Journal of Machine Learning Research (2025) 1-25 Submitted 7/24; Revised 5/25; Published 11/25

Graph-accelerated Markov Chain Monte Carlo using Approximate Samples

Leo L Duan LI.DUAN@UFL.EDU
Department of Statistics
University of Florida, USA

Anirban Bhattacharya ANIRBANB@STAT.TAMU.EDU
Department of Statistics
Texas A&M University, USA

Editor: Matthew Hoffman

Abstract

It has become increasingly easy nowadays to collect approximate posterior samples via fast algorithms such as variational Bayes, but concerns exist about the estimation accuracy. It is tempting to build solutions that exploit approximate samples in a canonical Markov chain Monte Carlo framework. As the dimension increases, a major barrier is that the approximate sample tends to have a low Metropolis–Hastings acceptance rate when used as a proposal. In this article, we propose a simple solution named graph-accelerated Markov Chain Monte Carlo. We build a graph with each node assigned to an approximate sample, then run Markov chain Monte Carlo with random walks over the graph. We optimize the graph edges to enforce small differences in posterior density/probability between nodes, while encouraging edges to have large distances in the parameter space. The graph allows us to accelerate a canonical Markov transition kernel through mixing with a large-jump Metropolis-Hastings step. The acceleration is easily applicable to existing Markov chain Monte Carlo algorithms. We theoretically quantify the rate of acceptance as dimension increases, and show the effects on improved mixing time. We demonstrate improved mixing performances for challenging problems, such as those involving multiple modes, non-convex density contour, or large-dimension latent variables.

Keywords: Conductance, Graph Bottleneck, Mixture Transition Kernel, Spanning Tree Graph, Latent Gaussian Model

1. Introduction

Bayesian approaches are convenient for incorporating prior information, enabling model-based uncertainty quantification, and facilitating flexible model extension. Among the sampling algorithms for posterior computation, Markov chain Monte Carlo is arguably the most popular method and uses a Markov transition kernel (a conditional distribution given the current parameter value) to produce an update to the parameter. As one can flexibly choose transition kernel under a set of fairly straightforward principles, the algorithm can handle parameters in a multi-dimensional and complicated space, such as those involving latent variables (Tanner and Wong, 1987; Gelman, 2004), constraints (Gelfand et al., 1992; Duan et al., 2020; Presman and Xu, 2023), discrete or hierarchical structure (Chib and Carlin, 1999; Papaspiliopoulos et al., 2007); among many others. A major strength is that many Markov chain Monte Carlo algorithms have an *exact* convergence guarantee (Roberts and Rosenthal, 2004) — as the number of Markov chain iterations goes to infinity, the distribution of the Markov chain samples converge to the target posterior distribution. There is a rich literature on establishing such convergence guarantee for popular Markov chain Monte Carlo algorithms. The literature covers Gibbs sampling (Roberts and Polson, 1994; Gelfand, 2000), Metropolis–Hastings (Roberts and Smith, 1994; Jones et al., 2014), slice sampling (Roberts and Rosenthal, 1999; Neal, 2003; Natarovskii et al., 2021), hybrid Monte Carlo (Durmus et al., 2017, 2020), and non-reversible extensions such as piecewise deterministic Markov process (Costa and Dufour, 2008; Fearnhead et al., 2018; Bierkens et al., 2019).

On the other hand, Markov chain Monte Carlo is not without its challenges, especially in advanced models that involve a high-dimensional parameter, latent correlation, or hierarchical structure. For example, there is recent literature characterizing the curse of dimensionality that leads to slow convergence of some routinely used Gibbs sampler (Johndrow et al., 2019), which has subsequently inspired many remedial algorithms (Johndrow et al., 2020; Vono et al., 2022). More broadly speaking, the computing inefficiency happens when the Markov transition kernel creates high auto-correlation along the chain — under such a scenario, the effective change of parameter becomes quite small over many iterations. Due to this slow mixing issue, practical issues arise in applications: the sampler may take a long time to move away from the start region (where the chain is initialized) into the high posterior density/probability region; it may have difficulty crossing low-probability region that divides multiple posterior modes; it may lack an efficient proposal distribution that could significantly change the parameter value, due to the dependence on a high-dimensional latent variable (Rue et al., 2009).

Conventionally, one often views optimization as a competitor to Markov chain Monte Carlo, and as a class of algorithms incompatible with Bayesian models. This belief has been rapidly changed by the recent study of diffusion-based methods. To give a few examples, it has been pointed out that the (unadjusted overdamped) Langevin diffusion algorithm is equivalent to a gradient descent algorithm adding a Gaussian random walk in each step (Roberts and Tweedie, 1996; Roberts and Rosenthal, 1998; Dalalyan, 2017); as a result, similar acceleration for gradient descent could be applied in the diffusion algorithm for posterior approximation (Ma et al., 2021). Mimicking second-order optimization such as Newton descent, one could obtain a rapid diffusion on the probability space using the Fisher information metric (Girolami and Calderhead, 2011).

In parallel to these developments, variational algorithms have become very popular. One uses optimization to minimize a statistical divergence between the posterior distribution and a prescribed variational distribution, from which one could draw independent samples as a posterior approximation. The choice of variational distribution spans from mean-field approximation (uncorrelated parameter elements) (Blei et al., 2017), variational boosting (mixture) (Miller et al., 2017; Campbell and Li, 2019), to normalizing flow neural networks (black-box non-linear transform) (Papamakarios et al., 2021). In particular, the normalizing flow neural networks have received considerable attention lately due to the high computing efficiency under modern computing platforms, and its high flexibility during density approximation (for continuous parameter). On the other hand, concerns exist about the accuracy of uncertainty estimates. In particular, the prescribed variational distribution may not be adequately flexible to approximate the target posterior. For example, the mean-field variational methods lead to a wrong estimate of posterior covariance, which has motivated the development of alternative covariance estimator (Giordano et al., 2018). For neural network-based approximation, although positive result has been obtained for approximating the class of sub-Gaussian and log-Lipschitz posterior densities via a feed-forward neural network (Lu and Lu, 2020), for normalizing flow (as a neural network restricted for one-to-one mapping), severe limitations have been discovered even for approximating some simple distributions (Dupont et al., 2019; Kong and Chaudhuri, 2020). Practically, another concern is in the lack of diagnostic measures on the accuracy of approximation — since the target posterior density/probability often contains intractable normalizing constant, usually we do not know how close the minimized statistical divergence is to zero.

Naturally, it is tempting to consider combining strengths from both approximation methods and the canonical Markov chain Monte Carlo framework. One intuitive idea is to adopt the approximate samples to build a proposal distribution and accept or reject each drawn proposal via a Metropolis-Hastings adjustment step. Nevertheless, a technical barrier is the acceptance rate often rapidly decays to zero, as the parameter (and latent variable) dimension grows, which has inspired several solutions. One remedy is to divide the proposal into several blocks (each in low dimension), then accept or reject the change in each block sequentially via a Metropolis-Hastings-within-Gibbs sampler, which unfortunately often leads to slow mixing. Another idea is to use each approximate sample as an initial state and run multiple parallel Markov chains (Hoffman et al., 2018). Lastly, a recently popularized solution is to combine approximate samples with a Metropolis-adjusted diffusion algorithm such as Hamiltonian Monte Carlo (Betancourt et al., 2017). For example, Gabrié et al. (2022) interleave two Metropolis-Hastings steps, one using Langevin/Hamiltonian diffusion and one using independent proposal from an approximate sampler (normalizing flow); Toth et al. (2020) approximate the diffusion by training the gradients of a neural network to match the time derivative of the Hamiltonian, gaining higher efficiency than a differential equation integrator. In addition, there are a few new adaptive Markov chain Monte Carlo algorithms for multi-modal posterior estimation (Pompe et al., 2020; Yi et al., 2023), based on interleaving the mode estimation steps and proposal moves between modes. The readers can find comprehensive reviews on accelerated Markov chain algorithms in Robert et al. (2018), and on recent machine learning algorithms in Winter et al. (2024).

Despite similar motivation, our goal is to build a simple and general Markov chain Monte Carlo algorithm for which one could exploit an approximate sampler in an *out-of-box manner* without any need for customization. The chosen approximation method could be as advanced as a normalizing flow neural network, or as simple as an existing Markov chain Monte Carlo (which could suffer from slow mixing). Our main idea is to first collect approximate samples, build a graph to connect these samples and run the Markov chain Monte Carlo via a mixture transition kernel of a canonical baseline kernel and a graph jump step. We will demonstrate how this method leads to accelerated mixing of the Markov chains in both theory and applications.

2. Method

Let $\theta \in \Theta \subseteq \mathbb{R}^p$ be the parameter of interest and our goal is to draw samples from the posterior distribution $\Pi(\theta \mid y) \propto L(y;\theta)\Pi_0(\theta)$, with L the likelihood and Π_0 the prior. To be general, this form also extends to augmented likelihood containing latent variable z in addition to the parameter of interest $\tilde{\theta}$, $\Pi\{(\tilde{\theta}, z) \mid y\} \propto L(y, z; \tilde{\theta})\Pi_0(\tilde{\theta})$, for which one may consider $\theta = (\tilde{\theta}, z)$. We use Π to represent both distribution and probability kernel (density or mass function). We will primarily focus on continuous θ, although the method can be extended to discrete θ.

2.1 Graph-accelerated Markov Chain Monte Carlo

Using an existing posterior approximation algorithm for $\Pi(\cdot \mid y)$, suppose we have collected m approximate samples $\beta = (\beta^1, \ldots, \beta^m)$. Using those m samples, we first build an undirected and connected graph $G = (V, E_G)$, with node set $V = (1, \ldots, m)$, and edge set $E_G = \{(i,j)\}$; see Section 2.2 for details. By connectedness, we mean that for any two i and j, there is a path consisting of edges in E_G between two nodes, $\text{path}(i,j) = \{(i,k_1),(k_1,k_2),\ldots,(k_l,j)\} \subseteq E_G$. Accordingly, we define a graph-walk distance between nodes $\text{dist}(i,j) = \min_{\text{all path}(i,j)} |\text{path}(i,j)|$, with $|\cdot|$ the set cardinality; and accordingly define $B(j;r) = \{k : \text{dist}(j,k) \le r\}$, a ball generated by this distance centered at node j with radius r. We view (G,β) as a graph with node attributes: each β^j is a location attribute for node j.

We consider an existing *baseline* Markov chain Monte Carlo algorithm with Markov transition kernel $\mathcal{K}(\theta,\cdot)$ and the posterior distribution $\Pi(\cdot \mid y)$ as its stationary distribution, i.e., if $\theta \sim \Pi(\cdot \mid y)$ and $\theta' \mid \theta \sim \mathcal{K}(\theta,\cdot)$, then $\theta' \sim \Pi(\cdot \mid y)$. Examples of such baseline samplers include random-walk Metropolis, Gibbs sampler, or Hamiltonian Monte Carlo sampler, etc. Our goal is to use (G,β) to accelerate the mixing of this baseline Markov chain in exploring the posterior surface.

To this end, we draw Markov chain samples via a two-component mixture Markov transition kernel:

$$(\theta^{t+1} \mid \theta^t) \sim \mathcal{R}(\theta^t,\cdot) = w\mathcal{Q}(\theta^t,\cdot) + (1-w)\mathcal{K}(\theta^t,\cdot), \tag{1}$$

where $w \in [0,1)$ is a tuning parameter. In each iteration, with probability $(1-w)$, the sampler will update θ using $\mathcal{K}(\theta^t,\cdot)$, the baseline algorithm; with probability w, the sampler will use $\mathcal{Q}(\theta^t,\cdot)$ to take a *graph jump* consisting of the following steps:

1. (Project to a node) Find the projection of θ^t to one β^j, $j = \mathbb{N}(\theta^t) := \arg\min_l \|\beta^l - \theta^t\|$.

