

Markov chain Monte Carlo without evaluating the target: an auxiliary variable approach

Wei Yuan, Guanyang Wang

Department of Statistics, Rutgers University

June 11, 2024

Abstract

In sampling tasks, it is common for target distributions to be known up to a normalizing constant. However, in many situations, evaluating even the unnormalized distribution can be costly or infeasible. This issue arises in scenarios such as sampling from the Bayesian posterior for tall datasets and the ‘doubly-intractable’ distributions. In this paper, we begin by observing that seemingly different Markov chain Monte Carlo (MCMC) algorithms, such as the exchange algorithm [50], PoissonMH [76], and TunaMH [75], can be unified under a simple common procedure. We then extend this procedure into a novel framework that allows the use of auxiliary variables in both the proposal and acceptance-rejection steps. We develop the theory of the new framework, applying it to existing algorithms to simplify and extend their results. Several new algorithms emerge from this framework, with improved performance demonstrated on both synthetic and real datasets.

1 Introduction

The general workflow of Markov chain Monte Carlo algorithms is similar, yet computational challenges arise in their own way. Interestingly, solutions to these unique challenges sometimes also share a common underlying principle, which often go unnoticed even by their proposers. This paper aims to explore this phenomenon by providing a unified framework for several varied MCMC algorithms across various Bayesian inference applications. New algorithms also emerge from this framework.

We briefly recall the workflow of Metropolis–Hastings type algorithms [48, 33], one of the most general and popular MCMC methods. Given a target distribution, these methods

typically iterate over a two-step process: they first propose a new state from a proposal distribution, given the current state. Then, they apply an acceptance-rejection mechanism to decide if the transition to the new state should occur. While the choices of the proposal vary by algorithm, their implementation often requires evaluating the unnormalized distribution—that is, the target distribution up to a scaling factor. Several algorithms such as the Metropolis-adjusted Langevin algorithm (MALA) [15] and Hamiltonian Monte Carlo (HMC) [27] also require evaluating the derivative of the unnormalized distribution.

In Bayesian inference, users are interested in different quantities associated with the posterior distribution $\pi(\theta | x)$, where x denotes all observed data, and θ is the parameter of interest. With a predefined prior $\pi(\theta)$ over the parameters and the model likelihood $p_\theta(x)$, the posterior distribution can be calculated as $\pi(\theta | x) = \pi(\theta)p_\theta(x) / \int p_\theta(x)\pi(d\theta)$ according to the Bayes rule. To ‘understand’ the typically complex posterior, MCMC is often the go-to strategy. Establishing a prior, selecting an MCMC algorithm aimed at the posterior, and examining posterior samples constitute the process of the modern Bayesian inference, with MCMC acting as the backbone. We refer the readers to a recent review [47] for more historical background, stories, and successful applications.

Although MCMC needs only the knowledge of the unnormalized posterior, which is $\pi(\theta)p(x | \theta)$ as a function of θ for fixed x , challenges still arise when evaluating such a function is expensive or not feasible. The source of difficulty can stem from both the θ and the x side. Consider the following scenarios:

Scenario 1 (Expensive θ : Doubly-intractable distribution). Suppose the model likelihood $p_\theta(x)$ is expressed as $f_\theta(x)/Z(\theta)$, with $Z(\theta)$ requiring a high-dimensional integration or the summation of an exponential number of terms, making it costly or impossible to compute directly. Standard Metropolis–Hastings type algorithms cannot be directly applied to these problems, as they necessitate calculating the ratio $Z(\theta')/Z(\theta)$ at each step in its acceptance-rejection process. The posterior distribution $\pi(\theta | x) \propto \pi(\theta)f_\theta(x)/Z(\theta)$ is called the **doubly-intractable distribution**, a term coined by [50]. This distribution has found use in Bayesian inference for large graphical models [61, 14], autologistic models for spatial data [13], permutation models [66, 26], spatial point processes [63], matrix-valued data [34, 59], among many others.

Scenario 2 (Expensive x : Tall Dataset). Imagine that users have gathered a large dataset $x = (x_1, \dots, x_N)$, where N can be in the millions or billions. Each data point is independently and identically distributed following the distribution p_θ . In this case, the joint model likelihood is $p_\theta(x) = \prod_{i=1}^N p_\theta(x_i)$. However, executing standard Metropolis–Hastings algorithms becomes expensive, as the per-step acceptance-rejection computation requires evaluations

across the full dataset. This scenario is often referred to as “tall data” [10], which represents the kind of datasets commonly found in machine learning tasks [72, 1].

