

$$e^{-KD} \leq \frac{\pi(\boldsymbol{\theta})}{\pi(\boldsymbol{\theta}')} \leq e^{KD}. \tag{13}$$

Thus, by setting $\mathcal{L} = e^{KD}$ and substituting into Equation (10), we derive:

$$R \geq \frac{\min_{\boldsymbol{\theta} \sim \bar{q}(\boldsymbol{\theta}', \cdot); \boldsymbol{\theta}' \in \mathcal{A}} \left[\frac{\pi(\boldsymbol{\theta})}{\pi(\boldsymbol{\theta}')} \right]}{\mathcal{L}} \geq e^{-2KD}.$$

From this analysis, we observe that reducing $\mathcal{D}$ leads to a higher lower bound for R, thereby guaranteeing a higher success rate for Algorithm 5.

Unlike Algorithm 3, Algorithm 5 requires an additional input

$$\mathcal{L} \geq \max_{\boldsymbol{\theta} \sim \bar{q}(\boldsymbol{\theta}', \cdot); \boldsymbol{\theta}' \in \mathcal{A}} \left[\frac{\pi(\boldsymbol{\theta})}{\pi(\boldsymbol{\theta}')} \right].$$

From the above analysis, we have shown that when $\log(\pi(\cdot))$ satisfies Equation (11), a better choice of $\mathcal{L} = e^{KD}$ can be achieved. Additionally, in Section 4, we provide an example of how to design $\bar{q}$ to meet the requirement in Equation (12) in the case of Ising model sampling. However, it is important to note that there is no guarantee to find such a constant $\mathcal{L}$ for arbitrary distributions $\pi(\cdot)$ and the Tjelmeland distributions $\bar{q}(\cdot,\cdot)$. Algorithm 4 and Algorithm 5 is only feasible when $\mathcal{L}$ is provided.

Still, R could be very small in certain cases, leading to an enormous rerun of Algorithm 5 as the scaling of $\mathcal{O}(1/R)$. In fact, the success rate R in Theorem 3.1 can be enhanced to surpass $\frac{1}{2}$ with $\mathcal{O}(1/\sqrt{R})$ calls of $O_{\bar{q}}$ and $O_{\pi^*_{\tilde{\theta}}}$, using the quantum amplitude amplification algorithm presented by Brassard et al. (2000). This improvement leads to the following corollary:

Corollary 3.2.1. *By applying the quantum amplitude amplification algorithm from Brassard et al. (2000) to Algorithm 5, the time complexity for implementing one iteration in* `QPMCMC2` *(line 2-5, Algorithm 4) is expressed as:*

$$(2\mathbb{T}(O_{\bar{q}}) + \mathbb{T}(O_{\pi^*_{\tilde{\theta}}}) + \mathcal{O}(1)) \cdot \mathcal{O}\left(\frac{1}{\sqrt{R}}\right). \tag{14}$$

The quantum amplitude amplification algorithm introduces a quadratic speedup on the scaling of the success rate R, which is a suitable solution to small R. Detailed description of the quantum amplitude amplification algorithm is provided in Section 3.3.

3.1 Significance

The significance of `QPMCMC2` (Algorithm 4) is twofold. First, it demonstrates an improvement in time complexity over the classical multiproposal MCMC (Algorithm 2) and the earlier quantum parallel MCMC algorithm `QPMCMC` proposed in (Holbrook, 2023b). Therefore, `QPMCMC2` can be used to enhance the convergence rate by increasing the number of proposals P without requiring P evaluating π^* and $\bar{q}$. Second, it enhances sampling efficiency, particularly by increasing the effective sample size (ESS) (Gelman et al., 1995), leading to more reliable estimates with fewer samples.

Improvement in Time Complexity

Previous studies have demonstrated the advantages of multiproposal MCMC algorithms over, e.g., the Metropolis-Hastings algorithm. Specifically, an increase in the number of proposals P in Algorithm 3, which is an iteration of Algorithm 2, leads to expedited convergence in the sampling process.

However, this accelerated convergence speed comes with a certain drawback. Typically, the bottleneck in each iteration of the Markov chain, as outlined in Algorithm 2, lies in the computation of $\pi^*(\cdot)$. The heightened number of $\pi^*(\cdot)$ computations required by this multiproposal MCMC algorithm demands significant computational resources when augmenting the proposal number P per iteration. In Algorithm 2, achieving a single Markov chain iteration through classical computation necessitates a time complexity of $\mathcal{O}(P)$.

Conversely, a quantum circuit can execute parallel calculations across different proposals concurrently. This encompasses the generation of proposal sets, evaluation of $\pi^*_{\tilde{\boldsymbol{\theta}}}(\boldsymbol{\theta}_p)$ for each $\boldsymbol{\theta}_p$ among this proposal sets, encoding them as a superposition state with $P+1$ components, and the selection of samples among them. This quantum approach holds promise in mitigating the bottleneck of the multiproposal MCMC algorithm, thereby expediting the convergence process. By concurrently processing multiple proposals, this quantum multiproposal MCMC algorithm becomes more competitive in comparison to traditional algorithms that rely on a single proposal.

This work is not the first to utilize quantum circuits in an endeavor to expedite Algorithm 2. In a prior investigation (Holbrook, 2023b), the employment of the Grover search approach and the Gumbel-Max trick aimed to devise a quantum algorithm (`QPMCMC`) for substituting lines 3-4 in Algorithm 2, thereby enhancing the time complexity of these steps from $\mathcal{O}(P)$ to $\mathcal{O}(\sqrt{P})$. It is noteworthy that, in that study, the acceleration did not extend to the process of generating P proposal sets (line 2, Algorithm 2), maintaining the overall complexity for a `QPMCMC` iteration at $\mathcal{O}(P)$.

Upon comparing this work to the previously mentioned study, it becomes evident that our approach signifies a notable advancement over them. When proposal count P is large enough, with a designed Tjelmeland distribution $\bar{q}(\cdot,\cdot)$ that generates proposals closed enough to the input state, we achieve an exponential speedup in terms of the P, when contrasted with Algorithm 2.

Improvement in Effective Sample Size

With Theorem 3.1, we can improve another indicator of the sampling efficiency, which is the effective sample size (ESS) (Gelman et al., 1995):

$$\text{ESS} := \frac{S}{\sum_{s=-\infty}^{\infty} \rho(s)},$$

where S is the number of MCMC samples, and $\rho(s)$ is the autocorrelation of a univariate time series at lag s. An effective sampler generally exhibits lower autocorrelation for key model summary statistics, resulting in a larger ESS. The ESS provides a measure of how many independent samples the correlated chain is equivalent to: it gives you an idea of the true amount of information your sample contains, taking into account the correlation between sample points. With the time complexity reduction in our quantum algorithm, we are able to achieve a larger effective sample size per oracle, making a more efficient MCMC sampling algorithm.

3.2 Improved Quantum Parallel MCMC and Its Time Complexity

In this subsection, we first provide a detailed description of the quantum-accelerated multiproposal MCMC iteration (Algorithm 5) used in `QPMCMC2` (Algorithm 4). We then establish its correctness and analyze its time complexity. Similar to Algorithm 3, Algorithm 5 takes the following inputs: an initial Markov chain state $\boldsymbol{\theta}_0$, the number of proposals P, and the quantum oracles $O_{\bar{q}}$ and $O_{\pi^*_{\tilde{\boldsymbol{\theta}}}}$, which correspond to $\bar{q}(\boldsymbol{\theta},\boldsymbol{\theta}')$

Algorithm 4 Quantum accelerated multiproposal MCMC (`QPMCMC2`)

Input: An initial Markov chain state $\boldsymbol{\theta}^{(0)}$; an oracle $O_{\bar{q}}$ for sampling $\boldsymbol{\theta}'$ from a Tjelmeland distribution $\bar{q}(\boldsymbol{\theta}, \boldsymbol{\theta}')$ symmetric in $\boldsymbol{\theta}$ and $\boldsymbol{\theta}'$; a constant $\mathcal{L} \geq \max_{\boldsymbol{\theta} \sim \bar{q}(\boldsymbol{\theta}', \cdot); \boldsymbol{\theta}' \in \mathcal{A}} [\frac{\pi(\boldsymbol{\theta})}{\pi(\boldsymbol{\theta}')}]$; a control rotation operator $CR(\cdot)$; the number of samples to generate S; the number of proposals P.

1: **for** $s \in \{1, \ldots, S\}$ **do**
2: $\quad \boldsymbol{\theta_0} \leftarrow \boldsymbol{\theta}^{(s-1)}$; $\boldsymbol{\theta}^{(s)} \leftarrow$ **Stop**
3: $\quad$ **while** $\boldsymbol{\theta}^{(s)} ==$ **Stop do**
4: $\quad\quad \boldsymbol{\theta}^{(s)} \leftarrow$ Quantum accelerated multiproposal MCMC iteration (Algorithm 5)
5: $\quad$ **end while**
6: **end for**
7: **return** $\boldsymbol{\theta}^{(1)}, \ldots, \boldsymbol{\theta}^{(S)}$.