2. (Walk on the graph) Draw a new node i uniformly from the ball $B(j;r)$.

3. (Relaxation from β^i) Draw a proposal θ^* from a relaxation distribution $F(\theta^* \mid \beta^i, \theta^t)$.

4. (Metropolis–Hastings adjustment) Accept θ^* as θ^{t+1} with probability

$$\alpha(\theta^t,\theta^*) = \min\left[1, \frac{\Pi(\theta^* \mid y)|B(i;r)|^{-1}F(\theta^t \mid \beta^j,\theta^*)}{\Pi(\theta^t \mid y)|B(j;r)|^{-1}F(\theta^* \mid \beta^i,\theta^t)}\right] 1\left[\mathbb{N}(\theta^*) = i\right]; \tag{2}$$

   otherwise keep θ^{t+1} as the same as θ^t.

Here $\|a-b\|$ refers to some distance between a and b, such as Euclidean distance or Mahalanobis distance $\sqrt{(a-b)'S^{-1}(a-b)}$, with S some $p \times p$ positive definite matrix, for example, the sample covariance matrix based on β. We assume $\mathbb{N}(\theta)$ is unique almost everywhere with respect to the posterior distribution of θ, and use F to allow θ^* to take different values from β^i. For low-dimensional θ, one could use common continuous F such as multivariate Gaussian or uniform centered at β^i. We will discuss specific choices of distance and F suitable for high dimensional θ in Section 2.3.

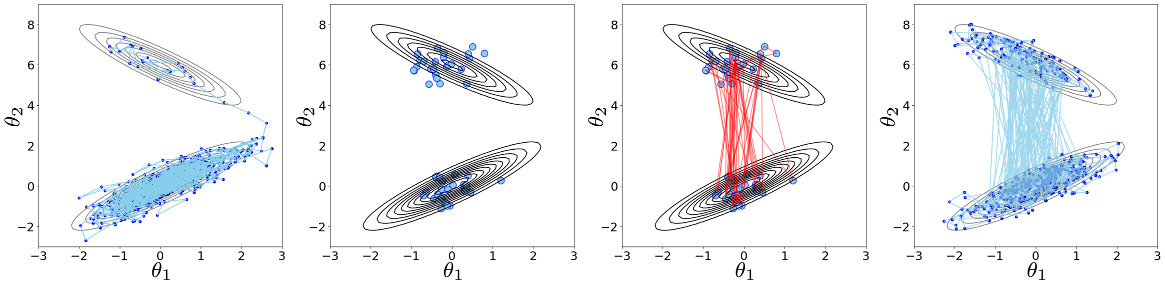
Theorem 1 *The graph jump step satisfies the detailed balance condition:* $\Pi(\theta^t \mid y)\mathcal{Q}(\theta^t,\theta^{t+1}) = \Pi(\theta^{t+1} \mid y)\mathcal{Q}(\theta^{t+1},\theta^t)$.

Remark 2 *Since it is possible that the relaxed θ^* has $\mathbb{N}(\theta^*) \neq i$, we use the indicator function in the acceptance rate to ensure reversibility. An alternative is to use acceptance rate*

$$\min\left[1, \frac{\Pi(\theta^* \mid y)|B[\mathbb{N}(\theta^*);r]|^{-1}\sum_{l\in B[\mathbb{N}(\theta^*);r]} F(\theta^t \mid \beta^l, \theta^*)}{\Pi(\theta^t \mid y)|B[\mathbb{N}(\theta^t);r]|^{-1}\sum_{i\in B[\mathbb{N}(\theta);r]} F(\theta^* \mid \beta^i, \theta^t)}\right],$$

which would be feasible to compute, provided $B[\mathbb{N}(\theta^t);r]$ and $B[\mathbb{N}(\theta^*);r]$ are not too large. For generality, we will use (2) *in this article.*

To illustrate the idea, we use a toy example of sampling from a two-component Gaussian mixture, $\theta \sim 0.6\mathrm{N}\{\left(\begin{smallmatrix}0\\0\end{smallmatrix}\right), \left(\begin{smallmatrix}1 & 0.9\\0.9 & 1\end{smallmatrix}\right)\} + 0.4\mathrm{N}\{\left(\begin{smallmatrix}0\\6\end{smallmatrix}\right), \left(\begin{smallmatrix}1 & -0.9\\-0.9 & 1\end{smallmatrix}\right)\}$. We consider the random-walk Metropolis algorithm with proposal $\theta^* \sim \mathrm{Unif}(\theta^t - \tilde{s}1_p, \theta^t + \tilde{s}1_p)$, with the step size $\tilde{s} = 1$, as the baseline algorithm (corresponding to transition kernel $\mathcal{K}$). Due to the high correlation within each mixture component and the low-density region separating the two modes, the random-walk Metropolis is stuck in one component for a long time as shown in Figure 1(a).



(a) Traceplot of a Markov chain produced by a canonical random-walk Metropolis algorithm. The sampler is stuck in one component for a long time, before moving to another.

(b) Approximate samples produced by a variational algorithm. Fifty approximate samples are drawn from a two-component Gaussian mixture, with each component having an isotropic covariance and equal mixture weight.

(c) A graph connecting the approximate samples, with the graph optimized to increase the parameter distances over the edges, while ensuring small density differences between the parameters on adjacent nodes.

(d) Traceplot of a Markov chain produced by the graph-accelerated algorithm. The sampler now jumps frequently over the two components.

Figure 1: An illustrative toy example: using a graph to accelerate the random-walk Metropolis algorithm for Markov chain sampling from a two-component Gaussian mixture distribution.

For acceleration, we use a variational distribution $\beta \sim 0.5\mathrm{N}\{\left(\begin{smallmatrix}0\\0\end{smallmatrix}\right), \sigma_1^2 I\} + 0.5\mathrm{N}\{\left(\begin{smallmatrix}0\\6\end{smallmatrix}\right), \sigma_2^2 I\}$, with σ_1^2 and σ_2^2 numerically calculated via minimizing the Kullbeck-Leibler divergence through the `numpyro` package (Phan et al., 2019), and then draw 50 independent approximate samples from the resulting variational approximation. We obtain a simple graph G, a spanning tree (Kruskal, 1956; Prim, 1957), that connects all those samples, and use the graph-accelerated algorithm in (1) with $w = 0.3$ and Gaussian for F. As shown in Figure 1(d), the sampler now jumps rapidly over the two components. We run both the baseline and accelerated algorithms for $10,000$ iterations, and using the effective sample size of θ_2 per iteration as a benchmark for mixing: the one of the baseline algorithm is 0.04%, and the one of the accelerated version is 4.5%, hence is roughly 100 times faster.

Remark 3 *Before we elaborate further on the details, we want to clarify two points to avoid potential confusion. First, since approximate algorithms may produce sub-optimal estimates of the posterior (such as ignoring the covariance as in the above example), we do not want to completely rely on the graph-jump step $\mathcal{Q}$ for Markov chain transition. Therefore, we consider a mixture kernel $\mathcal{R}(\theta^t, \cdot)$ with $w < 1$. Second, the graph-jump $\mathcal{Q}$ itself does not have to lead to an ergodic Markov chain (one that could visit every possible state) — rather, we use $\mathcal{Q}$ to form a network of* highways *and allow fast transition from one region to another one far away, while relying on $\mathcal{K}$ as* local roads *to ensure ergodicity.*

In the above, we use the mixture-based variational distribution mainly for illustration purpose. For constructing a graph in general high dimensional problems, there are potentially superior choices such as normalizing flow-based approximation, or a canonical Gibbs sampler — which may suffer from slow mixing but provide a reasonable starting graph nonetheless.

2.2 Choice of Graph for Fast-mixing Random Walk

Given $(\beta^1, \ldots, \beta^m)$, there are multiple ways to form a connected graph G. To begin the thought process, one choice for G is the complete graph, in which every pair of nodes is connected by an edge. However, it is not hard to see that a β^j is likely to have several β^k's in Θ-space neighborhood with small $\|\beta^j - \beta^k\|$; intuitively, the values of $\Pi(\beta^k \mid y)$ of those close-by points tend to dominate over the points far away. As a result, a jump over G would likely correspond to a small change and hence be not ideal.

To favor jumps over large $\|\beta^j - \beta^k\|$ with a simple choice of G, we consider the opposite to a complete graph, and focus on the smallest and connected graph: an undirected spanning tree G containing $(m-1)$ edges. The spanning tree enjoys a nice optimization property, that we can easily find the global minimum of a sum-over-edge loss function. As a result, we can customize the loss to balance between the posterior kernel difference and the jump distance. To be concrete, we use the following minimum spanning tree:

$$\begin{aligned} G &= \underset{\text{all spanning trees } T}{\arg\min} \sum_{(i,j)\in E_T} c_{i,j}, \\ c_{i,j} &= \begin{cases} \kappa/\{1 + \|\beta^i - \beta^j\|\}, & \text{if } |\log \Pi(\beta^i \mid y) - \log \Pi(\beta^j \mid y)| < \kappa, \\ |\log \Pi(\beta^i \mid y) - \log \Pi(\beta^j \mid y)|, & \text{otherwise} \end{cases}, \end{aligned} \tag{3}$$

with $\kappa > 0$ some chosen threshold. The minimum spanning tree can be found via several algorithms (Prim, 1957; Kruskal, 1956). We state Prim's algorithm (Prim, 1957) here to help illustrate an insight. One starts with a singleton node set $V_1 = \{1\}$ and an empty E_T to initialize the tree, and $V_2 = V \setminus V_1$; each time we add a new node $\hat{j}$ associated with

$$(\hat{i}, \hat{j}) = \underset{(i,j): i\in V_1, j\in V_2}{\arg\min} c_{i,j},$$

and add it to E, and move $\hat{j}$ from V_2 to V_1; we repeat until V_2 becomes empty. We can see that this algorithm is *greedy*, in the sense that it finds the locally optimal $c_{i,j}$ in each step; nevertheless, thanks to the M-convexity (Murota, 1998) (a discrete counterpart of continuous convexity) of the minimum spanning tree problem, the greedy algorithm will produce a globally optimal tree.

The equivalence between local and global optimality allows us to gain interesting insight. Each time we add a new node to the graph, (i) if β^i has all $\beta^j : |\log \Pi(\beta^i \mid y) - \log \Pi(\beta^j \mid y)| \geq \kappa$, then we will choose one with the smallest kernel difference in order to improve the acceptance rate; (ii) if there is more than one candidate edge $(i,j) : |\log \Pi(\beta^i \mid y) - \log \Pi(\beta^j \mid y)| < \kappa$, then we will choose the one with the largest distance $\|\beta^i - \beta^j\|$. In another word, the choice of $c_{i,j}$ is based two different levels of priorities: (i) ensuring high acceptance rate, (ii) maximizing parameter space distance if (i) can be met. Note a discontinuity of $c_{i,j}$ does exist when $\log \Pi(\beta_i|y) - \log \Pi(\beta_j|y) = \kappa$, but fixing the discontinuity such as using $|\log \Pi(\beta^i \mid y) - \log \Pi(\beta^j \mid y)|/(1 + \|\beta^i - \beta^j\|)$ in (3) could disrupt the priorities, hence we decide to use (3) as it is.