Various approaches have been proposed to address these challenges. Here, we review three algorithms: the exchange algorithm (Algorithm 1, [50]) for doubly-intractable distributions; PoissonMH (Algorithm 2, [76]) and TunaMH (Algorithm 3, [75]) for tall datasets. These algorithms have demonstrated promising empirical performance compared to other existing solutions, and are most relevant to our proposed framework and new algorithms. Additional relevant studies will be discussed in Section 1.2. For completeness, the beginning of each algorithm’s pseudocode contains a section titled ”promise,” which describes the technical assumptions and practical prerequisites that users need to know before executing the algorithm. It is worth noting that the posterior with N i.i.d. data points satisfies this promise of PoissonMH (and TunaMH) by setting $\phi_i(\theta, x) := N^{-1} \log \pi(\theta) + \log p_\theta(x_i)$ (and $U_i(\theta, x) := -N^{-1} \log \pi(\theta) - \log p_\theta(x_i)$).

Algorithm 1 Exchange Algorithm

- 1: **Promise:**
 - Target $\pi(\theta | x) \propto \pi(\theta)p_\theta(x)$,
 - Likelihood $p_\theta(x) = f_\theta(x)/Z(\theta)$ where the value of f_θ can be queried
 - A simulator $\mathcal{S}(\cdot)$ with input θ . Each run outputs a random variable $w \sim p_\theta$

- 2: **Initialize:** Initial state θ_0 , proposal q ; number of iterations T

- 3: **for** $t = 0$ to $T - 1$ **do**

- 4: propose $\theta' \sim q(\theta_t, \cdot)$

- 5: sample $\omega \sim \mathcal{S}(\cdot)$

- 6: compute

$$r \leftarrow \frac{\pi(\theta')f_{\theta'}(x)f_{\theta_t}(\omega)}{\pi(\theta_t)f_{\theta_t}(x)f_{\theta'}(\omega)} \cdot \frac{q(\theta', \theta_t)}{q(\theta_t, \theta')}$$

- 7: with probability $\min\{1, r\}$, set $\theta_{t+1} \leftarrow \theta'$; otherwise, $\theta_{t+1} \leftarrow \theta_t$

- 8: **end for**

The similarities and differences among these algorithms are both clear. Conceptually, all three algorithms are designed to sample from the posterior distribution without directly computing the (unnormalized) target distribution. In particular, they all preserve the correct stationary distribution, distinguishing them from many other algorithms without this property, such as [72]. On the other hand, the methodologies of these algorithms might appear irrelevant at first glance. Indeed, they are designed with different motivations, targeting different challenges, and operating under different assumptions. For example, the exchange algorithm is intended to sample from doubly-intractable distributions. It assumes that users

Algorithm 2 PoissonMH

1: **Promise:**

- Target $\pi(\theta | x) \propto \exp\{\sum_{i=1}^N \phi_i(\theta; x)\}$,
- For every θ , each $\phi_i(\theta; x) \in [0, M_i]$ with some $M_i > 0$

2: **Initialize:** Initial state θ_0 ; proposal q ; hyperparameter λ ; set $L := \sum_{i=1}^N M_i$; number of iterations T

3: **for** $t = 0$ to $T - 1$ **do**

4: propose $\theta' \sim q(\theta_t, \cdot)$

5: sample $s_i \sim \text{Poi}\left(\frac{\lambda M_i}{L} + \phi_i(\theta_t; x)\right)$ for each $i \in \{1, 2, \dots, N\}$, form minibatch $S = \{i \mid s_i > 0\}$

6: compute

$$r \leftarrow \frac{\exp\left\{\sum_{i \in S} s_i \log\left(1 + \frac{L}{\lambda M_i} \phi_i(\theta'; x)\right)\right\}}{\exp\left\{\sum_{i \in S} s_i \log\left(1 + \frac{L}{\lambda M_i} \phi_i(\theta_t; x)\right)\right\}} \cdot \frac{q(\theta', \theta_t)}{q(\theta_t, \theta')}$$

7: with probability $\min\{1, r\}$, set $\theta_{t+1} \leftarrow \theta'$; otherwise $\theta_{t+1} \leftarrow \theta_t$

8: **end for**

Algorithm 3 TunaMH

1: **Promise:**

- Target $\pi(\theta | x) \propto \exp\{-\sum_{i=1}^N U_i(\theta; x)\}$,
- For every θ and θ' , each $|U_i(\theta'; x) - U_i(\theta; x)| \leq c_i M(\theta, \theta')$ for some symmetric non-negative function M and positive constants c_i

2: **Initialize:** Initial state θ_0 ; proposal q ; hyperparameter χ ; set $C = \sum_{i=1}^N c_i$; number of iterations T

3: **for** $t = 0$ to $T - 1$ **do**

4: propose $\theta' \sim q(\theta_t, \cdot)$ and compute $M(\theta_t, \theta')$

5: set $\lambda = \chi C^2 M^2(\theta, \theta')$

6: sample $s_i \sim \text{Poi}\left(\frac{\lambda c_i}{C} + \phi_i(\theta_t, \theta'; x)\right)$ where $\phi_i(\theta_t, \theta'; x) = \frac{U_i(\theta'; x) - U_i(\theta_t; x)}{2} + \frac{c_i}{2} M(\theta_t, \theta')$, for each $i \in \{1, 2, \dots, N\}$, form minibatch $S = \{i \mid s_i > 0\}$