Algorithm 5 Quantum accelerated multiproposal MCMC iteration (An iteration of `QPMCMC2`)

Input: An input Markov chain state $\boldsymbol{\theta}_0$; an oracle $O_{\bar{q}}$ for sampling $\boldsymbol{\theta}'$ from a Tjelmeland distribution $\bar{q}(\boldsymbol{\theta}, \boldsymbol{\theta}')$ symmetric in $\boldsymbol{\theta}$ and $\boldsymbol{\theta}'$; an oracle $O_{\pi^*_{\boldsymbol{\theta}'}}$ for evaluating an unnormalized probability mass function $\pi^*_{\boldsymbol{\theta}'}(\cdot) = \frac{\pi(\cdot)}{\pi(\boldsymbol{\theta}')\mathcal{L}} \propto \pi(\cdot)$ with the given $\mathcal{L} \geq \max_{\boldsymbol{\theta} \sim \bar{q}(\boldsymbol{\theta}', \cdot); \boldsymbol{\theta}' \in \mathcal{A}} [\frac{\pi(\boldsymbol{\theta})}{\pi(\boldsymbol{\theta}')}]$; a control rotation operator $CR(\cdot)$; the number of proposals P.

1: Prepare a quantum state $|\psi_0\rangle = |0\rangle_{\mathcal{P}} |0\rangle_{\mathcal{H}_0} |0\rangle_{\mathcal{H}_1} |0\rangle_{\mathcal{H}_2} |0\rangle_{\Pi} |0\rangle_{S}$
2: Encode $\boldsymbol{\theta}_0$ in $\mathcal{H}_0$.
3: Apply $O_{\bar{q}}$, which takes query from $\mathcal{H}_0$ and responses the intermediate state $\bar{\boldsymbol{\theta}}$ in $\mathcal{H}_1$
4: Make a uniform superposition state in $\mathcal{P}$
5: Apply $O_{\bar{q}}$, which takes a query from $\mathcal{H}_1$, and responses in $\mathcal{H}_2$ on each state
6: Apply $O_{\pi^*_{\bar{\boldsymbol{\theta}}}}$ with , which takes a query from $\mathcal{H}_2$, and responses in Π on each state
7: Apply a control rotation gate CR (controlled by each $|p\rangle_{\mathcal{P}}$), which takes a query from Π and maps $|0\rangle_{\mathcal{S}}$ to $\sqrt{1 - \pi^*_{\bar{\boldsymbol{\theta}}}(\boldsymbol{\theta}_p)}\,|0\rangle_S + \sqrt{\pi^*_{\bar{\boldsymbol{\theta}}}(\boldsymbol{\theta}_p)}\,|1\rangle_S$
8: Make a measurement;
9: **if** $\mathcal{S}$ register is 0 **then**:
10: $\quad$ **return Stop**
11: **else**
12: $\quad \boldsymbol{\theta}^{(s)} \leftarrow$ the data in $\mathcal{H}_2$
13: $\quad$ **return** $\boldsymbol{\theta}^{(s)}$
14: **end if**

and $\pi^*_{\bar{\boldsymbol{\theta}}}$ in Algorithm 3, respectively. Additionally, Algorithm 5 requires an extra input $\mathcal{L} \geq \max_{\boldsymbol{\theta} \sim \bar{q}(\boldsymbol{\theta}', \cdot); \boldsymbol{\theta}' \in \mathcal{A}} [\frac{\pi(\boldsymbol{\theta})}{\pi(\boldsymbol{\theta}')}]$. Finally, it requires a controlled rotation operation CR. These quantum operations are introduced in Section 2.3.

The quantum algorithm begins by initializing several quantum registers according to the following scheme:

- The first register, denoted as $\mathcal{P}$, is encoded with the labels of proposals $\{0, \cdots, P\}$ as specified in Algorithm 2.
- The second register, labeled $\mathcal{H}_0$, is encoded with the input state $\boldsymbol{\theta}_0$.
- The third register, termed $\mathcal{H}_1$, is encoded with the random offset $\bar{\boldsymbol{\theta}}$ as described in Algorithm 2.
- The fourth register, denoted as $\mathcal{H}_2$, is encoded with the proposals $\boldsymbol{\theta}_p \overset{i.i.d.}{\sim} \bar{q}(\boldsymbol{\theta}, \boldsymbol{\theta}')$ for each label of proposal $p \in \{0, \cdots, P\}$.
- The fifth register, denoted as Π, is encoded with the evaluated value from the target distribution $\pi(\cdot)$ for each label of proposal p.
- The last register, designated as $\mathcal{S}$, is a register indicating whether the implementation of the Markov chain is successful or not.

The quantum Algorithm 5 commences by initializing these quantum registers to hold zero and subsequently executing five steps.

Initially, Algorithm 5 encodes the initial Markov chain state $\boldsymbol{\theta}_0$ into the register $\mathcal{H}_0$. This operation necessitates approximately $\mathcal{O}(\log(|\mathcal{A}|))$ controlled-NOT gate operations where $\mathcal{A}$ is the parameter space (introduced in Section 2.1).

Secondly, Algorithm 5 considers an operator $O_{\bar{q}}$ characterized by the Tjelmeland distribution $\bar{q}(\boldsymbol{\theta}_0, \cdot)$. This operation selects a state $\bar{\boldsymbol{\theta}}$ from the distribution $\bar{q}(\boldsymbol{\theta}_0, \cdot)$ and encodes this state into the register $\mathcal{H}_1$. The resulting state is represented as:

$$|0\rangle_{\mathcal{P}} |\boldsymbol{\theta}_0\rangle_{\mathcal{H}_0} |\bar{\boldsymbol{\theta}}\rangle_{\mathcal{H}_1} |0\rangle_{\mathcal{H}_2} |0\rangle_{\Pi} |0\rangle_{S},$$

where $\bar{\boldsymbol{\theta}} \sim \bar{q}(\boldsymbol{\theta}_0, \cdot)$. The time required for this step is $\mathbb{T}(O_{\bar{q}})$.

Thirdly, Algorithm 5 creates a uniformly distributed superposition in register $\mathcal{P}$ such that each state is entangled with the proposal states $\boldsymbol{\theta}_p$ encoded in register $\mathcal{H}_2$. This process can be achieved by employing approximately $\mathcal{O}(\log(P))$ rotation gate operations on register $\mathcal{P}$, followed by an operation $O_{\bar{q}}$ controlled by each $|p\rangle_{\mathcal{P}}$. The resultant state is given by:

$$\frac{1}{\sqrt{P+1}} \sum_{p=0}^{P} |p\rangle_{\mathcal{P}} |\boldsymbol{\theta}_0\rangle_{\mathcal{H}_0} |\bar{\boldsymbol{\theta}}\rangle_{\mathcal{H}_1} |\boldsymbol{\theta}_p\rangle_{\mathcal{H}_2} |0\rangle_{\Pi} |0\rangle_{S},$$

where $\boldsymbol{\theta}_1, \ldots, \boldsymbol{\theta}_P \overset{i.i.d.}{\sim} \bar{q}(\bar{\theta}, \cdot)$. Note that the time complexity for this operation is $\mathbb{T}(O_{\bar{q}}) + \mathcal{O}(1)$.

The fourth step involves encoding the evaluated value from the target distribution $\pi(\cdot)$ for each proposal label into the prefactor of each state. This task comprises two operations: the first is an oracle $O_{\pi^*_{\bar{\boldsymbol{\theta}}}}$ that accepts queries from $\mathcal{H}_2$ and responds with

the answer in the Π register. Subsequently, a controlled rotation operator CR receives a query from Π and rotates the qubit in $\mathcal{S}$. The resulting state is expressed as:

$$\begin{aligned}&\frac{1}{\sqrt{P+1}}\sum_{p=0}^{P}|p\rangle_{\mathcal{P}}\,|\boldsymbol{\theta}_0\rangle_{\mathcal{H}_0}\,\big|\bar{\boldsymbol{\theta}}\big\rangle_{\mathcal{H}_1}\,|\boldsymbol{\theta}_p\rangle_{\mathcal{H}_2}\,\big|\pi^*_{\bar{\boldsymbol{\theta}}}(\boldsymbol{\theta}_p)\big\rangle_{\Pi}\left[\sqrt{1-\pi^*_{\bar{\boldsymbol{\theta}}}(\boldsymbol{\theta}_p)}\,|0\rangle_{\mathcal{S}}+\sqrt{\pi^*_{\bar{\boldsymbol{\theta}}}(\boldsymbol{\theta}_p)}\,|1\rangle_{\mathcal{S}}\right]\\&=\sqrt{R}\,|\mathrm{SUCC}\rangle_{\mathcal{P},\mathcal{H}_0,\mathcal{H}_1,\mathcal{H}_2,\Pi}\,|1\rangle_{\mathcal{S}}+\sqrt{1-R}\,|\mathrm{FAIL}\rangle_{\mathcal{P},\mathcal{H}_0,\mathcal{H}_1,\mathcal{H}_2,\Pi}\,|0\rangle_{\mathcal{S}}\,,\end{aligned}\tag{15}$$

where we denote $R=\frac{\sum_{p'=0}^{P}\pi^*_{\bar{\boldsymbol{\theta}}}(\boldsymbol{\theta}_{p'})}{P+1}$ and set $|\mathrm{SUCC}\rangle$[2] as follows:

$$|\mathrm{SUCC}\rangle=\sum_{p=0}^{P}\sqrt{\frac{\pi^*_{\bar{\boldsymbol{\theta}}}(\boldsymbol{\theta}_p)}{\sum_{p'=0}^{P}\pi^*_{\bar{\boldsymbol{\theta}}}(\boldsymbol{\theta}_{p'})}}\,|p\rangle_{\mathcal{P}}\,|\boldsymbol{\theta}_0\rangle_{\mathcal{H}_0}\,\big|\bar{\boldsymbol{\theta}}\big\rangle_{\mathcal{H}_1}\,|\boldsymbol{\theta}_p\rangle_{\mathcal{H}_2}\,\big|\pi^*_{\bar{\boldsymbol{\theta}}}(\boldsymbol{\theta}_p)\big\rangle_{\Pi}\,.$$

The remaining states are left as $\sqrt{1-R}\,|\mathrm{FAIL}\rangle$. Notice that for all $p=0,\ldots,P$, $\pi^*_{\bar{\boldsymbol{\theta}}}(\boldsymbol{\theta}_p)=\frac{\pi(\cdot)}{\pi(\boldsymbol{\theta})\mathcal{L}}\in[0,1]$. This guarantees that $\sqrt{1-\pi^*_{\bar{\boldsymbol{\theta}}}}$ and $\sqrt{\pi^*_{\bar{\boldsymbol{\theta}}}}\in[0,1]$. The time complexity of this task is $\mathbb{T}(O_{\pi^*_{\bar{\boldsymbol{\theta}}}})+\mathcal{O}(1)$.