We will quantify the effect of κ on acceptance rate in Theorem 1. In this article, for simplicity, we choose $\kappa = 1$ and $r = 3$, as they seem adequate to show impressive empirical performance. Nevertheless, one may also consider two extensions that could further improve the mixing performance, although the procedures are more complicated.

First, one could numerically optimize $\kappa > 0$ and $r \in \{1, \ldots, m\}$ to approximately maximize the expected squared jumped distance (ESJD), a measure on the mixing of Markov chain (Gelman et al., 1997; Pasarica and Gelman, 2010). Since at the graph-construction stage, we do not yet have access to Markov chain samples collected from $\mathcal{R}$, we may use approximate samples β to form an empirical estimate for expected squared jumped distance in a random walk on the graph G_κ (a graph parameter varying with κ):

$$\frac{1}{m} \sum_{j=1}^{m} \frac{1}{B_\kappa(j;r)} \sum_{i \in B_\kappa(j;r)} \min \left\{ 1, \frac{\Pi(\beta^i \mid y) |B_\kappa(i;r)|^{-1}}{\Pi(\beta^j \mid y) |B_\kappa(j;r)|^{-1}} \right\} \|\beta^i - \beta^j\|^2,$$

where we use subscript on $B_\kappa(j;r)$, to indicate that the ball varies with the value of κ. The maximization over (κ, r) is non-convex, however, one could obtain local via standard grid search.

Remark 4 *If we choose G as a d-regular graph (instead of a spanning tree) and $r = 1$, we could instead optimize G under degree constraints to directly maximize the empirical ESJD*

$$\frac{1}{m(d+1)} \sum_{(i,j) \in E_G} \left[1 + \min\{ \frac{\Pi(\beta^j \mid y)}{\Pi(\beta^i \mid y)}, \frac{\Pi(\beta^i \mid y)}{\Pi(\beta^j \mid y)} \} \right] \|\beta^i - \beta^j\|^2,$$

although the optimization of d-regular graph is more complex than the one of spanning tree.

Second, instead of focusing on graph choice, one could generalize and focus on optimizing for a random walk transition probability matrix, equivalently to drawing non-uniform $i \in B(j; r = 1)$. To be concrete, consider a given bidirectional and connected graph $\bar{G}$ of m nodes, we want to estimate a transition probability matrix $P \in [0,1]^{m \times m}$ with $P_{i,j}$ the probability of moving from i to j. This matrix satisfies the following constraints:

$$P 1_m = 1_m, \qquad \pi_\beta^{\mathrm{T}} P = \pi_\beta^{\mathrm{T}}, \qquad P_{i,j} = 0 \text{ if } (i \to j) \notin E_{\bar{G}},$$

where π_β is a given target probability vector that we want the random walk to converge to in the marginal distribution ($\pi_\beta^{\mathrm{T}} = \lim_{t \to \infty} \pi_*^{\mathrm{T}} P^t$ for any initial probability vector π_*^{T}). A sensible specification is $\pi_\beta(j) \propto \Pi(\beta^j \mid y)$. The first equality above ensures that P is a valid transition probability matrix, and the second one gives the global balance condition for random walk on $\bar{G}$.

Since the convergence rate of $\pi_0^{\mathrm{T}} P^t$ toward π_β depends on the second largest magnitude of the eigenvalue of P, and its largest eigenvalue 1 corresponds to right eigenvector 1_m and left eigenvector π_β. We can formulate an optimization problem as

$$\hat{P} = \arg\min_P \|P - 1_m \pi_\beta^{\mathrm{T}}\|_2$$

where $P \in [0,1]^{m\times m}$ is subject to the two constraints above, $\|.\|_2$ above is the spectral norm. This is a convex problem that can be solved quickly. Note that when $\bar{G}$ is a complete graph, we would obtain a trivial solution $P = 1_m \pi_\beta^{\mathrm{T}}$, corresponding to $P_{i,j} \propto \Pi(\beta^j \mid y)$ for any i — since under moderate or high dimension, one $\beta^{\hat{j}}$ will likely dominate over all other β^j's in posterior density, the trivial solution will likely always draw node $\hat{j}$ when forming proposal θ^*, which would not be ideal. Therefore, one may want to exclude from $\bar{G}$ those $(i \to j)$ corresponding to short distance $\|\beta^i - \beta^j\|$. Once we obtain $\hat{P}$, we could draw i from $B(j;1)$ with probability $\hat{P}_{j,i}$ {replacing $|B(j,r)|^{-1}$ in (2)}. This extension is inspired by Boyd et al. (2004); nevertheless, the difference is that they focus on the random walk on an undirected graph with $P = P^{\mathrm{T}}$, with $\pi_\beta(i) = 1/m$ as the target. We provide the optimization algorithm and numerical illustration in the appendix.

2.3 Choice of Relaxation Distribution for High-dimensional Posterior

It is known that Metropolis–Hastings algorithms, if employed with a fixed step size for the proposal, suffer from the curse of dimensionality: the acceptance rate decays to zero quickly as dimension p increases. Based on existing study for Gaussian random-walk Metropolis algorithm with target distribution consisting of p independent components (Gelman et al., 1997; Roberts and Rosenthal, 2001), we can estimate that the vanishing speed of acceptance rate under a fixed step size is roughly $O\{\exp(-\tilde{c}p)\}$ for some constant $\tilde{c} > 0$, with detail given in the appendix.

As a result, if we use a continuous $F(\theta^* \mid \beta^i, \theta^t)$ such as multivariate Gaussian in the graph-jump step, our algorithm will also suffer from a fast decay of acceptance rate as p increase. Therefore, we propose a special relaxation distribution F to slow down the decay.

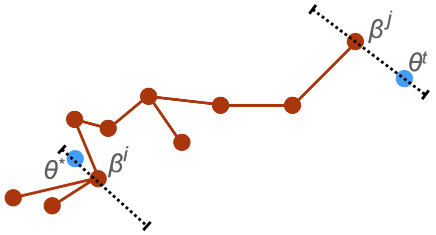


Figure 2: Diagram illustrating the graph jump using uniform relaxation F over a line segment (dashed line in the lower left corner). In the graph jump: (i) θ^t is projected to β^j; (ii) a node i is drawn from the ball $B(j;r)$; (iii) a new proposal is drawn from F, a uniform distribution on the line segment crossing β^i and parallel to the line $\theta^t - \beta^j$; (iv) Metropolis-Hastings adjustment is taken. Each red line represents an edge in the graph.

Figure 2 illustrates the graph jump under a specific F corresponding to uniform relaxation over a line segment. The line segment is parallel to the vector connecting θ^t and its projection β^j. The graph jump now consists of the following operations:

- Find $j = \mathbb{N}(\theta^t)$, and calculate the directional unit-length vector between θ^t and its projection β^j, $v = (\theta^t - \beta^j)/\|\theta^t - \beta^j\|$.
- On the line $\{x : x = \beta^i + \xi v\}$ with $\xi \in \mathbb{R}$, find the maximum-magnitude a^i and b^i, such that

$$\mathbb{N}(\beta^i + \xi v) = i \; \forall \xi \in (a^i, b^i), -l \le a^i \le 0, 0 \le b^i \le l.$$

- Draw θ^* uniformly from $\{x : x = \beta^i + \xi v, a^i < \xi < b^i\}$.
- Accept θ^* as θ^{t+1} with probability $\alpha(\theta^t, \theta^*)$.

It is not hard to see that

$$F(\theta^* \mid \beta^i, \theta^t) = \frac{1}{b^i - a^i} 1\{\|\theta^* - \beta^i\| \le l, \mathbb{N}(\theta^*) = i\},$$

where $l > 0$ is a truncation to ensure properness of F. We can find the line segment easily using one-dimensional bisection. The acceptance rate in (2) becomes:

$$\alpha(\theta^t, \theta^*) = \min\left[1, \frac{\Pi(\theta^* \mid y)|B(i;r)|^{-1}(b_j - a_j)^{-1}}{\Pi(\theta^t \mid y)|B(j;r)|^{-1}(b_i - a_i)^{-1}}\right]. \tag{4}$$

In general, to deal with the curse of dimensionality problem in Metropolis-Hastings algorithms, one often resorts to the Metropolis-Hastings-within-Gibbs strategy, which divides the parameter into blocks and updates each low-dimensional block sequentially. This can be understood as reducing the dimension of each proposed change from θ^t to θ^*.

In our relaxation distribution, although the proposed θ^* is different from θ^t at almost all of its coordinates, as the directional vector v is completely determined by θ^t, only two univariate random variables are drawn when forming θ^*: the choice of i and shift $\xi \in \mathbb{R}$. Therefore, the effective low dimension gives an intuition about how the specific F slows down the decay of acceptance rate. We will formally quantify the scaling of acceptance rate in the theory section.

3. Theoretical Results

In this section, we provide a theoretical exposition of the graph-accelerated algorithm. Compared to the simplicity of the method presented in the previous sections, the results here are more technical and obtained under a few assumptions set up for a tractable mathematical analysis.

3.1 Accelerated Mixing

We now focus on the mixing time of the accelerated algorithm. To provide the necessary background, denote the state space by Θ, and consider a Markov transition kernel $\mathcal{M}$, with $\mathcal{M}(x, \cdot)$ the transition probability measure from state x and $\pi(\cdot)$ the invariant distribution of $\mathcal{M}$. Under the context of posterior estimation, we have $\pi(\cdot)$ equal the posterior distribution associated with kernel $\Pi(\theta \mid y)$. We use $\mathcal{M}^t(x^0, \cdot)$ to denote the distribution after t iterations of transitioning via $\mathcal{M}$ with x^0 an initial point randomly drawn from π^0. Given a small positive number η, the η-mixing time of a Markov chain is:

$$\min\{t : \sup_{A \in \Theta} |\mathcal{M}^t(x^0, A) - \pi(A)| \le \eta\}.$$

Therefore, at a given η, the η-mixing time would be dependent on π^0 and $\mathcal{M}$. Since the left-hand side of the inequality is often intractable, one often derives an upper bound of the left-hand side as a diminishing function of t, and produces an upper bound estimate of the mixing time.

Now we review an important concept of *conductance*, which is useful for calculating the above upper bound. Consider an ergodic flow

$$\Phi_{\mathcal{M}}(A) = \int_{x \in A} \mathcal{M}(x, A^c)\, \pi(dx),$$

as the amount of total flow from A to $A^c = \Theta \setminus A$. The conductance of $\mathcal{M}$ is a measurement of the bottleneck flow adjusted by the volume:

$$\psi^*_{\mathcal{M}} := \inf_{A \subset \Theta, \pi(A) < 1/2} \frac{\Phi_{\mathcal{M}}(A)}{\pi(A)}.$$

The corrollary 3.3 of Lovász and Simonovits (1992) states that $\sup_{A \in \Theta} |\mathcal{M}^t(x^0, A) - \pi(A)| \leq \sqrt{M}\{1 - (\psi^*_{\mathcal{M}})^2/2\}^t$, with $M = \sup_{A \subset \Theta} \pi^0(A)/\pi(A)$.