7: compute

$$r \leftarrow \frac{\exp\left\{\sum_{i \in S} s_i \log\left(1 + \frac{C}{\lambda c_i} \phi_i(\theta', \theta_t; x)\right)\right\}}{\exp\left\{\sum_{i \in S} s_i \log\left(1 + \frac{C}{\lambda c_i} \phi_i(\theta_t, \theta'; x)\right)\right\}} \cdot \frac{q(\theta', \theta_t)}{q(\theta_t, \theta')}$$

8: with probability $\min\{1, r\}$, set $\theta_{t+1} \leftarrow \theta'$; otherwise $\theta_{t+1} \leftarrow \theta_t$

9: **end for**

have a simulator to generate synthetic data from the model for each fixed parameter. In contrast, PoissonMH and TunaMH are designed to avoid evaluating the entire dataset at each iteration, thereby addressing the “tall data” problem. PoissonMH and TunaMH also impose different technical assumptions on the form of the model likelihood. Therefore, their con-

ceptual similarities and methodological distinctions prompt us to ask: *Can these algorithms be understood within a unified framework?*

1.1 Our contribution

Our first contribution answers the above question affirmatively. We begin with a key observation. The exchange, PoissonMH, and TunaMH algorithms can be unified into a simple common procedure, detailed in Section 2. This procedure reveals the inherent similarities among these algorithms.

Furthermore, we take a step further to develop new algorithms. We notice that existing algorithms without evaluating the full posterior predominantly support random-walk-type proposals—which mix slowly in high-dimensional sampling problems, with reasons detailed in Section 3.1. Motivated by this, our major contribution is a novel framework in Section 3.2. Thanks to this framework, we develop new gradient-based minibatch algorithms that allow users to choose a high-efficiency proposal. We develop new variants of ‘locally balanced proposals’ [74, 44] and Stochastic Gradient Langevin Dynamics (SGLD) [72] to improve the PoissonMH and TunaMH algorithms. These new algorithms support the use of gradient-based proposals while preserving all other desired properties (correct stationary distribution, no evaluation of the target) with minimal overhead. They show significantly improved efficiency on both synthetic and real datasets compared to their predecessors [76, 75] as well as off-the-shelf algorithms like random-walk Metropolis, MALA, and HMC.

The framework itself may be of independent interest. Methodologically, it appears to be flexible and practical. It considers using auxiliary variables in both the proposal and acceptance-rejection steps. It includes previous frameworks [65] and Section 2, which focus exclusively on one aspect, thereby offering greater potential for further algorithm design. Theoretically, the framework is naturally linked to the theory of Markov chain comparison, an active area with several recent progresses [4, 56, 57, 9]. Both classical [25, 29] and recent techniques should be well-suited for analyzing the framework itself or its crucial applications. As applications of our theoretical analysis in Section 4, we can recover, strengthen, and extend several existing results, such as those in [70, 76, 75], in a simpler way.

1.2 Related works

Both the doubly-intractable problem (Scenario 1) and tall dataset problem (Scenario 2) are studied extensively in statistics and machine learning. Here we provide a concise review of the related literature. For the doubly-intractable problem, the exchange algorithm (Algorithm 1) is one of the most popular approaches. Further extensions, including [41], [42], and

[2], improve its efficiency or relax its assumptions. Theoretical properties of the exchange algorithm are examined in [53] and [70]. It can be seen as part of a broader framework introduced in [6]. Another popular solution, described in [49], falls within the pseudo-marginal MCMC framework [43, 12, 7]. The theory of the pseudo-marginal MCMC is studied in [7, 8]. Other solutions are also available, such as the ‘Russian-Roulette sampler’ [45].

For tall datasets, many minibatch MCMC algorithms are proposed to reduce the cost per iteration. These algorithms are broadly classified into exact and inexact, based on whether they maintain the target distribution as stationary. Inexact methods, such as those in [72, 22, 39, 58, 24, 73], introduce a systematic bias to trade off for implementation efficiency. Quantifying bias is often challenging. Theorem 1 in [75] demonstrates that the eventual bias can become arbitrarily large, even when the bias per-step is arbitrarily small. In contrast, exact methods such as [46, 10, 23, 76, 75] maintain the correct stationary distribution but typically rely on additional assumptions of the target distribution. This paper primarily focuses on [75] and its predecessor [76], as they show improved empirical performance with relatively mild assumptions. Other non-reversible exact methods, such as those in [17, 16], are also appealing but can be challenging to implement. In addition to minibatch MCMC algorithms, numerous divide-and-conquer algorithms, such as [35, 71, 52], are introduced for their suitability for parallel implementation.