In the final step, Algorithm 5 executes two measurements: the initial measurement targets the $\mathcal{S}$ register, followed by a subsequent measurement on the $\mathcal{H}_2$ register. Should the qubit within the $\mathcal{S}$ register yield a state of 1, the resultant state is altered to:

$$|\mathrm{SUCC}\rangle=\sum_{p=0}^{P}\sqrt{\frac{\pi^*_{\bar{\boldsymbol{\theta}}}(\boldsymbol{\theta}_p)}{\sum_{p'=0}^{P}\pi^*_{\bar{\boldsymbol{\theta}}}(\boldsymbol{\theta}_{p'})}}\,|p\rangle_{\mathcal{P}}\,|\boldsymbol{\theta}_0\rangle_{\mathcal{H}_0}\,\big|\bar{\boldsymbol{\theta}}\big\rangle_{\mathcal{H}_1}\,|\boldsymbol{\theta}_p\rangle_{\mathcal{H}_2}\,\big|\pi^*_{\bar{\boldsymbol{\theta}}}(\boldsymbol{\theta}_p)\big\rangle_{\Pi}\,|1\rangle_{\mathcal{S}}\,.$$

Subsequently, Algorithm 5 performs a measurement on the $\mathcal{H}_2$ register, denoting the outcome as $\boldsymbol{\theta}^{(s)}$, representing the selected state in the s^{th} Markov chain.

Next, we give a proof of Theorem 3.1.

Proof. The success rate R of Algorithm 5 is the probability of the event that the measurement in the register $\mathcal{S}$ yields 1. According to Equation (15), R has the following expression:

$$R=\frac{\sum_{p'=0}^{P}\pi^*_{\bar{\boldsymbol{\theta}}}(\boldsymbol{\theta}_{p'})}{P+1}\geq\min_{p\in\{0,\ldots,P\}}(\pi^*_{\bar{\boldsymbol{\theta}}}(\boldsymbol{\theta}_p)).\tag{16}$$

According to Algorithm 5, with the generated intermediate state $\bar{\boldsymbol{\theta}}$ and the given $\mathcal{L}\geq\max_{\boldsymbol{\theta}\sim\bar{q}(\boldsymbol{\theta}',\cdot);\boldsymbol{\theta}'\in\mathcal{A}}[\frac{\pi(\boldsymbol{\theta})}{\pi(\boldsymbol{\theta})}]$, we set $\pi^*_{\bar{\boldsymbol{\theta}}}(\cdot)=\frac{\pi(\cdot)}{\pi(\boldsymbol{\theta})\mathcal{L}}$.

$$R\geq\min_{p\in\{0,\ldots,P\}}(\pi^*_{\bar{\boldsymbol{\theta}}}(\boldsymbol{\theta}_p))\geq\frac{\min_{\boldsymbol{\theta}\sim\bar{q}(\boldsymbol{\theta}',\cdot);\boldsymbol{\theta}'\in\mathcal{A}}[\frac{\pi(\boldsymbol{\theta})}{\pi(\boldsymbol{\theta})}]}{\mathcal{L}}.$$

Consequently, Theorem 3.1 follows. □

In the Section 3.3, we introduce an advanced version of Algorithm 5 with improved success rate using quantum amplitude amplification (Brassard et al., 2000).

[2]To reduce the burden of notation, we omit the subscript.

3.3 QPMCMC2 with Amplitude Amplification

Let $\chi : X \to \{0, 1\}$ be a Boolean function that partitions the set X into "good" elements (where $\chi(x) = 1$) and "bad" elements. Consider a quantum algorithm $\mathbf{A}$ such that $\mathbf{A}|0\rangle = \sum_{x \in X} \alpha_x |x\rangle$, with $a = \sum_{x \text{ is good}} |\alpha_x|^2$ representing the probability of producing a good element. If we repeatedly run $\mathcal{A}$, measure the output, and use χ to verify the result, the expected number of trials to find a good element is $1/a$.

Quantum amplitude amplification (AA) (Brassard et al., 2000) is a well-studied technique that boost the amplitude of target state among superpositions. Using AA, the number of applications of $\mathbf{A}$ and its inverse needed to find a good element, without intermediate measurements, reduces to $\mathcal{O}(1/\sqrt{a})$. This generalizes Grover's search algorithm (Grover, 1996) and applies even when there is no promise of a unique solution.

Referring to Algorithm 5, which maps $|0\rangle$ to Equation (15) as follows:

$$\sqrt{R}\,|\text{SUCC}\rangle\,|1\rangle_{\mathcal{S}} + \sqrt{1-R}\,|\text{FAIL}\rangle\,|0\rangle_{\mathcal{S}}\,. \tag{17}$$

Applying quantum amplitude amplification with the setting where $\mathbf{A}$ is as defined in Algorithm 5, and the "good" state corresponds to the superposition state associated with $|1\rangle_{\mathcal{S}}$, the probability of measuring the good state is guaranteed to surpass $1/2$ after $\mathcal{O}(1/\sqrt{R})$ applications of lines 1-8 in Algorithm 5. Thus, we have the Corollary 3.2.1.

In Section 4 and 5, we'll discuss the performance of `QPMCMC2` through the implementation on inferring traits on a phylogenetic network.

4 Case Study: Inferring Traits on a Phylogenetic Network

In order to have a clearer image of how it works, we introduce the problem of inferring traits on a phylogenetic network as a suitable case study. In this section, we'll give brief introduction on this problem, then go through specific settings of Algorithm 4 we used in this case. Lastly, we'll provide further analysis on the success rate R corresponding to our specific settings. The implementation results are left to section 5.

4.1 Introduction to Comparative Phylogenetics and Ancestral Trait Reconstruction

Sampling algorithms are essential to the field of comparative phylogenetics, in general, and Bayesian phylogenetics (Suchard et al., 2018), in particular. Here, we start with a fixed phylogenetic tree structure and the traits of observed biological specimens (Figure 1). We make the basic assumption that closely related taxa tend to share the same traits and establish a phylogenetic Ising model to predict the trait combinations of unobserved ancestors. We also adapt this model to deviations from the basic tree graph structure in the context of bacterial reticulate evolution and extend this model to incorporate multiple traits.

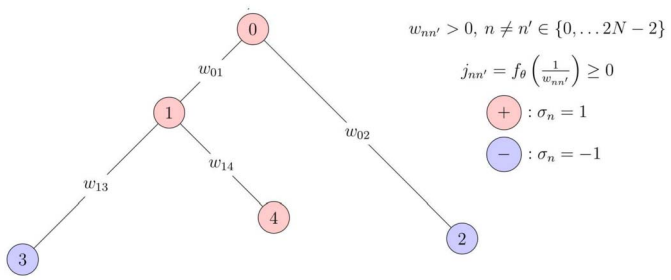


Figure 1: This phylogenetic tree $\mathcal{G}$ has $M_o = 3$ leaf nodes, $M_o - 1 = 2 = M_a$ internal nodes and $2M_o - 1 = 5 = M_{tot}$ total nodes. Leaf nodes represent observed taxa, and internal nodes are unobserved ancestors. We observe a binary trait variable σ_m for each of the leaf nodes and model all (both observed and unobserved) traits σ_m using an Ising model with interactions $j_{mm'}$ which condition on weights $w_{mm'} > 0$.