Therefore, when comparing two Markov chains, a large conductance $\psi^*_{\mathcal{M}} > \psi^*_{\mathcal{M}'}$ means that $\mathcal{M}$ has a faster-diminishing upper-bound rate on the total variation distance, hence a smaller upper-bound estimate on the mixing time, when compared with $\mathcal{M}'$. Although this is not a direct comparison between two mixing times, it offers theoretical insights into why one algorithm empirically shows a faster mixing of Markov chains than the other.

We now focus on the Markov chain generated by the baseline $\mathcal{K}(\theta^t, \cdot)$. For a sufficiently small $\epsilon > 0$, we define an ϵ-expansion from the infimum

$$\mathcal{A}^*_\epsilon(\mathcal{K}) := \left\{ A \subset \Theta \;\middle|\; \frac{\Phi_{\mathcal{K}}(A)}{\pi(A)} < \psi^*_{\mathcal{K}} + \epsilon, \pi(A) < \frac{1}{2} \right\}.$$

We consider the graph-accelerated Markov chain with $\mathcal{R} = w\mathcal{Q} + (1 - w)\mathcal{K}$ where the transition kernel $\mathcal{Q}$ has the same invariant distribution π. We have the following guarantee.

Theorem 5 *If there exists $\epsilon > 0$ such that $\Phi_{\mathcal{Q}}(A) > \Phi_{\mathcal{K}}(A)\ \forall A \in \mathcal{A}^*_\epsilon(\mathcal{K})$, then there exists $w \in (0, 1]$ such that $\psi^*_{\mathcal{R}} > \psi^*_{\mathcal{K}}$.*

The above result shows that $\mathcal{Q}$ only needs to improve the ergodic flow on $\mathcal{A}^*_\epsilon(\mathcal{K})$. This means that as long as $\mathcal{Q}$ improves the flow in $\mathcal{A}^*_\epsilon(\mathcal{K})$, the mixture kernel $\mathcal{R}$ will have potential acceleration. We provide further discussion in the appendix.

3.2 Scaling of Acceptance Rate in High Dimension

For the specific relaxation distribution introduced in Section 2.3, we give a theoretical characterization of the acceptance rate in terms of its rate of change as p grows. For now, we treat the approximate sample size m as a sufficiently large number that satisfies the two assumptions below, and we will discuss the associated requirement on m later.

Our goal is to obtain a lower bound on the expected acceptance rate $\mathbb{E}_{\theta^t \sim \Pi(\theta|y)} \alpha(\theta^t, \theta^*)$. Since $\mathbb{E}_{\theta^t \sim \Pi(\theta|y)} \alpha(\theta^t, \theta^*) \geq \mathbb{E}_{\theta^t \sim \Pi(\theta|y)} 1(\theta^t \in \mathcal{B}) \alpha(\theta^t, \theta^*)$ for $\mathcal{B} \subset \Theta$. We now find a $\mathcal{B}$ that could yield a tractable bound.

We first exclude those $\beta^j : \min_{i\in B(j,r)\setminus j} |\log \Pi(\beta^i \mid y) - \log \Pi(\beta^j \mid y)| > \kappa$ with κ the chosen constant in (3). For the remaining β^j's, we can form an δ-covering, denoted by $\tilde{B}_1$. That is, for any $x \in \tilde{B}_1$, $\min_j \|x - \beta^j\|_2 \le \delta$. We choose $\delta = c_2 p^{-c_3}$ with $c_2 > 0$ and $-\infty < c_3 < 1/2$. We consider the following assumptions:

- (A1) There exists set $\tilde{B}_2$ and p-independent constants $c_1 > 0$ and $\gamma \ge 0$ such that for any $(\theta, \theta') \in \tilde{B}_2 \times \tilde{B}_2$,

$$|\log \Pi(\theta \mid y) - \log \Pi(\theta' \mid y)| \le c_1 p^{\gamma} \|\theta - \theta'\|,$$

  where $\|\cdot\|$ denotes some norm.

- (A2) $\mathcal{B} = \tilde{B}_1 \cap \tilde{B}_2$ has posterior probability $\int_{\mathcal{B}} \Pi(\theta \mid y) d\theta = \mu_{\mathcal{B}}$ bounded away from zero.

- (A3) The interval length ratio satisfies $(b_i - a_i)/(b_j - a_j) < c_4$ for all $(i, j) : i \in B(j; r)$. Ball size satisfies $|B(j; r)| \le c_5$ for all j. Truncation satisfies $l \le \delta$.

In the above, (A1) is commonly referred to as a $(c_1 p^{\gamma})$-smoothness condition (Bubeck, 2015) if one uses Euclidean norm, and recently considered by Tang and Yang (2024) in the study of Metropolis-Adjusted Langevin algorithms. The difference here is that we only impose this condition on a subset $\tilde{B}_2 \subset \Theta$, hence the condition is relatively easy to satisfy.

Theorem 6 *Under (A1-A3), the expected acceptance rate* (4) *has the following bound*

$$\mathbb{E}_{\theta^t \sim \Pi(\theta|y)} \alpha(\theta^t, \theta^*) > \frac{\mu_{\mathcal{B}}}{c_4 c_5} e^{-\kappa} \exp\{-2c_1 p^{(\gamma - c_3)}\}.$$

Remark 7 *With suitable γ and $c_3 < 1/2$, we have the acceptance rate vanishing at a rate slower than $O\{\exp(-\tilde{c}p)\}$, as seen in algorithms such as Gaussian random-walk Metropolis.*

We now discuss the required size m on the approximate samples. Obviously, the larger m, the larger area $\tilde{B}_1$ and $\mu_{\mathcal{B}}$ will be. To more precisely characterize its dependency on p, and suggest choice for c_3, we think of a high posterior probability polytope $\mathcal{P} = (\theta : \theta = k_0 \Sigma_0^{1/2} x + a_0, \|x\|_1 \le 1)$ with for some $a_0 \in \Theta$, Σ_0 positive definite, and some fixed and dimension-independent $k_0 > 0$ so that $\mu_{\mathcal{P}} = \int_{\mathcal{P}} \Pi(\theta \mid y) d\theta \gg 0$. That is, we want $\tilde{B}_1 \supseteq \mathcal{P}$. Assuming the approximate sampler can generate points in $\mathcal{P}$, the key question is how many balls $(x : \|x - \theta_j\| \le \delta)$ are needed for covering $\mathcal{P}$?

The answer depends on the type of norm used in $\|x - \theta_j\|$. In the following, we consider using $\|x - \theta_j\|_{\Sigma_0} = \sqrt{(x - \theta_j)^T \Sigma_0^{-1} (x - \theta_j)}$, which simplifies the problem to covering a unit $L1$-ball using small $L2$-balls. The celebrated Maurey's empirical method (Pisier, 1999) shows that, to cover a unit $L1$-ball, we only need at least $m = (2p + 1)^{O(1/\delta_0^2)}$-many δ_0-$L2$-balls with radius δ_0, provided that $\delta_0 > p^{-1/2}$. With appropriate scaling, we reach the choice of $\delta = c_2 p^{-c_3}$, with $c_3 < 1/2$. Substituting into the lower bound of m, we see that $m = (2p + 1)^{O(p^{2c_3})}$. In addition, we note that if $\mathcal{P}$ were an ellipsoid, then there would be a curse of dimensionality in the covering number (using δ_0-radius $L2$-ball) $m = O(1/\delta_0)^p$ (Vershynin, 2015).

Therefore, setting $c_3 = 0$ yields our suggestion of $m = O(p)$, which balances between controlling acceptance rate decay and preventing excessive demand on the number of approximate samples.

Remark 8 *Before concluding this section, we provide practical guidance on choosing the approximation sample size m and graph jump probability w. For the sample size m, when using a reasonably good approximate sampler, we find that $m = 100$ works effectively for low-dimensional problems where $p \leq 100$. As dimensionality increases beyond this, we recommend setting $m = O(p)$. For the jump probability w, while our proof of Theorem 2 provides theoretical choices based on conductance constants, these are typically intractable in practice. We therefore suggest using $w = 0.5$ as a default value, which corresponds to alternating between the baseline and graph jump kernels.*

4. Simulations

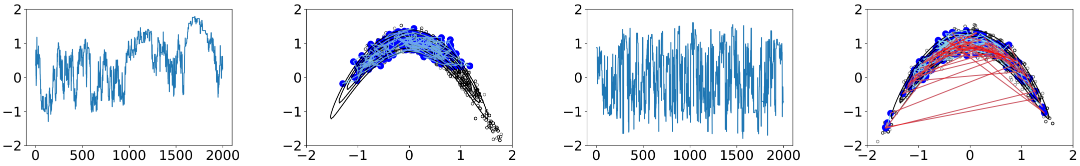
In this section, we provide numerical evidence that the graph-accelerated algorithm works well in both low and high dimensional settings.

4.1 Sampling Posterior with Non-convex Density Contour

For sampling low-dimensional $\Pi(\theta \mid y)$, the random-walk Metropolis algorithm is appealing due to its low computational cost. For low dimensional problems, a common choice for random walk proposal is $\mathrm{N}(\cdot; \theta^t, s\, I)$, with $s > 0$ the step size. A potential issue is that when the high posterior density region is not close to a convex shape, the step size s would have to be small, leading to computing inefficiency. The following example is often used as a challenging case (Haario et al., 1999), with likelihood and prior

$$y_i \overset{iid}{\sim} \mathrm{N}(\theta_1^2 + \theta_2, 1^2), \text{ for } i = 1, \ldots, n, \qquad \theta \sim \mathrm{N}(0, I_2).$$

If the true parameters are chosen subject to the constraint $\theta_1^2 + \theta_2 = 1$, the posterior distribution of (θ_1, θ_2) would spread around the banana-shaped curve $\{(\theta_1, \theta_2) : \theta_1^2 + \theta_2 = 1\}$.



(a) Traceplot of θ_1 using random-walk Metropolis.

(b) Markov chain sample of (θ_1, θ_2) from the random-walk Metropolis. The first 400 sample points and traces are shown in blue.

(c) Traceplot of θ_1 using the accelerated algorithm.

(d) Markov chain sample of (θ_1, θ_2) from the accelerated algorithm. The first 400 sample points and traces are shown in blue, with successful graph jump steps shown in red.

Figure 3: Graph-accelerated random-walk Metropolis for sampling a posterior distribution of banana shape.

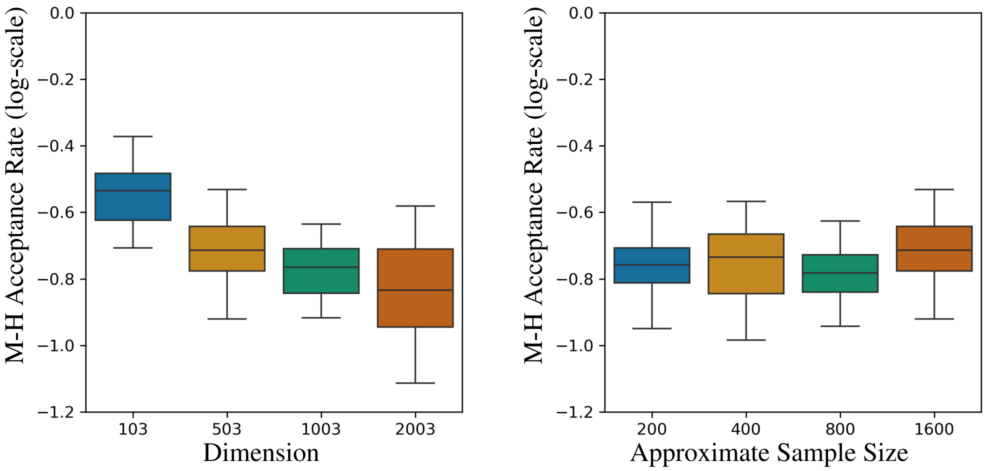
Using random-walk Metropolis as the baseline algorithm, we tweak s to around 0.5 so that the Metropolis acceptance rate is around 0.234. We run the algorithm for 3000 iterations and use the last 2000 as a Markov chain sample. Figure 3(a)(b) shows that it takes a long time for the sampler to move from one end to the other.