1.3 Notations

We use Θ to represent a measurable space equipped with a σ -algebra \mathcal{F} , from which we aim to draw samples from a given distribution. As our primary focus is on examples in Bayesian inference, we preserve the letter x to represent all the observed data and $\pi(\cdot | x)$ to denote the posterior distribution with a fixed prior $\pi(\theta)$ and likelihood $p_\theta(x)$. Given two probability distributions P, Q on the same measurable space, we use $d_{\text{TV}}(P, Q) := \sup_A |P(A) - Q(A)|$ to denote their total-variation (TV) distance. If P is absolutely continuous with respect to Q , we use $d_{\text{KL}}(P || Q) := \int \log((P/Q)(dx)) P(dx)$ to denote their Kullback–Leibler (KL) divergence. Given a differentiable function $f(x, \theta)$ with $x \in \mathbb{R}^m$ and $\theta \in \mathbb{R}^n$, we denote the partial derivative with respect to θ_i as $\partial_{\theta_i} f := \partial f / \partial \theta_i$, and $\nabla_\theta f := (\partial_{\theta_1} f, \partial_{\theta_2} f, \dots, \partial_{\theta_n} f)$. Similarly, the gradient on x is denoted as $\nabla_x f$. For $x \in \mathbb{R}^m$, we denote $\|x\|_p := (|x_1|^p + |x_2|^p + \dots + |x_m|^p)^{1/p}$ for $p > 0$. We use $\text{Poi}(\cdot)$, $\mathbb{N}(\cdot, \cdot)$, and $\text{Unif}(\cdot, \cdot)$ to denote the Poisson, Gaussian, and uniform distribution, respectively. In particular, $\text{Unif}(A, k)$ represents k elements uniformly chosen from a finite set A without replacement.

1.4 Organization

We present a common procedure that unifies the exchange algorithm, PoissonMH, and Tu-naMH in Section 2. Our main framework extends this further and is presented as a meta-algorithm (Algorithm 4) in Section 3. Section 3.3 introduces new algorithms as special cases of our meta-algorithm. Section 4 provides a theory for the mixing time of Algorithm 4, with applications in Section 4.4. Section 5 demonstrates the effectiveness of our new algorithms through three experiments. Section 6 concludes this paper with future directions. Additional proofs, experiment details, and extra experiments are provided in the Appendix.

2 A common procedure of existing algorithms

Although the details of Algorithm 1,2,3 differ significantly, they all follow the following general procedure at each iteration. Given target distribution $\pi(\cdot | x)$, a proposal kernel $q(\cdot, \cdot)$, and current state θ :

1. Propose a new state $\theta' \sim q(\theta, \cdot)$
2. Generate an auxiliary variable ω according to certain distribution denoted as $P_{\theta \rightarrow \theta'}(\cdot)$
3. Construct an estimator $R_{\theta \rightarrow \theta'}(\omega)$ for $\pi(\theta' | x)/\pi(\theta | x)$, set $r := R_{\theta \rightarrow \theta'}(\omega)q(\theta', \theta)/q(\theta, \theta')$
4. With probability $\min\{r, 1\}$, set $\theta_{\text{new}} := \theta'$. Otherwise, $\theta_{\text{new}} := \theta$.

Compared to standard Metropolis–Hastings algorithms, this method uses an estimator based on an auxiliary variable to estimate the target ratio $\pi(\theta' | x)/\pi(\theta | x)$ instead of directly computing it. This procedure still defines a Markov chain on Θ . It will also preserve $\pi(\cdot | x)$ as the stationary distribution if the following equation is satisfied:

Proposition 1. *Let K be the one-step Markov transition kernel defined by the process described above. Suppose for every $\theta, \theta' \in \Theta \times \Theta$, the estimator $R_{\theta \rightarrow \theta'}(\omega)$ satisfies*

$$R_{\theta \rightarrow \theta'}(\omega)\pi(\theta | x)P_{\theta \rightarrow \theta'}(\omega) = \pi(\theta' | x)P_{\theta' \rightarrow \theta}(\omega). \quad (1)$$

Then we have:

- *The estimator $R_{\theta \rightarrow \theta'}$ is unbiased for $\pi(\theta' | x)/\pi(\theta | x)$.*
- *The Markov chain with transition kernel K is reversible with respect to $\pi(\theta | x)$.*

Proof of Proposition 1. For unbiasedness, integrating both sides of (1) with respect to ω gives:

$$\pi(\theta | x)\mathbb{E}_{P_{\theta \rightarrow \theta'}}[R_{\theta \rightarrow \theta'}(\omega)] = \pi(\theta' | x) \int P_{\theta' \rightarrow \theta}(\omega) d\omega = \pi(\theta' | x),$$

where the second equality follows from the fact that integrating a probability measure over the entire space equals one. Dividing both sides by $\pi(\theta | x)$ gives us the desired result.