Specifically, suppose we assume a phylogenetic tree $\mathcal{G}$ (Figure 1) structure that describes the shared evolutionary history giving rise to M_o observed taxa indexed $m \in \{M_o - 1, \ldots, 2M_o - 1 = M_{\text{tot}}\}$. This phylogenetic tree is a rooted, undirected, bifurcating and weighted graph that contains $M_{\text{tot}} = 2M_o - 1$ nodes, M_o of which (corresponding to observed taxa) are leaf nodes, and $M_a = M_o - 1$ are internal nodes. This graph also contains $2M_o - 2$ edges, each of which has its own weight $w_{mm'} > 0$. If no edge exists between the node pair m, m', we say $w_{m,m'} = \infty$. When edges exist, these weights are roughly proportional to the length of time spanning the existence of two organisms. Furthermore, suppose that we observe a binary trait, $\sigma_m \in \{-1, 1\}$ for each of our observed taxa. We then may use a simple Ising model (Daskalakis et al., 2011) to describe the joint distribution over observed and unobserved traits $\boldsymbol{\sigma} = (\sigma_0, \ldots, \sigma_{M_{\text{tot}}-1})$:

$$Pr(\boldsymbol{\sigma}|\beta, \gamma, \mathcal{G}) \propto \exp\left(\beta \sum_{m,m'} j_{mm'}\sigma_m\sigma_{m'}\right), \quad \text{where} \quad j_{mm'} = f_\gamma\left(\frac{1}{w_{mm'}}\right) \tag{18}$$

and $\beta > 0$, $f_\gamma : [0, \infty) \to [0, \infty)$, $f_\gamma(0) = 0$ and f_γ is an increasing function. For example, $f_\gamma(x) = \gamma\sqrt{x}$ for $\gamma > 0$ is one of many possibilities. In the following, we treat γ and β as fixed, but one may learn them simultaneously with the rest of the model parameters in the context of Bayesian inference. From (18), we obtain the likelihood for the observed traits $\boldsymbol{\sigma}_o = (\sigma_{M_a}, \ldots, \sigma_{M_{\text{tot}}-1})$ by conditioning on unobserved ancestral traits $\boldsymbol{\sigma}_a = (\sigma_0, \ldots, \sigma_{M_a-1})$:

$$Pr(\boldsymbol{\sigma}_o|\boldsymbol{\sigma}_a, \beta, \gamma, \mathcal{G}) \propto \exp\left(\beta \sum_{m,m'} j_{m,m'}\sigma_m\sigma_{m'}\right). \tag{19}$$

Placing the uniform prior on the ancestral traits $Pr(\boldsymbol{\sigma}_a) \propto 1$, the posterior distribution

for ancestral traits conditioned on observed traits becomes

$$Pr(\boldsymbol{\sigma}_a|\boldsymbol{\sigma}_o,\beta,\gamma,\mathcal{G}) \propto Pr(\boldsymbol{\sigma}_o|\boldsymbol{\sigma}_a,\beta,\gamma,\mathcal{G}) \cdot Pr(\boldsymbol{\sigma}_a) \propto \exp\left(\beta \sum_{m,m'} j_{m,m'}\sigma_m\sigma_{m'}\right). \quad (20)$$

Within the Bayesian paradigm of statistical inference, the problem of inferring unobserved ancestral traits $\boldsymbol{\sigma}_a$ reduces to simulating from the Ising model (3) while keeping observed traits $\boldsymbol{\sigma}_o$ fixed. Note that it is relatively simple to infer the joint posterior $p(\boldsymbol{\sigma}_a,\beta,\gamma|\boldsymbol{\sigma}_o,\mathcal{G})$, although we do not consider this task here.

We build on this core model in two orthogonal ways. First, we consider the multi-trait scenario and model T binary traits by allotting the m^{th} specimen a spin of the form $\boldsymbol{\sigma}_m = (\sigma_{m,1},\ldots,\sigma_{m,T})$. Following a development analogous to that of (18), (19) and (20), we specify a multi-trait phylogenetic Ising model that leads to the posterior distribution

$$Pr(\boldsymbol{\sigma}_a|\boldsymbol{\sigma}_o,\beta,\gamma,\mathcal{G}) \propto \exp\left(\beta \sum_{m,m'} j_{m,m'}\boldsymbol{\sigma}_m \cdot \boldsymbol{\sigma}_{m'}\right), \quad (21)$$

and $\boldsymbol{\sigma}_a = (\boldsymbol{\sigma}_0,\ldots,\boldsymbol{\sigma}_{M_a-1})$. Second, we consider failures of the bifurcating evolutionary tree hypothesis. Bacterial reticulate evolution (Figure 2) arises from the exchange of genetic material between microbes. In this context, it is appropriate to model evolution using a phylogenetic network. The Neighbor-net (Bryant and Moulton, 2004) algorithm is a popular algorithm for phylogenetic network construction that uses distances between genetic sequences to construct a planar splits graph. In this graph, extremal nodes are observed specimens, and interior nodes are potential ancestors. Whereas this evolutionary network model does not represent an explicit history of individual reticulations, it does represent conflicting signals regarding potential reticulations. These candidate reticulations take the form of the interior boxes that manifest in Figure 3.

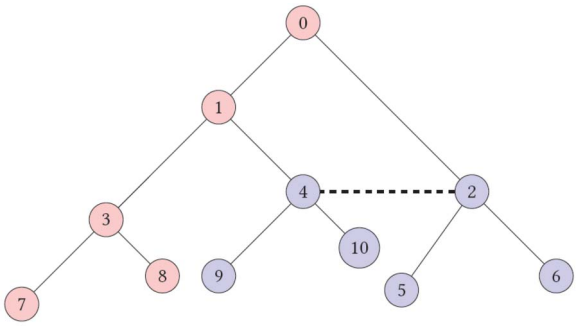


Figure 2: Reticulate evolution. This stylized bacterial phylogenetic network includes a reticulation (dashed line) that characterizes the exchange of genetic material between microbes. Whereas the network deviates from the bifurcating tree hypothesis of Figure 1, the problem of ancestral trait reconstruction is still meaningful.

On the one hand, using such a phylogenetic network as the base lattice structure in phylogenetic Ising models does not alter the mathematical details of the posterior distributions (20) and (21). On the other hand, the existence of cycles in the splits graph makes sampling these distributions significantly more difficult. Holbrook (2023b) shows Algorithm 2's potential for sampling from such challenging target distributions and advances `QPMCMC`, which approximately performs this algorithm. In the following, we present `QPMCMC2` and its massive speedups over the $\mathcal{O}(P)$ complexity of Algorithm 2 and the $\mathcal{O}(\sqrt{P})$ complexity of `QPMCMC`.

4.2 Functions Used in QPMCMC2

The learning of ancestral traits (Section 4.1) within a known phylogenetic network illustrates our algorithm's speed, flexibility and fully-explicit nature. Consider a phylogenetic network $\mathcal{G}(V, E)$, where V denotes a set of M_{tot} vertices and E represents a set of edges. Let V_o be a designated subset of V signifying the observed taxa within this context, and $V_a = V \backslash V_o$ be the complement set of V_o. For the network shown in Figure 2, we have $M_{tot} = |V| = 11$, $V_o = \{5, \ldots, 10\}$ and $V_a = \{0, \ldots, 4\}$. Using the notation of Sections 2.1, 2.2 and 3, we identify any Markov chain state with a collection of ancestral traits thus:

$$\boldsymbol{\theta} = (\boldsymbol{\sigma}_0, \boldsymbol{\sigma}_1, \ldots, \boldsymbol{\sigma}_{M_a-1}), \tag{22}$$

where $\boldsymbol{\sigma}_m = (\sigma_{m,1}, \ldots, \sigma_{m,T})$ for $\sigma_{m,t} \in \{-1, 1\}$. Assuming that each $(m, m') \in E$ possesses an identical weight J, we rewrite the posterior (21) as

$$Pr(\boldsymbol{\sigma}_a|\boldsymbol{\sigma}_o, J, \mathcal{G}) \propto \exp\left(J \sum_{(m,m')\in E} \boldsymbol{\sigma}_m \cdot \boldsymbol{\sigma}_{m'}\right) \quad \text{and set} \quad \pi(\boldsymbol{\theta}) := Pr(\boldsymbol{\sigma}_a|\boldsymbol{\sigma}_o, J, \mathcal{G}). \tag{23}$$

Fixing the observed traits and sampling unobserved ancestral traits using `QPMCMC2` amounts to efficient posterior inference.

In the following, we specify the Tjelmeland distribution $\bar{q}(\cdot, \cdot)$ and detail the target distribution $\pi^*(\cdot)$ for the phylogenetic Ising model. Next, we analyze the qubit requirements when applying Algorithm 4 for this specific inferential task. Finally, the success rate R of Algorithm 4 in this application scenario is introduced.

Tjelmeland Distribution $\bar{q}(\cdot, \cdot)$

We specify the symmetric Tjelmeland distribution $\bar{q}(\cdot, \cdot)$ by defining the distribution $\bar{q}(\boldsymbol{\theta}, \cdot)$ centered at a generic state (22). For each $t \in \{1, \ldots, T\} = \mathcal{T}$ and $m \in \{0, \ldots, M_a - 1\} = V_a$, we define the result state

$$\boldsymbol{\theta}_{m,t} = (\boldsymbol{\sigma}_0, \ldots \boldsymbol{\sigma}'_m, \ldots, \boldsymbol{\sigma}_{M_a-1}), \tag{24}$$

where $\boldsymbol{\sigma}'_m = (\sigma_{m,1}, \ldots - \sigma_{m,t}, \ldots, \sigma_{m,T})$. The vectors $\boldsymbol{\theta}$ and $\boldsymbol{\theta}_{m,t}$ only differ by a negative sign at the trait (t, m). Since there are $M_a T = (M_{\text{tot}} - M_o)T$ possibilities of $\boldsymbol{\theta}_{m,t}$, we

write down $\bar{q}(\boldsymbol{\theta}, \boldsymbol{\theta}')$ formally as

$$\bar{q}(\boldsymbol{\theta}, \boldsymbol{\theta}') = \begin{cases} \frac{1}{M_a T+1} & \text{if } \boldsymbol{\theta}' \in \boldsymbol{\Theta} \\ 0 & \text{otherwise}, \end{cases} \tag{25}$$

where $\boldsymbol{\Theta} = \{\boldsymbol{\theta}_{m,t} : t \in \mathcal{T} \text{ and } m \in V_a\} \cup \{\boldsymbol{\theta}\}$. In words, $\bar{q}(\boldsymbol{\theta}, \cdot)$ is a uniform distribution over the nearest neighbors to $\boldsymbol{\theta}$ and $\boldsymbol{\theta}$ itself.