For acceleration, we first obtain 100 approximate samples from a variational method based on a 10-component Gaussian mixture $\sum_{k=1}^{10} \tilde{w}_k \mathrm{N}(\tilde{\mu}_k, I\tilde{\sigma}^2)$, then we run the accelerated algorithm. As shown in Figure 3, the accelerated algorithm jumps rapidly between the two ends, leading to improved mixing performance. The effective sample size per iteration for θ_1 from the baseline algorithm is 0.16%, while the one for the accelerated version is 6.1%.

4.2 Numerical Results on the Change of Acceptance Rate

In Section 3, we gave a lower-bound quantification of the Metropolis-Hastings acceptance rate under a theoretical setting with increasing dimensions. To show empirical evidence that the acceptance rate remains positive and away from zero in practical settings, we conduct simulations under different dimensions p and approximate sample sizes m.

To show empirical evidence that the acceptance rate remains positive and away from zero in practice, we adopt the latent Gaussian model used in the application (to be presented in Section 5), but now fit the model to simulated data of different sample size $n \in \{100, 500, 1000, 2000\}$. We use $\tau = 1$, $h = 0.25$, and $r = 2$ during data simulation. Since each data point y_i is associated with a latent z_i, the effective dimension of variables to sample is $p = (n + 3)$. We run the Gibbs sampling algorithm (as the baseline algorithm described in the main text) for 2000 iterations, with first 400 discarded as burn-in, then take a subset of size m as the approximate samples. In each experiment, we run the accelerated algorithm for 2000 iterations with $w = 0.5$ and $r = 3$, and report the empirical acceptance rate as the number of accepted graph jump steps divided by 1000, as equal to 2000×0.5.



(a) Acceptance rates (at $\log_{10}$ scale) versus different dimensions p, under fixed $m = 1600$.

(b) Acceptance rates (at $\log_{10}$ scale) versus different approximate sample sizes m, under fixed $p = 503$.

Figure 4: Numerical result of the Metropolis-Hastings acceptance rate in the graph jump step. The acceptance rates are calculated from simulated experiments of posterior sampling from a latent Gaussian model.

We first conduct experiments under $m = 1600$ and different $p \in \{103, 503, 1003, 2003\}$. We repeat the experiments for 20 times under each value of p. As shown in the boxplots in Figure 4(a), the decrease of the Metropolis-Hastings acceptance rate over p is slow: on average, the acceptance rate is around 26% when $p = 103$, and around 17% when $p = 2003$. Next, we conduct experiments under $p = 503$ with different sizes of approximate samples $m \in \{200, 400, 800, 1600\}$. The Prim's algorithm for finding the minimum spanning tree scales efficiently in $O(m^2)$, and it takes about 2 seconds to find the minimum spanning tree for $m = 1600$ on a laptop with Apple Silicon processor. As shown in the boxplots in Figure 4(b), the average acceptance rates show no clear difference under different m in the range.

5. Application: Estimating Latent Gaussian Model for Power Outage Data

To show the performance of our algorithm in a relatively large dimension, we experiment with a latent Gaussian model for count data. We use the power outage count for a zip code area in the south of Florida collected during a 90-day time period in the 2009 hurricane season. There are $n = 513$ records of outage counts $y_i \in \mathbb{Z}_{\geq 0}$, reported at irregularly-spaced time points. We rescale the time records to be in $[0, 1]$, and denote the transformed time by t_i.

For modeling count data, it is canonical to consider a generalized linear model with a count distribution (typically, Poisson or negative binomial) as the stochastic component. We choose negative binomial due to its tractable form in the Gibbs sampling algorithm with Pólya-Gamma augmentation (Polson et al., 2013). We use the following likelihood with latent Gaussian covariance $\Sigma_{i,j}(\tau, h) = \tau \exp[-(t_i - t_j)^2/2h]$, leading to likelihood:

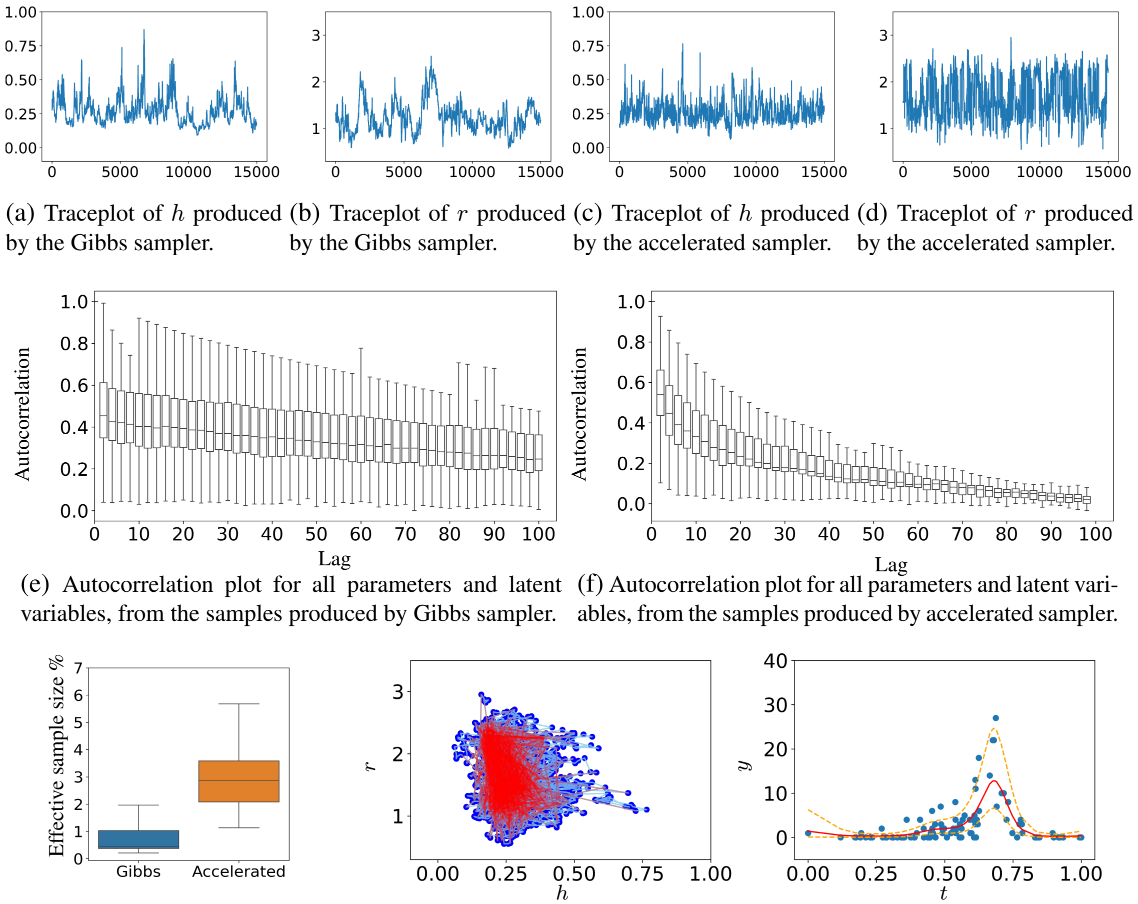
$$L(y, z \mid \tau, h) = (2\pi)^{-n/2} |\Sigma(\tau, h)|^{-1/2} \exp\left[-\frac{1}{2} z^{\mathrm{T}} \{\Sigma(\tau, h)\}^{-1} z \right] \prod_{i=1}^{n} \frac{\exp(r z_i)}{\{1 + \exp(z_i)\}^{r + y_i}}.$$

In prior specification, we use $h \sim$ Inverse-Gamma$(2, 1)$ for the bandwidth, $r \sim \mathrm{N}_{(0,\infty)}(0, 1)$ half-Gaussian for the inverse dispersion parameter, $\tau \sim$ Inverse-Gamma$(2, 1)$ for the scale.

We first describe the baseline algorithm for posterior sampling. Using Pólya-Gamma latent variable ω_i (Polson et al., 2013), we have augmented likelihood

$$\frac{\{\exp(z_i)\}^r}{\{1 + \exp(z_i)\}^{r + y_i}} = 2^{-(r + y_i)} \exp\{(\frac{r - y_i}{2}) z_i\} \int \exp(-\omega_i z_i^2/2) \mathrm{PG}(\omega_i \mid r + y_i, 0) \mathrm{d}\omega_i.$$

where PG$(\cdot \mid r + y_i, 0)$ is the Pólya-Gamma distribution, and we refer the readers to Polson et al. (2013) for its mathematical definition, and Abril-Pla et al. (2023) for its sampler implementation. We have closed-form updates for most of the latent variables and parameters, $\omega_i \sim$ PG$(r + y_i, z_i)$ for $i = 1, \ldots, n$, $z \sim \mathrm{N}[\{\Sigma^{-1} + \mathrm{diag}(\omega_i)\}^{-1}\{(r - y)/2\}, \{\Sigma^{-1} + \mathrm{diag}(\omega_i)\}^{-1}]$ and $\tau \sim$ Inverse-Gamma$\{n/2 + 2, z^{\mathrm{T}} \tilde{\Sigma}^{-1}(h) z/2 + 1\}$ with $\tilde{\Sigma}_{i,j}(h) = \exp[-(t_i - t_j)^2/2h]$. On the other hand, since h and r do not have full conditional distribution available in closed form, we use softplus reparameterization $h = \log\{1 + \exp(\tilde{h})\}$ and $r = \log\{1 + \exp(\tilde{r})\}$ and use random-walk Metropolis algorithm with proposal $\mathrm{N}\{\cdot; (\tilde{h}, \tilde{r})^t, Is\}$ to obtain an update on $(\tilde{h}, \tilde{r}) \in \mathbb{R}^2$, then transform to (h, r). In the random-walk Metropolis algorithm, we use the posterior with (τ, ω) integrated out, and tweak s so that the acceptance rate is around 0.234. We run the baseline algorithm for 20,000 iterations, and treat the first 5,000 as burn-in. As shown in Figure 5(a)(b) and (e), the baseline Gibbs sampling algorithm suffers from critically slow mixing. Even at the 100-th lag, most of the parameters and latent variables still show autocorrelations near 40%.



(a) Traceplot of h produced by the Gibbs sampler.

(b) Traceplot of r produced by the Gibbs sampler.

(c) Traceplot of h produced by the accelerated sampler.

(d) Traceplot of r produced by the accelerated sampler.

(e) Autocorrelation plot for all parameters and latent variables, from the samples produced by Gibbs sampler.

(f) Autocorrelation plot for all parameters and latent variables, from the samples produced by accelerated sampler.

(g) Boxplots showing effective sample sizes per iteration for two algorithms.

(h) Markov chain sample of (h, τ) from the accelerated algorithm, with successful graph jumps shown in red.

(i) Posterior estimate of the mean curve using the accelerated algorithm. The point-wise 95% credible band is shown in dashed lines.

Figure 5: Sampling a posterior distribution of a latent Gaussian model for count data.