For reversibility, we prove this by checking the detailed balance equation. For every fixed $\theta \neq \theta'$, we aim to show:

$$\pi(\theta | x)K(\theta, \theta') = \pi(\theta' | x)K(\theta', \theta)$$

Expanding the left side of the above equation:

$$\begin{aligned} \pi(\theta | x)K(\theta, \theta') &= \pi(\theta | x)q(\theta, \theta')\mathbb{E}_\omega \left[\min \left(R_{\theta \rightarrow \theta'}(\omega) \frac{q(\theta', \theta)}{q(\theta, \theta')}, 1 \right) \right] \\ &= \int_\Omega \pi(\theta | x)q(\theta, \theta')P_{\theta \rightarrow \theta'}(\omega) \min \left(R_{\theta \rightarrow \theta'}(\omega) \frac{q(\theta', \theta)}{q(\theta, \theta')}, 1 \right) d\omega \\ &= \int_\Omega \min (\pi(\theta | x)P_{\theta \rightarrow \theta'}(\omega)R_{\theta \rightarrow \theta'}(\omega)q(\theta', \theta), \pi(\theta | x)q(\theta, \theta')P_{\theta \rightarrow \theta'}(\omega)) d\omega \\ &= \int_\Omega \min (\pi(\theta' | x)P_{\theta' \rightarrow \theta}(\omega)q(\theta', \theta), \pi(\theta | x)q(\theta, \theta')P_{\theta \rightarrow \theta'}(\omega)) d\omega \quad \text{by (1)} \\ &= \int_\Omega \min (\pi(\theta' | x)P_{\theta' \rightarrow \theta}(\omega)q(\theta', \theta), q(\theta, \theta')R_{\theta' \rightarrow \theta}(\omega)\pi(\theta' | x)P_{\theta' \rightarrow \theta}(\omega)) d\omega \quad \text{by (1)} \\ &= \int_\Omega \pi(\theta' | x)P_{\theta' \rightarrow \theta}(\omega)q(\theta', \theta) \min \left(1, \frac{q(\theta, \theta')}{q(\theta', \theta)} R_{\theta' \rightarrow \theta}(\omega) \right) d\omega \\ &= \pi(\theta' | x)q(\theta', \theta)\mathbb{E}_\omega \left[\min \left(1, \frac{q(\theta, \theta')}{q(\theta', \theta)} R_{\theta' \rightarrow \theta}(\omega) \right) \right] \\ &= \pi(\theta' | x)K(\theta', \theta), \end{aligned}$$

gives the desired result. \square

Now we are ready to verify that the exchange algorithm in [50] for doubly-intractable distributions, as well as PoissonMH and TunaMH [75, 76] for tall dataset, can all be viewed as examples of this procedure. They all satisfy equation (1) in Proposition 1.

Example 1 (Exchange algorithm). The exchange algorithm (Algorithm 1) considers sampling from posterior distribution of the form $\pi(\theta|x) \propto p(\theta) \frac{f_\theta(x)}{Z(\theta)}$ with intractiable $Z(\theta)$, as described in Scenario 1. In the exchange algorithm, the auxiliary variable ω is an element in the sample space, which can be viewed as synthetic data. Meanwhile:

- The distribution $P_{\theta \rightarrow \theta'}(\omega) := p_{\theta'}(\omega) = f_{\theta'}(\omega)/Z(\theta')$.

- The estimator

$$R_{\theta \rightarrow \theta'}(\omega) := \frac{\pi(\theta')f_{\theta'}(x)f_{\theta}(\omega)}{\pi(\theta)f_{\theta}(x)f_{\theta'}(\omega)}.$$

To check equation (1), recall $\pi(x) := \int \pi(\theta)p_{\theta}(x)d\theta$ is the marginal distribution of x :

$$\begin{aligned} R_{\theta \rightarrow \theta'}(\omega) \cdot \pi(\theta | x) \cdot P_{\theta \rightarrow \theta'}(\omega) &= \frac{\pi(\theta')f_{\theta'}(x)f_{\theta}(\omega)}{\pi(\theta)f_{\theta}(x)f_{\theta'}(\omega)} \cdot \frac{\pi(\theta)f_{\theta}(x)}{Z(\theta)\pi(x)} \cdot \frac{f_{\theta'}(\omega)}{Z(\theta')} \\ &= \frac{\pi(\theta')f_{\theta'}(x)f_{\theta}(\omega)}{\cancel{\pi(\theta)f_{\theta}(x)f_{\theta'}(\omega)}} \cdot \frac{\cancel{\pi(\theta)f_{\theta}(x)}}{Z(\theta)\pi(x)} \cdot \frac{\cancel{f_{\theta'}(\omega)}}{Z(\theta')} \\ &= \frac{\pi(\theta')f_{\theta'}(x)}{\pi(x)Z(\theta')} \cdot \frac{f_{\theta}(\omega)}{Z(\theta)} \\ &= \pi(\theta' | x)p_{\theta}(\omega) \\ &= \pi(\theta' | x)P_{\theta' \rightarrow \theta}(\omega). \end{aligned}$$

All the equalities follow from either the Bayes formula, or the problem definition.