Notice that we are able to provide L with this given form Tjelmeland distribution $\bar{q}(\cdot, \cdot)$. For two states $\bar{\boldsymbol{\theta}}$ and $\boldsymbol{\theta} \sim \bar{q}(\bar{\boldsymbol{\theta}}, \cdot)$, they only differ in at most one bit. This leads to the existence of $\mathcal{L}$:

$$0 < e^{-2J\deg(\mathcal{G})} \leq \frac{\pi(\boldsymbol{\theta})}{\pi(\bar{\boldsymbol{\theta}})} \leq e^{2J\deg(\mathcal{G})} =: \mathcal{L}. \tag{26}$$

Here, we find the value of $\mathcal{M} := \min_{\boldsymbol{\theta} \sim \bar{q}(\boldsymbol{\theta}', \cdot); \boldsymbol{\theta}' \in \mathcal{A}} [\frac{\pi(\boldsymbol{\theta})}{\pi(\boldsymbol{\theta}')}] = e^{-2J\deg(\mathcal{G})}$ described in Theorem 3.1, too. According to Theorem 3.1, using this Tjelmeland distribution $\bar{q}(\cdot, \cdot)$ `QPMCMC2` are able to run with time complexity independent of P. The success rate of Algorithm 5 is lowerbounded as follows:

$$1 \geq R \geq \frac{\mathcal{M}}{\mathcal{L}} = e^{-4J\deg(\mathcal{G})} \tag{27}$$

With Equation (27), we analyze how graph types influence the success rate R of Algorithm 5: for graphs such as ideal tree graphs and 2D square lattice graphs, $\deg(\mathcal{G})$ is guaranteed to be small, resulting in a higher success rate R. In contrast, graphs like star graphs can exhibit very high degrees $\deg(\mathcal{G})$ depend on the number of "legs", making the success rate R exponentially small. As a result, our proposed method `QPMCMC2` tends to be less efficient in scenarios where $\deg(\mathcal{G})$ is large.

Fortunately, for the ancestral trait reconstruction problems we are interested in, $\deg(\mathcal{G})$ is generally small: for an ideal tree, $\deg(\mathcal{G}) = 3$. The $\deg(\mathcal{G})$ of a realistic phylogenetic tree could exceed 3 due to the reticular evolution, however, the degrees remain small in general. In Section 5, we focus on sampling Ising models from a 2D square lattice graph and a realistic *Salmonella* phylogenetic tree with $\deg(\mathcal{G}) = 8$. With small given Js, in these cases, `QPMCMC2` demonstrates high efficiency with high success rate Rs.

Target Function $\pi^*_{\bar{\boldsymbol{\theta}}}(\cdot)$

To introduce the relative target distribution $\pi^*_{\bar{\boldsymbol{\theta}}}(\cdot)$ in Algorithm 4, we first define a function $f_{(m,t)}$ which maps a state $\boldsymbol{\theta}$ in parameter space to $\mathbb{R}^+$ as follows:

$$f_{(m,t)}(\boldsymbol{\theta}) = \sum_{m'; (m',m) \in E} \sigma_{m,t} \cdot \sigma_{m',t} + \deg(\mathcal{G}),$$

where $\sigma_{m,t}$ is the trait of $\boldsymbol{\theta}$ and $\deg(\mathcal{G})$ is the degree of the phylogenetic network $\mathcal{G}$.

Considering the Tjelmeland distribution (25) in Algorithm 4, each proposal state $\boldsymbol{\theta}_p$ has at most one trait that is different from the intermediate state $\bar{\boldsymbol{\theta}}$. Therefore, the function $\pi^*_{\bar{\boldsymbol{\theta}}}(\cdot)$ in Algorithm 4 that satisfied $\pi^*_{\bar{\boldsymbol{\theta}}}(\cdot) \propto \pi(\cdot)$ can be expressed as

$$\pi^*_{\bar{\boldsymbol{\theta}}}(\boldsymbol{\theta}_p; \bar{\boldsymbol{\theta}}) = \begin{cases} \exp\{-2Jf_{(m_p,t_p)}(\bar{\boldsymbol{\theta}})\} & \text{if } \boldsymbol{\theta}_p \neq \bar{\boldsymbol{\theta}} \\ \exp\{-2J\deg(\mathcal{G})\} & \text{otherwise}, \end{cases} \tag{28}$$

where (m_p, t_p) is the flipped trait in the proposal state $\boldsymbol{\theta}_p$. Note that the image of the function $\pi^*_{\bar{\boldsymbol{\theta}}}(\cdot)$ belongs to $(0, 1]$.

Qubit Requirement

Given the Tjelmeland distribution Equation (25) and the target function Equation (28) introduced in this section, we can analyze the qubit requirement for Algorithm 4. Encoding $\boldsymbol{\theta}_p$ in $\mathcal{H}_2$ requires $\lceil TM_a \log_2(TM_a) \rceil$ qubits, which becomes infeasible for near-term applications. However, this dilemma can be mitigated by encoding (m_p, t_p) (the flipped trait in the proposal state $\boldsymbol{\theta}_p$), which is sufficient for calculating the relative target distribution $\pi^*_{\bar{\boldsymbol{\theta}}}(\cdot)$ and requires only $\lceil \log_2(TM_a) \rceil$ qubits.

Secondly, the calculation of the function $\pi^*_{\bar{\boldsymbol{\theta}}}(\cdot)$ is required for each iteration in Algorithm 4, which is relatively challenging for a near-term quantum computer due to the complexity of computing this exponential function. However, by considering a constant J in Equation (28), we can pre-calculate $2\deg(\mathcal{G}) + 1$ possibilities of the image of Equation (28). Consequently, the calculation of $\pi^*_{\bar{\boldsymbol{\theta}}}(\boldsymbol{\theta}_p)$ can be obtained by providing (m_p, t_p) and consulting a lookup table.

4.3 Success Rate Analysis

In this subsection, instead of deriving the lower bound as in Equation (27), we focus on the expectation value of the success rate R over different random proposal sets. We believe this serves as a better benchmark for evaluating the efficiency of our algorithm. For a given input state $\boldsymbol{\theta}_0$ and a intermediate state $\bar{\boldsymbol{\theta}}$ selected according to $\bar{q}(\boldsymbol{\theta}_0, \cdot)$, we introduce the expectation and variance of R in Theorem 4.1 in terms of $\boldsymbol{\theta}_0$, $\bar{\boldsymbol{\theta}}$ and some problem-dependent parameters, over all possible sets of proposals generated according to $\bar{q}(\bar{\boldsymbol{\theta}}, \cdot)$. We provide the proof of Theorem 4.1 in the Supplement.

Theorem 4.1. *Consider one iteration in Algorithm 4 by providing the number of proposals P, the previous state $\boldsymbol{\theta}_0$, the relative target distribution π^* defined in Equation (28), and the Tjelmeland distribution defined in Equation (25). For a given Ising distribution with the coupling constant J and graph $\mathcal{G}$, the expectation $\mathbb{E}[R]$ and variance $\mathbb{V}[R]$ of R over all possible proposal sets $\{\boldsymbol{\theta}_0, \ldots, \boldsymbol{\theta}_P\}$, where $\boldsymbol{\theta}_p \overset{iid}{\sim} \bar{q}(\bar{\boldsymbol{\theta}}, \cdot)$ for $p = 1, \ldots, P$, can be bounded as follows:*

$$\mathbb{E}[R] \geq Pr(\boldsymbol{\theta}_0)^{\frac{-4}{TM_a}} \exp\left[-2Jdeg(\mathcal{G})(1 + \epsilon_2 + \frac{4}{TM_a})\right](1 - \epsilon_1) + \exp\left[-2Jf_{(m_0,t_0)}(\bar{\boldsymbol{\theta}})\right]\epsilon_1. \tag{29}$$

$$\mathbb{V}[R] \leq \epsilon_1. \tag{30}$$

Here (m_0, t_0) *is the flipped trait in the previous state* $\boldsymbol{\theta}_0$. *We denote* $\epsilon_1 = \frac{1}{P+1}$, $\epsilon_2 = \frac{M_o}{M_a}$, *and* $Pr(\boldsymbol{\theta}) = \exp\left(J \sum_{t\in\mathcal{T}} \sum_{(m,m')\in E} \sigma_{m,t} \cdot \sigma_{m',t}\right)$ *where* $\sigma_{m,t}$ *are the traits of* $\boldsymbol{\theta}$ *as expressed in Equation* (22).