For the acceleration algorithm, we obtain approximate samples of $\beta^j = (z, \tau, h, r)^j \in \mathbb{R}^{103}$ by simply taking the first $1,000$ Markov chain samples after the burn-in period from the slow-mixing Gibbs sampler, then we construct the graph and run the accelerated algorithm for 20,000 iterations (with the first 5,000 as burn-in). Despite the relatively large dimension, the graph jump steps had 18.4% of success rate {red lines in Figure 5(h)}. As shown in Figure 5(c)(d) and (f), the accelerated algorithm leads to much-improved mixing performance. Almost all parameters and latent variables have autocorrelations reduce to small values after the 60-th lag. We plot the posterior mean curve $r\exp(-z_i)$, and the point-wise 95% credible band in Figure 5(i).

Since both algorithms target the same posterior distribution, we expect them to yield almost identical inference results when collecting equal numbers of effective samples (through running and thinning the Markov chains). According to the effective sample size comparison in Figure 5(g), the baseline Gibbs sampler requires approximately 3 times as many iterations as the accelerated sampler to obtain the same number of effective samples.

6. Discussion

In our accelerated algorithm, we treat the graph as a fixed object. An interesting extension to explore is to allow the graph to keep growing, by adding some samples collected from the Markov chain. This idea is especially appealing in the sense that a *chain graph* is in fact a special tree graph without branches, which suggests opportunities to develop new algorithms such as *Markov tree Monte Carlo*. On the other hand, a critical issue is that the growing of tree graph would break the detailed balance condition, hence risking a failure of convergence to the target posterior distribution. One possible solution is to employ the well-known diminishing adaptation strategy (Roberts and Rosenthal, 2009), by making the differences between proposal kernels vanish as the iteration increases, or simply reducing the frequency of graph-updating toward zero as the iteration increases (Chimisov et al., 2018). Another possibility is to explore algorithms that do not require the detailed balance condition, such as those non-reversible algorithms (Sohl-Dickstein et al., 2014; Bierkens, 2016) and unadjusted diffusion algorithms (Durmus and Moulines, 2019; Dalalyan, 2017).

For growing the graph, one could consider alternative graph structures that are more computationally efficient to learn than those presented earlier in this article. A promising choice is the approximate nearest neighbor graph, which can be constructed in an online fashion using the Hierarchical Navigable Small World (HNSW) algorithm (Malkov and Yashunin, 2018). This algorithm has an insertion complexity of only $O(\log m)$, making it particularly suitable for incrementally growing graphs.

The accelerated algorithm described in this article can be generalized to the posterior sampling of discrete or combinatorial parameters. Nevertheless, choosing a relaxation distribution in high-dimensional discrete space can be challenging. This issue could be potentially circumvented using continuous embedding as considered by several diffusion-based algorithms (Pakman and Paninski, 2013; Nishimura et al., 2020).

For parameter on an assumed known low-dimensional manifold, one could use geodesic distance instead of the Euclidean distance. On the other hand, extra care is needed for handling issues such as the possible non-uniqueness of projection (which could affect the algorithm in Figure 2) and potentially expensive computation of geodesic distance. Furthermore, when the manifold is unknown, the posterior estimation task becomes more challenging and warrants future work.

Acknowledgement

The authors would like to thank Cheng Zeng and Gareth Roberts for helpful discussions. The authors would also like to thank the editor, action editor and anonymous reviewers for their helpful comments and suggestions. The research of Leo Duan is partially supported by the U.S. National Science Foundation under grant NSF-DMS 2319551. The research of Anirban Bhattacharya is partially supported by the U.S. National Science Foundation under grant NSF-DMS 2210689.

Appendix A. Proof of Theorems

A.1 Proof of Theorem 1

Proof To verify the detailed balance, it suffices to check the case when $\theta^{t+1} = \theta^*$. For any θ^t and θ^*,

$$\begin{aligned}
&\Pi(\theta^t \mid y)\mathcal{Q}(\theta^t, \theta^*) = \sum_{i \in B\{\mathbb{N}(\theta^t); r\}} \Pi(\theta^t \mid y)|B(j;r)|^{-1} F(\theta^* \mid \beta^i, \theta^t)\alpha(\theta^t, \theta^*) \\
&= \sum_{i \in B(j;r)} 1\big[\mathbb{N}(\theta^*) = i\big] \\
&\qquad \times \min\Big[\Pi(\theta^* \mid y)|B(i;r)|^{-1} F(\theta^t \mid \beta^j, \theta^*), \Pi(\theta^t \mid y)|B(j;r)|^{-1} F(\theta^* \mid \beta^i, \theta^t)\Big] \\
&\overset{(a)}{=} 1\big[\mathbb{N}(\theta^*) = i\big] 1\big[\mathbb{N}(\theta^t) = j\big] \\
&\qquad \times \min\Big[\Pi(\theta^* \mid y)|B(i;r)|^{-1} F(\theta^t \mid \beta^j, \theta^*), \Pi(\theta^t \mid y)|B(j;r)|^{-1} F(\theta^* \mid \beta^i, \theta^t)\Big]
\end{aligned}$$

where (a) uses the almost sure uniqueness of projection, so that there is only one $i : 1[\mathbb{N}(\theta^*) = i] \neq 0$ at given θ^*, and the fact that $1[\mathbb{N}(\theta^t) = j] = 1$. Clearly, the last line is symmetric in (θ^t, θ^*), hence $\Pi(\theta^t \mid y)\mathcal{Q}(\theta^t, \theta^*) = \Pi(\theta^* \mid y)\mathcal{Q}(\theta^*, \theta^t)$. ■

A.2 Proof of Theorem 2

Proof We consider the conductance under two cases:

1) Transitioning from $A \in \mathcal{A}^*_\epsilon(\mathcal{K})$:

For any $A \in \mathcal{A}^*_\epsilon(\mathcal{K})$, we have for any $w \in (0, 1]$:

$$\frac{\Phi_{\mathcal{R}}(A)}{\pi(A)} = \frac{w\Phi_{\mathcal{Q}}(A) + (1 - w)\Phi_{\mathcal{K}}(A)}{\pi(A)} > \frac{\Phi_{\mathcal{K}}(A)}{\pi(A)} \geq \psi^*_{\mathcal{K}}. \tag{5}$$

2) Transitioning from $B \in \Theta \setminus \mathcal{A}^*_\epsilon(\mathcal{K})$:

For any $B \in \mathcal{B} = \{A \in \Theta : \pi(A) < 1/2, A \notin \mathcal{A}^*_\epsilon(\mathcal{K})\}$, we have $\Phi_{\mathcal{K}}(B)/\pi(B) \geq \psi^*_{\mathcal{K}} + \epsilon$.

Let $m_{\mathcal{B}} := \inf_{B \in \mathcal{B}} \Phi_{\mathcal{Q}}(B)/\Phi_{\mathcal{K}}(B) \geq 0$, and for any $w \in (0, 1]$ such that:

$$w(1 - m_{\mathcal{B}}) < \frac{\epsilon}{\psi^*_{\mathcal{K}} + \epsilon}, \tag{6}$$

we have

$$\begin{aligned}
\frac{\Phi_{\mathcal{R}}(B)}{\pi(B)} &= \frac{w\Phi_{\mathcal{Q}}(B) + (1 - w)\Phi_{\mathcal{K}}(B)}{\pi(B)} = \{w\Phi_{\mathcal{Q}}(B)/\Phi_{\mathcal{K}}(B) + (1 - w)\}\frac{\Phi_{\mathcal{K}}(B)}{\pi(B)} \\
&\geq \{wm_{\mathcal{B}} + (1 - w)\}\frac{\Phi_{\mathcal{K}}(B)}{\pi(B)} > \frac{\psi^*_{\mathcal{K}}\Phi_{\mathcal{K}}(B)}{(\psi^*_{\mathcal{K}} + \epsilon)\pi(B)} \geq \psi^*_{\mathcal{K}}.
\end{aligned} \tag{7}$$

To show that such a w always exists, as well as choosing a large value for w: when $m_{\mathcal{B}} \geq 1$, we can choose $w = 1$; when $m_{\mathcal{B}} < 1$, we can choose $w = (1 - m_{\mathcal{B}})^{-1}\epsilon/(\psi^*_{\mathcal{K}} + \epsilon) - \eta$, with $\eta > 0$ sufficiently small so that $w > 0$.

Combining 1) and 2), we see that there exists $w \in (0, 1]$, such that $\psi^*_{\mathcal{R}} > \psi^*_{\mathcal{K}}$. ■

A.3 Proof of Theorem 3

Proof The acceptance rate under our specified $F(\theta^* \mid \beta^i, \theta^t)$ is

$$\alpha(\theta^t, \theta^*) = \min\left\{1, \frac{\Pi(\theta^* \mid y)|B(i;r)|^{-1}(b_j - a_j)^{-1}}{\Pi(\theta^t \mid y)|B(j;r)|^{-1}(b_i - a_i)^{-1}}\right\}.$$

We see that $\min_j \|\theta^* - \beta^j\| \le \delta$ by the way we generate θ^*, hence $\theta^* \in \mathcal{B}$. Consider any $\theta^t \in \mathcal{B}$,

$$\begin{aligned}
&\log \Pi(\theta^* \mid y) - \log \Pi(\theta^t \mid y)\\
&\ge \log \Pi(\theta^* \mid y) - \log \Pi(\beta^i \mid y) + \log \Pi(\beta^j \mid y) - \log \Pi(\theta^t \mid y) - |\log \Pi(\beta^i \mid y) - \log \Pi(\beta^j \mid y)|\\
&\ge -c_1 p^\gamma(\|\theta^* - \beta^i\| + \|\theta^t - \beta^j\|) - \kappa\\
&\ge -2c_1 p^\gamma \delta - \kappa.
\end{aligned}$$

Since we know $B(j;r) \ge 1$, we have $|B(j;r)|/|B(i;r)| \le c_5$. Including the bound ratio $(b_i - a_i)/(b_j - a_j) < c_4$, and taking expectation over $\theta^t \sim \Pi(\theta \mid y)$ yields the result. ■

A.4 Additional Remarks on the Theory Results

On Theorem 5, the result is qualitative because we are limited to comparing two upper bounds of mixing time. Nevertheless, the result formalizes our comment in Remark 3 — we do not need $\mathcal{Q}$ alone to form a fast-mixing Markov chain. As an intuitive example, for sampling a k-modal distribution via the mixture kernel $\mathcal{K}$, including jumps over a barebone graph with only k nodes (each located near a unique mode) as $\mathcal{Q}$ will help improve the mixing of Markov chains.

On Theorem 6, we can obtain $\gamma \le 1/2$ for many commonly seen posterior densities. For example, for $\log \Pi(\theta \mid y) = -\theta' A\theta + o(\|\theta\|_2^2)$ with positive definite A, we can find a $\tilde{B}_2$ inside the ball $\{\theta : \|\theta\|_1 \le a_1\sqrt{p}\}$ which implies $\|\theta\|_2 \le \|\theta\|_1 \le a_1\sqrt{p}$. In that case, we have $\gamma = 1/2$ for the Euclidean norm.

On the covering number, we focus on a general high-dimensional setting with a high posterior probability set $\mathcal{P}$. On the other hand, in special but often encountered cases where the high posterior probability set can be found as a δ-neighborhood of a $\tilde{p}$-dimensional polytope (with $\tilde{p} \ll p$, such as in sparse regression where most elements of θ are close to 0), we can change the above paragraph to be based on a $\tilde{p}$-dimensional $L1$-ball. A further reduction of m could be possible under additional assumptions on the $\tilde{p}$-dimensional polytope.