Example 2 (PoissonMH). PoissonMH (Algorithm 2) aims to sample from the posterior distribution $\pi(\theta | x) \propto \exp(\sum_{i=1}^N \phi_i(\theta; x))$, which is the product of numerous data points. The auxiliary variable $\omega = (s_1, s_2, \dots, s_N)$ belongs to $\{0, 1, 2, \dots\}^N$, i.e., a vector of natural numbers with the same length as the number of data points. Each component of ω follows an independent Poisson distribution with different parameters. More precisely:

- The distribution $P_{\theta \rightarrow \theta'} := \bigotimes_{i=1}^N \text{Poi}\left(\frac{\lambda M_i}{L} + \phi_i(\theta; x)\right)$.
- The estimator

$$R_{\theta \rightarrow \theta'}(\omega) := \frac{\exp\left\{\sum_{i \in S} s_i \log\left(1 + \frac{L}{\lambda M_i} \phi_i(\theta'; x)\right)\right\}}{\exp\left\{\sum_{i \in S} s_i \log\left(1 + \frac{L}{\lambda M_i} \phi_i(\theta; x)\right)\right\}},$$

with all the notations defined in Algorithm 2. Here the estimator relies solely on the data points in S (the strictly positive entries of ω). With appropriate selections of the parameters, the expected size of S may be significantly lower than the number of data points, which explains the reduction in computational cost per iteration.

To check equation (1), recall $\pi(\theta | x) = \exp(\sum_{i=1}^N \phi_i(\theta; x)) / Z(x)$:

$$\begin{aligned}
R_{\theta \rightarrow \theta'}(\omega) \cdot \pi(\theta | x) \cdot P_{\theta \rightarrow \theta'}(\omega) &= R_{\theta \rightarrow \theta'}(\omega) \frac{\exp(\sum_{i=1}^N \phi_i(\theta; x))}{Z(x)} \prod_{i=1}^N P_{\theta \rightarrow \theta'}(s_i) \\
&= \frac{1}{Z(x)} \prod_{i=1}^N \left(\frac{\lambda M_i L^{-1} + \phi_i(\theta'; x)}{\lambda M_i L^{-1} + \phi_i(\theta; x)} \right)^{s_i} \cdot e^{\phi_i(\theta; x)} \cdot e^{-\lambda M_i L^{-1}} e^{-\phi_i(\theta; x)} \frac{(\lambda M_i L^{-1} + \phi_i(\theta; x))^{s_i}}{s_i!} \\
&= \frac{1}{Z(x)} \prod_{i=1}^N \left(\frac{\lambda M_i L^{-1} + \phi_i(\theta'; x)}{\lambda M_i L^{-1} + \phi_i(\theta; x)} \right)^{s_i} \cdot e^{\phi_i(\theta; x)} \cdot e^{-\lambda M_i L^{-1}} e^{-\phi_i(\theta; x)} \frac{(\lambda M_i L^{-1} + \phi_i(\theta; x))^{s_i}}{s_i!} \\
&= \frac{1}{Z(x)} \prod_{i=1}^N \frac{(\lambda M_i L^{-1} + \phi_i(\theta'; x))^{s_i}}{s_i!} (e^{-\lambda M_i L^{-1}} e^{-\phi_i(\theta'; x)}) e^{\phi_i(\theta'; x)} \\
&= \pi(\theta' | x) \prod_{i=1}^N P_{\theta' \rightarrow \theta}(s_i) \\
&= \pi(\theta' | x) P_{\theta' \rightarrow \theta}(\omega).
\end{aligned}$$

Example 3 (TunaMH). TunaMH (Algorithm 3) aims to solve the same problem as PoissonMH under more practical assumptions. The posterior is assumed to again have a product form $\pi(\theta) \propto \exp\left\{-\sum_{i=1}^N U_i(\theta; x)\right\}$. The boundedness assumption in PoissonMH is replaced by a Lipschitz-like condition: $|U_i(\theta) - U_i(\theta')| \leq c_i M(\theta, \theta')$. Similar to PoissonMH, the auxiliary variable $\omega = (s_1, s_2, \dots, s_N) \in \{0, 1, 2, \dots\}^N$. Each s_i follows an independent Poisson distribution with different parameters. More precisely:

- The distribution $P_{\theta \rightarrow \theta'} := \bigotimes_{i=1}^N \text{Poi}\left(\frac{\lambda c_i}{C} + \phi_i(\theta, \theta'; x)\right)$.
- The estimator

$$R_{\theta \rightarrow \theta'}(\omega) := \frac{\exp\left\{\sum_{i \in S} s_i \log\left(1 + \frac{C}{\lambda c_i} \phi_i(\theta', \theta; x)\right)\right\}}{\exp\left\{\sum_{i \in S} s_i \log\left(1 + \frac{C}{\lambda c_i} \phi_i(\theta, \theta'; x)\right)\right\}}.$$

with all the notations defined in Algorithm 3.

The verification of equation (1) can be derived through a calculation similar to that of Example 2, which is deferred to Appendix A.1.