This proposition suggests that the expected success rate R is approximated by $\exp[-2J\deg(\mathcal{G})]$, and the variance of R approaches 0 when ϵ_1 and ϵ_2 are close to 0 when TM_a is large. To address ϵ_1, we can consistently set it to a small value by increasing the number of proposals in `QPMCMC2`. Regarding ϵ_2, we observe that a realistic phylogenetic network graph $\mathcal{G}$ may feature numerous reticulations and M_a being much larger than M_o.

5 Application: Salmonella and Antibiotic Resistance

Conditioned on antibacterial drug resistance scores for 248 *Salmonella* bacterial isolates, we apply our `QPMCMC2` algorithm to the Bayesian inference of ancestral traits on a Neighbor-net phylogenetic network (Section 4.1). Mather et al. (2013); Cybis et al. (2015) previously used this biological dataset to analyze the development of antibiotic resistances within the genus *Salmonella*, but their analyses did not account for bacterial reticulate evolution. Our phylogenetic network, denoted as $\mathcal{G}_{\text{sal}}(V, E)$, comprises $M_{\text{tot}} = 3{,}313$ vertices and $|E| = 5{,}945$ edges. Among these vertices, there are $M_{\text{o}} = 248$ observed taxa, representing the observed biological isolates with known traits. Pertinent to the theoretical developments in Section 5, the degree of our network is $\deg(\mathcal{G}_{\text{sal}}) = 8$. In this section, we use a classical simulator to execute Algorithm 4 and evaluate its efficiency. We apply our algorithm to two Neighbor-net phylogenetic networks: 1) a square lattice graph which contains 100×100 interior nodes with additional 400 extremal nodes along 4 sides, representing the observed isolates, and 2) the aforementioned Neighbor-net phylogenetic network describing the shared evolutionary history of 248 *Salmonella* bacterial isolates (see Figure 3). Additionally, we consider two cases: the single-trait case with the trait number $T = 1$ and the multi-trait case with the trait number $T = 4$, accounting for four traits in each *Salmonella* bacterial isolate. Each figure is plotted with results averaged over 10 repetitions of the experiment. The implementation code is available at `https://github.com/CYLin1113/Quantum-Parallel-MCMC-2`. To check the implementational correctness of our code, we run `QPMCMC2` on a 3 times 3 square lattice model, see Supplement.

In all experiments presented in Section 5, the coupling constants J of the Ising models are set to $J = 0.3$ for square lattices and $J = 0.03$ for the *Salmonella* phylogenetic tree. These selections of J result in high success rates R, and their impact on the running time of `QPMCMC2` (Algorithm 4) can be mitigated by executing multiple copies of Algorithm 5 simultaneously, with additional qubits independent of proposal number P. Please note that in specific cases with large Js or large $\deg(\mathcal{G})$s of the target phylogenetic trees, the acceptance rate Rs could be very small, although they remain independent of P. In such cases, amplitude amplification techniques, as described in Section 3.3, can be applied to efficiently boost the success rate R over 0.5 (Brassard et al., 2000).

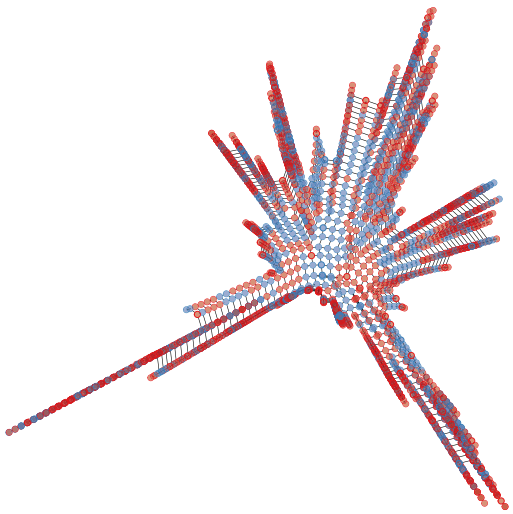
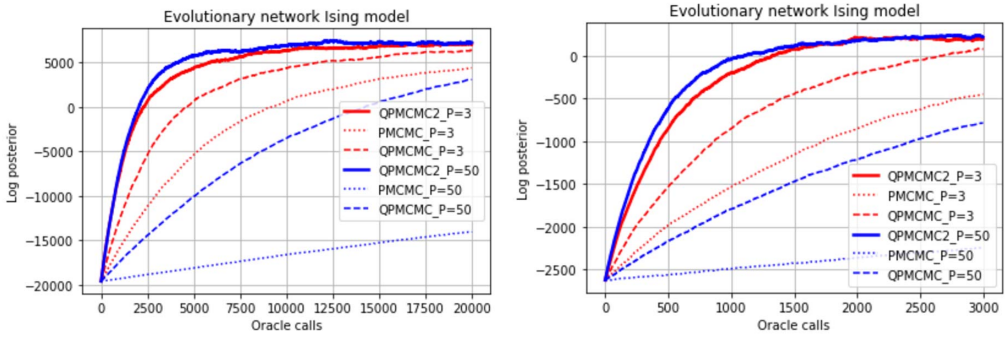


Figure 3: A Neighbor-net phylogenetic network describes the shared evolutionary history of 248 salmonella bacteria isolates. The extremal nodes correspond to the 248 observed isolates, and the $M_a = 3{,}065$ interior nodes correspond to unobserved ancestors. Interior squares are potential reticulation events. Colors (red, resistance; blue, no resistance) are observed and posterior mode resistances to the antibiotic ampicillin for observed microbes and unobserved ancestors, respectively.

From the relationships shown in Figure 4, we demonstrate the value of `QPMCMC2` the classical version of multiproposal MCMC Algorithm 2, labeled as PMCMC, requires $\mathcal{O}(P)$ oracle calls to execute one iteration, which slows down the convergence process when using a larger number of proposals. With the help of quantum parallel computing in our approach, `QPMCMC2` is able to compute all P proposals in parallel with $\mathcal{O}(\log P)$ qubits during each iteration. This resolves the computational bottleneck of using large P values in Algorithm 2, where we find potential advantages exist.

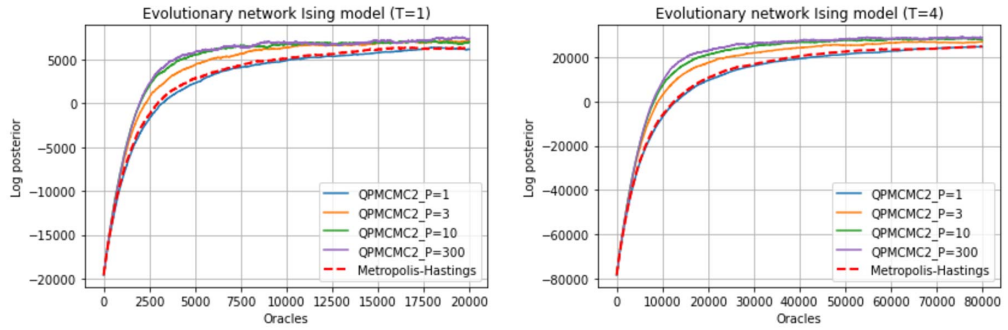
In Figure 5 and Figure 6, we include the Metropolis-Hastings (MH) algorithm in our analysis, as it is generally considered more efficient than Barker-acceptance-based MCMC. From the relationships shown in plot (a) of Figure 5 and Figure 6, it is evident that by evaluating more proposals in a single iteration, `QPMCMC2` converges faster and eventually surpasses the MH algorithm. In the case of the square lattice graph, this speedup is more significant: `QPMCMC2` converges 3.8 times faster compared to the MH algorithm when $P = 300$. These results indicate that, with the aid of quantum parallel computing, this Barker-acceptance-based multiproposal MCMC can approach or even surpass the efficiency of the Metropolis-Hastings algorithm.

We not only apply this method to cases with a single trait ($T = 1$) but also extend `QPMCMC2` to a phylogenetic network Ising model with multiple traits (ampicillin, chloramphenicol, ciprofloxacin, and furazolidone resistances). Plot (b) in Figure 5 and Figure 6 shows the trace plot for the corresponding log-posterior with $T = 4$. As the



(a) Square lattices (b) Salmonella phylogenetic network

Figure 4: Comparison between trace plots generated by the `QPMCMC2`, `QPMCMC`(Holbrook, 2023b) and Algorithm 2 (PMCMC) for $P = 3$ and $P = 50$. For implementation of one MCMC iteration, `QPMCMC2` requires 1 oracle calls of $\pi^*_{\theta}(\cdot)$, while `QPMCMC` requires $\mathcal{O}(\sqrt{P})$ calls and multiproposal MCMC requires $P+1$ calls. In these cases, only `QPMCMC2` improves the converge rate when using a larger P.