Appendix B. Optimization Algorithm for Further Improvement on Graph Choice

We provide the details on the optimization of a random walk transition probability matrix P. One solution is using the dual ascent algorithm. The minimization of spectral norm is equivalent to:

$$\begin{aligned}
&\min_{P,s}\ s\\
\text{subject to } &\|P - 1_m \pi_\beta^{\mathrm{T}}\|_2 \le s,\ s \ge 0\\
&P1_m = 1_m, \qquad \pi_\beta^{\mathrm{T}} P = \pi_\beta^{\mathrm{T}},\\
&P_{i,j} = 0 \text{ if } (i \to j) \notin E_{\bar{G}}, \qquad P_{i,j} \ge 0
\end{aligned}$$

Using semi-definite programming, we can set up the Lagrangian:

$$\begin{aligned}\mathcal{L}(P, Z, s, u, v, Y, \lambda) =& s - \mathrm{tr}\left\{ \begin{bmatrix} Z_{11} & Z_{12} \\ Z_{12}^{\mathrm{T}} & Z_{22} \end{bmatrix} \begin{bmatrix} sI & (P - 1_m \pi_\beta^{\mathrm{T}}) \\ (P - 1_m \pi_\beta^{\mathrm{T}})^{\mathrm{T}} & sI \end{bmatrix} \right\} \\ & + u^{\mathrm{T}}(P 1_m - 1_m) + (\pi_\beta^{\mathrm{T}} P - \pi_\beta^{\mathrm{T}}) v - \mathrm{tr}(YP) - \lambda s.\end{aligned}$$

where $Z \succeq 0$ is a four-block positive semi-definite matrix, $u \in \mathbb{R}^p$, $v \in \mathbb{R}^p$, $\lambda \geq 0$, lastly, $Y \in \mathbb{R}^{p \times p}$, except $Y_{i,j} \geq 0$ if $(i \to j) \in E_{\bar{G}}$. Clearly, the Lagrangian dual $\inf_{P,s} \mathcal{L}(\cdot)$ would be $-\infty$, unless:

$$\begin{aligned}& -2Z_{12}^{\mathrm{T}} + 1_m u^{\mathrm{T}} + v \pi_\beta^{\mathrm{T}} - Y = 0, \\ & 1 - \mathrm{tr}(Z) - \lambda = 0,\end{aligned}$$

which are equivalent to dual feasibility conditions:

$$\begin{aligned}& Z_{12}(j, i) \leq \frac{u_j + v_i \pi_\beta(j)}{2} \text{ if } (i \to j) \in E_{\bar{G}}, \\ & \mathrm{tr}(Z) \leq 1, \; Z \succeq 0\end{aligned}$$

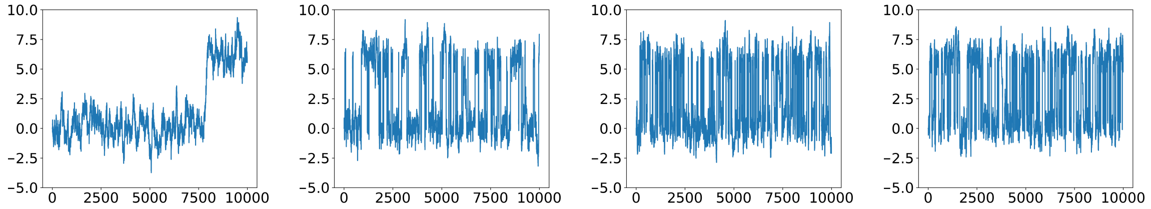
for the dual problem:

$$\sup_{Z,u,v} 2\mathrm{tr}\{Z_{12}^{\mathrm{T}}(1_m \pi_\beta^{\mathrm{T}})\} - u^{\mathrm{T}} 1_m - \pi_\beta^{\mathrm{T}} v.$$

We parameterize $Z = \tilde{Z}\tilde{Z}^{\mathrm{T}}$, with $\tilde{Z} \in \mathbb{R}^{p \times p}$, and use log-barrier to enforce inequalities, then use gradient ascent algorithm {via the JAX package (Bradbury et al., 2018)} to find out $\hat{Z}, \hat{u}, \hat{v}$. Then using complementary slackness condition $s \cdot \mathrm{tr}(Z_{11} + Z_{22}) + 2\mathrm{tr}\{Z_{12}^{\mathrm{T}}(P - 1_m \pi_\beta^{\mathrm{T}})\} = 0$ and primal optimal condition $s = \|P - 1_m \pi_\beta^{\mathrm{T}}\|_2$, we can find out the value of $\hat{P}$.

Appendix C. Numerical Illustration on Different Choices of Graph

For numerical illustration, we use the Gaussian mixture example we consider in the main text. In addition to (i) the baseline random walk Metropolis and (ii) the accelerated algorithm with spanning tree graph under default value ($r = 1, \kappa = 1$), we experiment with (iii) the accelerated algorithm with greedy optimization on (r, κ) to maximize the expected squared jumped distance (with $r = 5$ and $\kappa = 0.65$), and (iv) the accelerated algorithm using optimized random walk (with edges excluded if $\|\beta^i - \beta^j\| \leq 0.5$). For (ii)(iii) and (iv), we use the same collection of $m = 100$ samples, and we compare the mixing performance of those algorithm via the traceplot of θ_2 in Figure 6. As can be seen, (iii) and (iv) further improve the mixing compared to (ii), although these two extensions are much more complicated.



(a) Traceplot of θ_2 using random-walk Metropolis.

(b) Traceplot of θ_2 using accelerated algorithm with spanning tree graph with $r = 1$, $\kappa = 1$.

(c) Traceplot of θ_2 using acceleration via spanning tree graph with greedily optimized (r, κ).

(d) Traceplot of θ_2 using acceleration via optimized random walk graph.

Figure 6: Comparing acceleration algorithms using different graphs, for sampling a two-component Gaussian mixture distribution.

Appendix D. Estimate on the Vanishing Rate of Acceptance Probability of Gaussian Random-walk Metropolis algorithm

It has been shown (Gelman et al., 1997; Roberts and Rosenthal, 2001) that for Gaussian random-walk Metropolis algorithm with target distribution consisting of p independent components, one may use a Gaussian proposal with standard deviation at $cp^{-1/2}$, so that the acceptance rate could stay above zero and converge to $2\Phi(-\tilde{m}c)$ as $p \to \infty$, with some $\tilde{m} > 0$ depending on the target distribution and Φ the standard Gaussian cumulative distribution function.

To roughly estimate the vanishing speed of the acceptance rate under a fixed step size, we can replace c by $cp^{1/2}$ and obtain $2\Phi(-\tilde{m}cp^{1/2})$. For $x > 0$ and $t > x$,

$$\begin{aligned}\Phi(-x) &= (2\pi)^{-1/2}\int_x^\infty \exp(-t^2/2)\mathrm{d}t\\ &\le (2\pi)^{-1/2}\int_x^\infty (t/x)\exp(-t^2/2)\mathrm{d}t = (2\pi)^{-1/2}\exp(-x^2/2)/x.\end{aligned}$$

Plugging $x = mcp^{1/2}$ yields $O(p^{-1/2}\exp\{-\tilde{c}p)\}$ for some constant $\tilde{c} > 0$. Omitting the dominated $p^{-1/2}$ leads to the $O(\exp\{-\tilde{c}p)\}$ rate.

Software

The software is hosted and maintained on github repository under the following link:
`https://github.com/leoduan/graph_acc_mcmc`

References

Oriol Abril-Pla, Virgile Andreani, Colin Carroll, Larry Dong, Christopher J Fonnesbeck, Maxim Kochurov, Ravin Kumar, Junpeng Lao, Christian C Luhmann, Osvaldo A Martin, Michael Osthege, Ricardo Vieira, Thomas Wiecki, and Robert Zinkov. PyMC: A Modern, and Comprehensive Probabilistic Programming Framework in Python. *PeerJ Computer Science*, 9:e1516, 2023.

Michael Betancourt, Simon Byrne, Sam Livingstone, and Mark Girolami. The Geometric Foundations of Hamiltonian Monte Carlo. *Bernoulli*, 23(4A):2257 – 2298, 2017. doi: 10.3150/16-BEJ810. URL https://doi.org/10.3150/16-BEJ810.

Joris Bierkens. Non-Reversible Metropolis-Hastings. *Statistics and Computing*, 26(6):1213–1228, 2016.

Joris Bierkens, Paul Fearnhead, and Gareth Roberts. The Zig-Zag Process and Super-Efficient Sampling for Bayesian Analysis of Big Data. *The Annals of Statistics*, 47(3):1288 – 1320, 2019. doi: 10.1214/18-AOS1715. URL https://doi.org/10.1214/18-AOS1715.

David M Blei, Alp Kucukelbir, and Jon D McAuliffe. Variational Inference: A Review for Statisticians. *Journal of the American Statistical Association*, 112(518):859–877, 2017.

Stephen Boyd, Persi Diaconis, and Lin Xiao. Fastest Mixing Markov Chain on a Graph. *SIAM Review*, 46(4):667–689, 2004.

James Bradbury, Roy Frostig, Peter Hawkins, Matthew James Johnson, Chris Leary, Dougal Maclaurin, George Necula, Adam Paszke, Jake VanderPlas, Skye Wanderman-Milne, and Qiao Zhang. JAX: Composable transformations of Python+NumPy programs, 2018. URL http://github.com/google/jax.

Sébastien Bubeck. Convex Optimization: Algorithms and Complexity. *Foundations and Trends® in Machine Learning*, 8(3–4):231–357, November 2015. ISSN 1935-8237, 1935-8245. doi: 10.1561/2200000050.

Trevor Campbell and Xinglong Li. Universal Boosting Variational Inference. *Advances in Neural Information Processing Systems*, 32, 2019.

Siddhartha Chib and Bradley P Carlin. On MCMC Sampling in Hierarchical Longitudinal Models. *Statistics and Computing*, 9(1):17–26, 1999.

Cyril Chimisov, Krzysztof Latuszynski, and Gareth O. Roberts. Air Markov Chain Monte Carlo. *arXiv preprint arXiv:1801.09309*, 2018.

Oswaldo LV Costa and François Dufour. Stability and Ergodicity of Piecewise Deterministic Markov Processes. *SIAM Journal on Control and Optimization*, 47(2):1053–1077, 2008.

Arnak S Dalalyan. Theoretical Guarantees for Approximate Sampling From Smooth and Log-Concave Densities. *Journal of the Royal Statistical Society Series B: Statistical Methodology*, 79 (3):651–676, 2017.

Leo L Duan, Alexander L Young, Akihiko Nishimura, and David B Dunson. Bayesian Constraint Relaxation. *Biometrika*, 107(1):191–204, 2020.

Emilien Dupont, Arnaud Doucet, and Yee Whye Teh. Augmented Neural ODEs. In *Advances in Neural Information Processing Systems*, volume 32, 2019.

Alain Durmus and Éric Moulines. High-Dimensional Bayesian Inference via the Unadjusted Langevin Algorithm. *Bernoulli*, 25(4A):2854–2882, 2019.

Alain Durmus, Gareth O. Roberts, Gilles Vilmart, and Konstantinos C. Zygalakis. Fast Langevin based algorithm for MCMC in high dimensions. *The Annals of Applied Probability*, 27 (4):2195–2237, 2017. doi: 10.1214/16-AAP1257. URL `https://doi.org/10.1214/16-AAP1257`.