After observing that the framework naturally includes the aforementioned algorithms, we provide three additional remarks here. First, the novelty of our finding in this section lies in discovering the inherent connections between algorithms designed for doubly-intractable distributions and those for tall datasets. The common procedure itself is not new. In Section

2.1 of [3], the authors present a slightly broader framework in which they substitute $P_{\theta' \rightarrow \theta}(\omega)$ with $P_{\theta' \rightarrow \theta}(\varphi(\omega))$, where φ represents any measurable involution. Therefore our framework can be viewed as taking $\varphi = \text{id}$, the identity map. On the other hand, the primary goal in [3] is to improve the quality of the estimator within the exchange algorithm while preserving the stationary distribution. Consequently, they do not consider the issue of tall datasets.

Second, it is interesting to observe that Algorithm 1, 2, and 3 use three different ways of generating the auxiliary variable. More precisely:

- In the exchange algorithm, ω depends only on the proposed state θ' .
- In PoissonMH, ω depends only on the current state θ .
- In TunaMH, ω depends on both the current θ and the proposed state θ' .

While this may appear as a simple finding, it will significantly impact the design of new algorithms in Section 3.

Third, a close examination of Algorithm 2 and 3 shows that the key ingredient is the ‘Poisson estimator’, which provides an unbiased estimate of the exponential of an expectation. In the case of tall dataset, the quantity of interest is the target ratio which can be represented as $\exp\{\sum_{i=1}^N (\phi_i(\theta'; x) - \phi_i(\theta; x))\}$ in Algorithm 2 (or $\exp\{(-\sum_{i=1}^N U_i(\theta'; x) + U_i(\theta; x))\}$ in Algorithm 3). This ratio can be evaluated precisely, albeit at a high cost. Alternatively, we can express the ratio as $\exp(\mathbb{E}[f(I)])$, where $f(i) := N(\phi_i(\theta'; x) - \phi_i(\theta; x))$ and $I \sim \text{Unif}\{1, 2, \dots, N\}$. This formulation aligns the problem with the Poisson estimator’s framework. An additional subtlety is the estimator has to be non-negative, as it will be used to determine the acceptance probability (Step 7–8 in Algorithm 2, and Step 8–9 in Algorithm 3). Consequently, all the technical assumptions and design elements in both Algorithm 2 and 3 are intended to ensure that the non-negativity of the Poisson estimator. Other applications of the Poisson estimator can be found in [30, 69, 54] and the references therein. This insight can be interpreted either positively or negatively: it implies that there is potential for developing new algorithms using similar ideas under weaker assumptions (such as non-i.i.d. data or less stringent technical conditions). However, eliminating all technical assumptions seems unlikely due to the impossibility result in Theorem 2.1 of [37].

The current common procedure already provides a convenient basis for analyzing the theoretical properties of these algorithms. Nevertheless, we choose to postpone the theoretical analysis after Section 3, where a more comprehensive framework will be introduced. By then, we will recognize the advantages of studying within a more general framework: all results can be directly applied to existing algorithms, and some even lead to improved results. Meanwhile, many existing tools used for analyzing specific cases remain insightful for our new analysis and can be naturally generalized.

3 A new framework with (two) auxiliary variables

3.1 Motivation: gradient-based proposal design

Another notable similarity among the exchange algorithm, PoissonMH, and TunaMH is that their proposal distributions predominantly involve either independent or random-walk types. For more details, see Section 5 of [50], Section 4 and Appendix D of [76], and Section 5 of [75]. However, these types of proposals are known to mix poorly with dimensionality. Gradient-based MCMC algorithms, such as MALA and HMC, are regarded as much more efficient because they utilize gradient information to guide the proposed state.

A simple idea is to choose q as a gradient-based proposal in the procedure described in Section 2. However, upon second thought, one will quickly realize the challenges in implementing such a strategy:

- For the doubly-intractable distribution, the gradient of the log-posterior equals:

$$\nabla_{\theta} \log \left(\frac{\pi(\theta) f_{\theta}(x)}{Z(\theta)} \right) = \nabla_{\theta} \log (\pi(\theta) f_{\theta}(x)) - \nabla_{\theta} \log(Z(\theta)),$$

which involves the gradient of the unknown $Z(\theta)$. The challenge in (directly) using a gradient-based proposal lies in the difficulty of evaluating $\nabla_{\theta} \log(Z(\theta))$.

- For sampling from the tall dataset, recall from Scenario 2 that the posterior has the form $\pi(\theta | x) \propto \pi(\theta) \prod_{i=1}^N p_{\theta}(x_i)$, the gradient of the log-posterior equals:

$$\nabla_{\theta} \log (\pi(\theta | x)) = \nabla_{\theta} \log (\pi(\theta)) + \sum_{i=1}^N \nabla_{\theta} \log(p_{\theta}(x_i)).$$

Thus, computing the gradient again requires evaluations across the entire dataset, incurring a cost of $\Theta(N)$. Since the primary goal of minibatch MCMC algorithms like PoissonMH and TunaMH is to reduce the cost per iteration, incorporating gradient information would directly violate this objective.