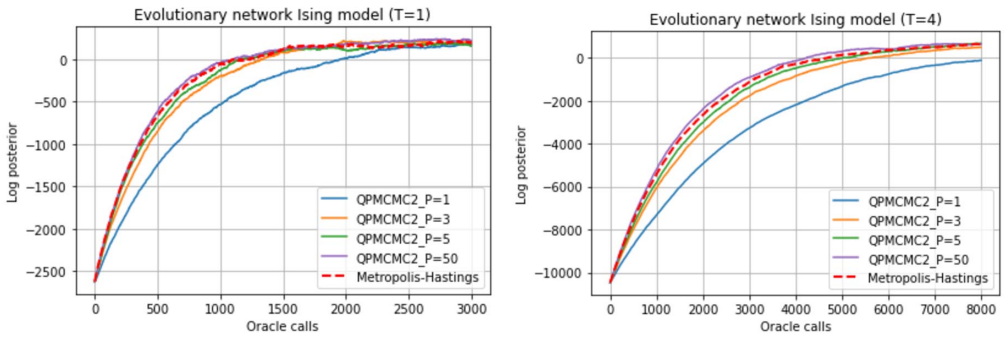


(a) Single-trait (square lattices) (b) Four-trait (square lattices)

Figure 5: Trace plots generated by the `QPMCMC2` algorithm for different numbers of proposals P and Metropolis-Hastings algorithm, tested on the square lattice graph. For both the single-trait and multi-trait problem, increasing P accelerates convergence to higher posterior probability states. Here, we observe that `QPMCMC2` significantly outperforms the Metropolis-Hastings algorithm: for $P = 300$, `QPMCMC2` achieves a 3.8-fold improvement in convergence rate compared to the Metropolis-Hastings algorithm.

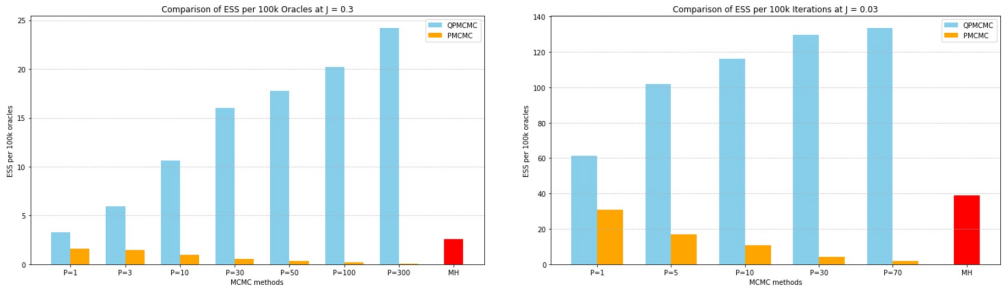
number of parallel proposals P increases, the ancestral trait configuration tends to converge faster while maintaining detailed balance. As expected, the algorithm appears to require approximately T times the number of iterations compared to the $T = 1$ case.

Next, we focus on comparing the ESS per oracle among Algorithm 2 (PMCMC),



(a) Single-trait (salmonella) (b) Four-trait (salmonella)

Figure 6: Trace plots generated by the `QPMCMC2` algorithm for different numbers of proposals P and Metropolis-Hastings algorithm, tested on the phylogenetic tree of salmonella bacteria isolates (Section 3). For both the single-trait and multi-trait problem, increasing P accelerates convergence to higher posterior probability states. Due to the higher degree $\deg(\mathcal{G}_{sal}) = 8$ which leads to a lower success rate (Equation (27)), we failed to observe the same relative performance gain over MH as we observed for the square lattices model.



(a) Square lattices (b) Salmonella phylogenetic tree

Figure 7: Effective sample size (ESS) for the log posterior per 100,000 oracle calls for different numbers of proposals P. ESS produced by `QPMCMC2` shows a significant improvement as P increases, with a noticeable gap compared to the results of the MH algorithm: using `QPMCMC2`, we obtain a 11-fold advantage in the case of square lattices, while it's a 3.5-fold advantage in the case of salmonella phylogenetic tree, comparing to the MH algorithm. In contrast, classical multiproposal MCMC (PMCMC) exhibits the decreasing performance due to the $\mathcal{O}(P)$ complexity using P proposals.

Algorithm 4 (`QPMCMC2`), and the Metropolis-Hastings (MH) algorithm. We estimate ESS using the `Python` package `ArviZ` (Kumar et al., 2019). As shown in Figure 7, the ESS per 10k oracles increases significantly when a larger number of proposals P is

used in `QPMCMC2`. For the square lattice graph and *Salmonella* phylogenetic tree, with sufficiently large P, `QPMCMC2` generates samples with ESS values that are 11 and 3.5 times greater than those produced by the MH algorithm, respectively, demonstrating the remarkable advantages of `QPMCMC2` over this classical approach. The same improvement is not observed in the classical multiproposal MCMC (labeled as PMCMC) due to its $\mathcal{O}(P)$ cost.

In both the single- and the multi-trait experiments, we observe that using a large proposal count P when applying multiproposal MCMC leads to improved convergence. This highlights two major strengths of the quantum algorithm we propose. First, `QPMCMC2` obtains an exponential speedup: for large P in multiproposal MCMC algorithms, we reduce the dependence of P of the time complexity from $\mathcal{O}(P)$ to $\mathcal{O}(1)$ with $\mathcal{O}(\log P)$ ancillary qubits. This is an exponential speedup as a function of P and resolves the bottleneck of the original Algorithm 2. Second, `QPMCMC2` provides accelerated sampling for real-world problems: we have demonstrate the benefits for our quantum algorithm in accelerating sampling for a realistic and non-trivial class of graphical models. This quantum algorithm shows the potential to accelerate Bayesian reconstruction of bacterial antibiotic resistances, an important problem in medicine and evolutionary biology.

6 Discussion

Quantum computing is set to revolutionize certain areas of science (computational physics/chemistry), but its future impact on many other areas remains unknown. Similarly, quantum computing promises extreme speedups for certain technical challenges (prime factorization in cryptography) while benefits for other prominent challenges remain elusive. In particular, many statisticians may wonder how quantum computing will eventually impact their day-to-day data scientific pipelines. Here, we develop a fast quantum algorithmic implementation of an advanced MCMC algorithm. Given (1) that MCMC is a workhorse algorithm of modern statistical inference and (2) the significant scale of current investment in quantum computing knowledge and infrastructure, other approaches to quantum accelerated MCMC are sure to follow. We find three particular avenues of future research interesting.

First, it is clear that the strategies we develop here will provide similar exponential speedups for other advanced MCMC algorithms. For example, the locally-balanced proposal scheme of Zanella (2019) generates proposals by selecting among members of a fixed proposal set with probability proportional to the square-root target function. One may further combine this strategy with other MCMC approaches that encourage fast mixing. Nonreversible Metropolis-Hastings schemes (Turitsyn et al., 2011) maintain momentum between successive MCMC iterations and can lead to orders-of-magnitude faster convergence when sampling from discrete models. Unfortunately, changing directions in this framework requires a significant number of target evaluations. Our strategy may confer exponential speedups here as well. Second, our Ising model case study makes use of local moves, but quantum computers may prove useful for generating proposals far away from the current position in a manner that preserves high probabilities of acceptance. Layden et al. (2023) achieve this but must compute the target probability at the

new proposal state from scratch using a conventional computer. Ideally, one would be able to make global jumps in a manner that uses the quantum device for both proposal and acceptance steps, as we do here. Finally, the Ising model is a foundational model that one can also use to approximate diverse targets (Leng et al., 2023), but unlocking the power of quantum computing for statistics will also require adapting additional discrete models to frameworks like ours. One powerful possibility is applications that include Bayesian tree-based classifiers and regression models (Chipman et al., 2012; Ma, 2017). An open question is whether these methods may also confer exponential speedups when sampling discrete topologies for the trees that underpin these models.

Funding

This work was supported through National Institutes of Health grants R01 AI153044, R01 AI162611 and K25 AI153816. PL and MAS acknowledge support from the European Union's Horizon 2020 research and innovation programme (grant agreement no. 725422-ReservoirDOCS) and from the Wellcome Trust through project 206298/Z/17/Z. PL acknowledges support from the Research Foundation – Flanders ('Fonds voor Wetenschappelijk Onderzoek – Vlaanderen', G0D5117N and G051322N) and from the European Union's Horizon 2020 project MOOD (grant agreement no. 874850). AJH acknowledges support from the National Science Foundation (grants DMS 2152774 and DMS 2236854).

Supplementary Material

Supplement to "Quantum Speedups for Multiproposal MCMC" (DOI: 10.1214/25-BA1546SUPP; .pdf). Contains proofs and further simulation experiments.