Alain Durmus, Éric Moulines, and Eero Saksman. Irreducibility and Geometric Ergodicity of Hamiltonian Monte Carlo. 2020.

Paul Fearnhead, Joris Bierkens, Murray Pollock, and Gareth O Roberts. Piecewise Deterministic Markov Processes for Continuous-Time Monte Carlo. *Statistical Science*, 33(3):386–412, 2018.

Marylou Gabrié, Grant M Rotskoff, and Eric Vanden-Eijnden. Adaptive Monte Carlo Augmented with Normalizing Flows. *Proceedings of the National Academy of Sciences*, 119(10): e2109420119, 2022.

Alan E Gelfand. Gibbs Sampling. *Journal of the American Statistical Association*, 95(452):1300–1304, 2000.

Alan E Gelfand, Adrian FM Smith, and Tai-Ming Lee. Bayesian Analysis of Constrained Parameter and Truncated Data Problems Using Gibbs Sampling. *Journal of the American Statistical Association*, 87(418):523–532, 1992.

Andrew Gelman. Parameterization and Bayesian Modeling. *Journal of the American Statistical Association*, 99(466):537–545, 2004.

Andrew Gelman, Walter R Gilks, and Gareth O Roberts. Weak Convergence and Optimal Scaling of Random Walk Metropolis Algorithms. *The Annals of Applied Probability*, 7(1):110–120, 1997. ISSN 1050-5164, 2168-8737. doi: 10.1214/aoap/1034625254.

Ryan Giordano, Tamara Broderick, and Michael I Jordan. Covariances, Robustness and Variational Bayes. *Journal of Machine Learning Research*, 19(51), 2018.

Mark Girolami and Ben Calderhead. Riemann Manifold Langevin and Hamiltonian Monte Carlo Methods. *Journal of the Royal Statistical Society: Series B (Statistical Methodology)*, 73(2): 123–214, 2011.

Heikki Haario, Eero Saksman, and Johanna Tamminen. Adaptive Proposal Distribution for Random Walk Metropolis Algorithm. *Computational Statistics*, 14:375–395, 1999.

Matthew Hoffman, Pavel Sountsov, Joshua V Dillon, Ian Langmore, Dustin Tran, and Srinivas Vasudevan. Neutra-Lizing Bad Geometry in Hamiltonian Monte Carlo Using Neural Transport. In *Symposium on Advances in Approximate Bayesian Inference*, pages 1–6, 2018.

James Johndrow, Paulo Orenstein, and Anirban Bhattacharya. Scalable Approximate MCMC Algorithms for the Horseshoe Prior. *Journal of Machine Learning Research*, 21(73), 2020.

James E Johndrow, Aaron Smith, Natesh Pillai, and David B Dunson. MCMC for Imbalanced Categorical Data. *Journal of the American Statistical Association*, 114(527):1394–1403, 2019.

Galin L Jones, Gareth O Roberts, and Jeffrey S Rosenthal. Convergence of Conditional Metropolis-Hastings Samplers. *Advances in Applied Probability*, 46(2):422–445, 2014.

Zhifeng Kong and Kamalika Chaudhuri. The Expressive Power of a Class of Normalizing Flow Models. In *International Conference on Artificial Intelligence and Statistics*, pages 3599–3609. PMLR, 2020.

Joseph B Kruskal. On the Shortest Spanning Subtree of a Graph and the Traveling Salesman Problem. *Proceedings of the American Mathematical Society*, 7(1):48–50, 1956.

László Lovász and Miklós Simonovits. On the Randomized Complexity of Volume and Diameter. In *Proceedings., 33rd Annual Symposium on Foundations of Computer Science*, pages 482–492. IEEE Computer Society, 1992.

Yulong Lu and Jianfeng Lu. A Universal Approximation Theorem of Deep Neural Networks for Expressing Probability Distributions. *Advances in Neural Information Processing Systems*, 33: 3094–3105, 2020.

Yi-An Ma, Niladri S. Chatterji, Xiang Cheng, Nicolas Flammarion, Peter L. Bartlett, and Michael I. Jordan. Is There an Analog of Nesterov Acceleration for Gradient-Based MCMC? *Bernoulli*, 27(3):1942 – 1992, 2021. doi: 10.3150/20-BEJ1297. URL `https://doi.org/10.3150/20-BEJ1297`.

Yu A Malkov and Dmitry A Yashunin. Efficient and Robust Approximate Nearest Neighbor Search Using Hierarchical Navigable Small World Graphs. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 42(4):824–836, 2018.

Andrew C Miller, Nicholas J Foti, and Ryan P Adams. Variational Boosting: Iteratively Refining Posterior Approximations. In *International Conference on Machine Learning*, pages 2420–2429. PMLR, 2017.

Kazuo Murota. Discrete Convex Analysis. *Mathematical Programming*, 83(1-3):313–371, 1998.

Viacheslav Natarovskii, Daniel Rudolf, and Björn Sprungk. Geometric Convergence of Elliptical Slice Sampling. In *International Conference on Machine Learning*, pages 7969–7978. PMLR, 2021.

Radford M Neal. Slice Sampling. *The Annals of Statistics*, 31(3):705–767, 2003.

Akihiko Nishimura, David B Dunson, and Jianfeng Lu. Discontinuous Hamiltonian Monte Carlo for Discrete Parameters and Discontinuous Likelihoods. *Biometrika*, 107(2):365–380, 2020.

Ari Pakman and Liam Paninski. Auxiliary-Variable Exact Hamiltonian Monte Carlo Samplers for Binary Distributions. In *Advances in Neural Information Processing Systems*, volume 26, 2013.

George Papamakarios, Eric Nalisnick, Danilo Jimenez Rezende, Shakir Mohamed, and Balaji Lakshminarayanan. Normalizing Flows for Probabilistic Modeling and Inference. *The Journal of Machine Learning Research*, 22(1):2617–2680, 2021.

Omiros Papaspiliopoulos, Gareth O Roberts, and Martin Sköld. A General Framework for the Parametrization of Hierarchical Models. *Statistical Science*, pages 59–73, 2007.

Cristian Pasarica and Andrew Gelman. Adaptively Scaling the Metropolis Algorithm Using Expected Squared Jumped Distance. *Statistica Sinica*, pages 343–364, 2010.

Du Phan, Neeraj Pradhan, and Martin Jankowiak. Composable Effects for Flexible and Accelerated Probabilistic Programming in NumPyro. In *Program Transformations for ML Workshop at NeurIPS 2019*, 2019.

Gilles Pisier. *The Volume of Convex Bodies and Banach Space Geometry*, volume 94. Cambridge University Press, 1999.

Nicholas G Polson, James G Scott, and Jesse Windle. Bayesian Inference for Logistic Models Using Pólya–Gamma Latent Variables. *Journal of the American Statistical Association*, 108 (504):1339–1349, 2013.

Emilia Pompe, Chris Holmes, and Krzysztof Łatuszyński. A Framework for Adaptive MCMC Targeting Multimodal Distributions. *The Annals of Statistics*, 48(5):2930 – 2952, 2020. doi: 10.1214/19-AOS1916. URL `https://doi.org/10.1214/19-AOS1916`.

Rick Presman and Jason Xu. Distance-to-Set Priors and Constrained Bayesian Inference. In *International Conference on Artificial Intelligence and Statistics*, pages 2310–2326. PMLR, 2023.

Robert Clay Prim. Shortest Connection Networks and Some Generalizations. *The Bell System Technical Journal*, 36(6):1389–1401, 1957.

Christian P Robert, Víctor Elvira, Nick Tawn, and Changye Wu. Accelerating MCMC Algorithms. *Wiley Interdisciplinary Reviews: Computational Statistics*, 10(5):e1435, 2018.

Gareth O Roberts and Nicholas G Polson. On the Geometric Convergence of the Gibbs Sampler. *Journal of the Royal Statistical Society: Series B (Methodological)*, 56(2):377–384, 1994.

Gareth O Roberts and Jeffrey S Rosenthal. Optimal Scaling of Discrete Approximations to Langevin Diffusions. *Journal of the Royal Statistical Society: Series B (Statistical Methodology)*, 60(1): 255–268, 1998.

Gareth O Roberts and Jeffrey S Rosenthal. Convergence of Slice Sampler Markov Chains. *Journal of the Royal Statistical Society: Series B (Statistical Methodology)*, 61(3):643–660, 1999.

Gareth O. Roberts and Jeffrey S. Rosenthal. Optimal Scaling for Various Metropolis-Hastings Algorithms. *Statistical Science*, 16(4):351–367, 2001. ISSN 0883-4237, 2168-8745. doi: 10.1214/ss/1015346320.

Gareth O Roberts and Jeffrey S Rosenthal. General State Space Markov Chains and MCMC Algorithms. 2004.

Gareth O Roberts and Jeffrey S Rosenthal. Examples of Adaptive MCMC. *Journal of Computational and Graphical Statistics*, 18(2):349–367, 2009.

Gareth O Roberts and Adrian FM Smith. Simple Conditions for the Convergence of the Gibbs Sampler and Metropolis-Hastings Algorithms. *Stochastic Processes and Their Applications*, 49 (2):207–216, 1994.

Gareth O Roberts and Richard L Tweedie. Exponential Convergence of Langevin Distributions and Their Discrete Approximations. *Bernoulli*, pages 341–363, 1996.

Håvard Rue, Sara Martino, and Nicolas Chopin. Approximate Bayesian Inference for Latent Gaussian Models by Using Integrated Nested Laplace Approximations. *Journal of the Royal Statistical Society Series B: Statistical Methodology*, 71(2):319–392, 2009.

Jascha Sohl-Dickstein, Mayur Mudigonda, and Michael DeWeese. Hamiltonian Monte Carlo Without Detailed Balance. In *International Conference on Machine Learning*, pages 719–726. PMLR, 2014.

Rong Tang and Yun Yang. On the Computational Complexity of Metropolis-Adjusted Langevin Algorithms for Bayesian Posterior Sampling. *Journal of Machine Learning Research*, 25(157): 1–79, 2024.

Martin A Tanner and Wing Hung Wong. The Calculation of Posterior Distributions by Data Augmentation. *Journal of the American Statistical Association*, 82(398):528–540, 1987.

Peter Toth, Danilo J. Rezende, Andrew Jaegle, Sébastien Racanière, Aleksandar Botev, and Irina Higgins. Hamiltonian Generative Networks. In *International Conference on Learning Representations*, 2020.

Roman Vershynin. Estimation in High Dimensions: A Geometric Perspective. In *Sampling Theory, a Renaissance: Compressive Sensing and Other Developments*, pages 3–66. Springer, 2015.

Maxime Vono, Daniel Paulin, and Arnaud Doucet. Efficient MCMC Sampling with Dimension-Free Convergence Rate Using ADMM-type Splitting. *The Journal of Machine Learning Research*, 23 (1):1100–1168, 2022.

Steven Winter, Trevor Campbell, Lizhen Lin, Sanvesh Srivastava, and David B Dunson. Emerging Directions in Bayesian Computation. *Statistical Science*, 39(1):62–89, 2024.

Si-Yu Yi, Ze Liu, Min-Qian Liu, and Yong-Dao Zhou. Global Likelihood Sampler for Multimodal Distributions. *Journal of Computational and Graphical Statistics*, pages 927–937, 2023.