References

Barker, A. A. (1965). "Monte carlo calculations of the radial distribution functions for a proton? electron plasma." *Australian Journal of Physics*, 18(2): 119–134. 4

Brassard, G., Hoyer, P., Mosca, M., and Tapp, A. (2000). "Quantum amplitude amplification and estimation." MR1947332. doi: https://doi.org/10.1090/conm/305/05215. 9, 13, 14, 20

Bryant, D. and Moulton, V. (2004). "Neighbor-Net: An Agglomerative Method for the Construction of Phylogenetic Networks." *Molecular Biology and Evolution*, 21(2): 255–265.
URL https://doi.org/10.1093/molbev/msh018 3, 16

Calderhead, B. (2014). "A general construction for parallelizing Metropolis- Hastings algorithms." *Proceedings of the National Academy of Sciences*, 111(49): 17408–17413. 1

Campos, R., Casares, P. A., and Martin-Delgado, M. (2023). "Quantum Metropolis Solver: a quantum walks approach to optimization problems." *Quantum Machine Intelligence*, 5(2): 28. 2

Chipman, H. A., George, E. I., and McCulloch, R. E. (2012). "BART: Bayesian additive regression trees." *Annals of Applied Statistics*, 6(1): 266–298. MR2758172. doi: https://doi.org/10.1214/09-AOAS285. 25

Cybis, G. B., Sinsheimer, J. S., Bedford, T., Mather, A. E., Lemey, P., and Suchard, M. A. (2015). "Assessing phenotypic correlation through the multivariate phylogenetic latent liability model." *The annals of applied statistics*, 9(2): 969. MR3371344. doi: https://doi.org/10.1214/15-AOAS821. 3, 20

Daskalakis, C., Mossel, E., and Roch, S. (2011). "Evolutionary trees and the Ising model on the Bethe lattice: a proof of Steel's conjecture." *Probability Theory and Related Fields*, 149(1-2): 149–189. MR2773028. doi: https://doi.org/10.1007/s00440-009-0246-2. 15

Delmas, J.-F. and Jourdain, B. (2009). "Does waste recycling really improve the multi-proposal Metropolis–Hastings algorithm? An analysis based on control variates." *Journal of applied probability*, 46(4): 938–959. MR2582699. doi: https://doi.org/10.1239/jap/1261670681. 1

Dorner, U., Demkowicz-Dobrzanski, R., Smith, B. J., Lundeen, J. S., Wasilewski, W., Banaszek, K., and Walmsley, I. A. (2009). "Optimal quantum phase estimation." *Physical review letters*, 102(4): 040403. 2

Duane, S., Kennedy, A., Pendleton, B. J., and Roweth, D. (1987). "Hybrid Monte Carlo." *Physics Letters B*, 195(2): 216–222.
URL https://www.sciencedirect.com/science/article/pii/037026938791197X MR3960671. doi: https://doi.org/10.1016/0370-2693(87)91197-x. 4

Durr, C. and Hoyer, P. (1996). "A quantum algorithm for finding the minimum." *arXiv preprint arXiv:quant-ph/9607014*. 2

Felsenstein, J. (1985). "Phylogenies and the comparative method." *The American Naturalist*, 125(1): 1–15. 3

Frenkel, D. (2004). "Speed-up of Monte Carlo simulations by sampling of rejected states." *Proceedings of the National Academy of Sciences*, 101(51): 17571–17575. 1

Gelman, A., Carlin, J. B., Stern, H. S., and Rubin, D. B. (1995). *Bayesian data analysis*. Chapman and Hall/CRC. MR1385925. 4, 9, 10

Glatt-Holtz, N. E., Holbrook, A. J., Krometis, J. A., and Mondaini, C. F. (2024a). "Parallel MCMC algorithms: Theoretical foundations, algorithm design, case studies." *Transactions of Mathematics and its Applications (In Press)*. MR4790268. doi: https://doi.org/10.1093/imatrm/tnae004. 2, 5

Glatt-Holtz, N. E., Holbrook, A. J., Krometis, J. A., and Mondaini, C. F. (2024b). "Parallel MCMC algorithms: theoretical foundations, algorithm design, case studies." *Transactions of Mathematics and Its Applications*, 8(2): tnae004.
URL https://doi.org/10.1093/imatrm/tnae004 MR4790268. doi: https://doi.org/10.1093/imatrm/tnae004. 6

Grover, L. K. (1996). "A fast quantum mechanical algorithm for database search." In *Proceedings of the twenty-eighth annual ACM symposium on Theory of computing*, 212–219. MR1427516. doi: https://doi.org/10.1145/237814.237866. 2, 14

Hassler, G. W., Magee, A. F., Zhang, Z., Baele, G., Lemey, P., Ji, X., Fourment, M., and Suchard, M. A. (2023). "Data Integration in Bayesian Phylogenetics." *Annual Review of Statistics and Its Application*, 10: 353–377. MR4567797. doi: https://doi.org/10.1146/annurev-statistics-033021-112532. 3

Hastings, W. K. (1970). "Monte Carlo sampling methods using Markov chains and their applications." *Biometrika*, 57(1): 97–109.
URL https://doi.org/10.1093/biomet/57.1.97 MR3363437. doi: https://doi.org/10.1093/biomet/57.1.97. 1

Holbrook, A. J. (2023a). "Generating MCMC proposals by randomly rotating the regular simplex." *Journal of Multivariate Analysis*, 194: 105106. MR4524586. doi: https://doi.org/10.1016/j.jmva.2022.105106. 2, 5, 6

Holbrook, A. J. (2023b). "A quantum parallel Markov chain Monte Carlo." *Journal of Computational and Graphical Statistics*, 32(4): 1402–1415. MR4669256. doi: https://doi.org/10.1080/10618600.2023.2195890. 2, 5, 9, 10, 17, 22

Kumar, R., Carroll, C., Hartikainen, A., and Martin, O. (2019). "ArviZ a unified library for exploratory analysis of Bayesian models in Python." *Journal of Open Source Software*, 4(33): 1143. 23

Layden, D., Mazzola, G., Mishmash, R. V., Motta, M., Wocjan, P., Kim, J.-S., and Sheldon, S. (2023). "Quantum-enhanced markov chain Monte Carlo." *Nature*, 619(7969): 282–287. 24

Leng, J., Hickman, E., Li, J., and Wu, X. (2023). "Quantum hamiltonian descent." *arXiv preprint arXiv:2303.01471*. 25

Lin, Chin-Yi, Chen, Kuo-Chin, Lemey, P., Suchard, M. A., Holbrook, A. J., and Hsieh, Min-Hsiu. (2025). "Supplement to "Quantum Speedups for Multiproposal MCMC"." *Bayesian Analysis*.
URL https://doi.org/10.1214/25-BA1546SUPP 3

Luo, X. and Tjelmeland, H. (2019). "A multiple-try Metropolis–Hastings algorithm with tailored proposals." *Computational Statistics*, 34(3): 1109–1133. MR3982858. doi: https://doi.org/10.1007/s00180-019-00878-y. 1

Ma, L. (2017). "Adaptive Shrinkage in Pólya Tree Type Models." *Bayesian Analysis*, 12(3). MR3655876. doi: https://doi.org/10.1214/16-BA1021. 25

Mather, A., Reid, S., Maskell, D., Parkhill, J., Fookes, M., Harris, S., Brown, D., Coia, J., Mulvey, M., Gilmour, M., et al. (2013). "Distinguishable epidemics of multidrug-resistant Salmonella Typhimurium DT104 in different hosts." *Science*, 341(6153): 1514–1517. 3, 20

Metropolis, N., Rosenbluth, A. W., Rosenbluth, M. N., Teller, A. H., and Teller, E.

(1953). "Equation of state calculations by fast computing machines." *The journal of chemical physics*, 21(6): 1087–1092. 1, 4

Montanaro, A. (2015). "Quantum speedup of Monte Carlo methods." *Proceedings of the Royal Society A: Mathematical, Physical and Engineering Sciences*, 471(2181): 20150301. MR3404289. doi: https://doi.org/10.1098/rspa.2015.0301. 2

Neal, R. M. (2011). "MCMC using ensembles of states for problems with fast and slow variables such as Gaussian process regression." *arXiv preprint arXiv:1101.0387*. 1

Somma, R. D., Boixo, S., Barnum, H., and Knill, E. (2008). "Quantum Simulations of Classical Annealing Processes." *Physical Review Letters*, 101: 130504. URL https://link.aps.org/doi/10.1103/PhysRevLett.101.130504 2

Suchard, M. A., Lemey, P., Baele, G., Ayres, D. L., Drummond, A. J., and Rambaut, A. (2018). "Bayesian phylogenetic and phylodynamic data integration using BEAST 1.10." *Virus evolution*, 4(1): vey016. 14

Szegedy, M. (2004). "Quantum speed-up of Markov chain based algorithms." In *45th Annual IEEE symposium on foundations of computer science*, 32–41. IEEE. 2

Tierney, L. (1994). "Markov chains for exploring posterior distributions." *the Annals of Statistics*, 1701–1728. MR1329166. doi: https://doi.org/10.1214/aos/1176325750. 2

Tjelmeland, H. (2004). "Using all Metropolis–Hastings proposals to estimate mean values." Technical report, Citeseer. 1, 5, 6

Turitsyn, K. S., Chertkov, M., and Vucelja, M. (2011). "Irreversible Monte Carlo algorithms for efficient sampling." *Physica D: Nonlinear Phenomena*, 240(4-5): 410–414. 24

Yoder, T. J., Low, G. H., and Chuang, I. L. (2014). "Fixed-point quantum search with an optimal number of queries." *Physical review letters*, 113(21): 210501. 2

Zanella, G. (2019). "Informed proposals for local MCMC in discrete spaces." *Journal of the American Statistical Association*. MR4107684. doi: https://doi.org/10.1080/01621459.2019.1585255. 24

Zhang, Z., Nishimura, A., Bastide, P., Ji, X., Payne, R. P., Goulder, P., Lemey, P., and Suchard, M. A. (2021). "Large-scale inference of correlation among mixed-type biological traits with phylogenetic multivariate probit models." *The Annals of Applied Statistics*, 15(1): 230–251. MR4255276. doi: https://doi.org/10.1214/20-aoas1394. 3

Zhang, Z., Nishimura, A., Trovao, N. S., Cherry, J. L., Holbrook, A. J., Ji, X., Lemey, P., and Suchard, M. A. (2023). "Accelerating Bayesian inference of dependency between mixed-type biological traits." *PLOS Computational Biology*, 19(8): e1011419. 3