Nevertheless, designing an ‘informed proposal’ remains a fascinating idea. With this motivation in mind, we introduce a novel framework (that extends the procedure in Section 2) in Section 3.2. This framework leads to new algorithms that use cheap proxies for the actual gradient to guide the proposed moves, as illustrated in Section 3.3.

3.2 General methodology

To introduce auxiliary-variable-based MCMC methods, we first present certain notations and assumptions in both algorithmic and probabilistic languages. While the definitions may appear concise or abstract, subsequent texts will provide intuitions and examples to clarify.

Supposing users can access two simulators for generating auxiliary variables, labeled as \mathcal{S}_1 and \mathcal{S}_2 . The first simulator takes θ as input and produces a random variable ω_1 based on a distribution that might rely on θ . The second simulator takes $(\theta, \theta', \omega_1)$ as input and outputs another random variable ω_2 . Formally, consider two measurable spaces $(\Omega_1, \mathcal{F}_1)$ and $(\Omega_2, \mathcal{F}_2)$ with base measure λ_1, λ_2 , respectively. We define two families of probability measures: the first family \mathbb{P}_θ is defined on Ω_1 and parameterized by θ in Θ ; the second family $\mathbb{P}_{\theta, \theta'}(\cdot | \omega_1)$ is defined on Ω_2 and parameterized by both $(\theta, \theta') \in \Theta \times \Theta$ and $\omega_1 \in \Omega_1$. The second family can be understood as the conditional distribution of ω_2 given ω_1 with parameters (θ, θ') . For any fixed pair (θ, θ') , the provided definition naturally leads to a joint distribution on $\Omega_1 \times \Omega_2$, expressed as $\mathbb{P}_{\theta, \theta'}(\omega_1, \omega_2) = \mathbb{P}_\theta(\omega_1)\mathbb{P}_{\theta, \theta'}(\omega_2 | \omega_1)$. Thus, the assumption regarding simulators essentially implies that users can simulate from both the marginal distribution of ω_1 and the conditional distribution of ω_2 . Additionally, we introduce a unique single-point probability space $\text{NULL} := \{\text{null}\}$. Scenarios where either Ω_1 or Ω_2 are set to be NULL should be interpreted as no auxiliary variables being generated. Lastly, we introduce a family of proposal kernels, labeled as $\{Q_{\omega_1}\}_{\omega_1 \in \Omega_1}$, where each specific ω_1 is associated with a Markov transition kernel Q_{ω_1} on Θ . We assume each $Q_{\omega_1}(\theta, \cdot)$ admits a density $q_{\omega_1}(\theta, \cdot)$ with respect to a fixed reference measure λ on Θ .

Our intuition of introducing ω_1, ω_2 is actually simple. We can view ω_1 as an auxiliary variable for determining the proposal; naturally, its distribution can solely depend on the current state θ . The sampled ω_1 , combined with the current state θ , determines the proposed state θ' . Subsequently, the second auxiliary variable ω_2 is generated to estimate the ratio of the target distribution. The distribution of ω_2 may depend on the current state θ , the proposed state θ' , and the first auxiliary variable ω_1 . One example of ω_1 might involve selecting a minibatch uniformly from a large dataset. This minibatch is then used to estimate the gradient of the whole dataset, subsequently influencing the proposal. One example of ω_2 could be the synthetic data generated at each iteration of the exchange algorithm. Further examples will be presented shortly.

The procedure of our auxiliary-variable-based meta-algorithm is detailed in Algorithm 4. For an additional illustration, refer to Figure 1. Algorithm 4 is presented in its general form, assuming the target is a generic distribution Π on Θ . In all applications discussed in this paper, $\Pi(\cdot)$ is taken as the posterior distribution $\pi(\cdot | x)$. It is important to note that, evaluating the ratio is in Step 6 can be much cheaper than evaluating $\Pi(\theta')/\Pi(\theta_t)$. The

costly part of $\Pi(\theta')/\Pi(\theta_t)$ is often offset by the ratio $\mathbb{P}_{\theta',\theta_t}(\omega_1, \omega_2)/\mathbb{P}_{\theta_t,\theta'}(\omega_1, \omega_2)$ by design.

Algorithm 4 Auxiliary-based MCMC

- 1: **Initialize:** Initial state θ_0 , auxiliary-based proposal $\{q_{\omega_1}\}_{\omega_1 \in \Omega_1}$; number of iterations T ; target distribution $\Pi(\theta)$
 - 2: **for** $t = 0$ to $T - 1$ **do**
 - 3: sample $\omega_1 \sim \mathbb{P}_{\theta_t}(\cdot)$ via \mathcal{S}_1
 - 4: propose $\theta' \sim q_{\omega_1}(\theta_t, \cdot)$
 - 5: sample $\omega_2 \sim \mathbb{P}_{\theta_t, \theta'}(\cdot | \omega_1)$ via \mathcal{S}_2
 - 6: compute the acceptance ratio
- $$r \leftarrow \frac{\Pi(\theta') \mathbb{P}_{\theta', \theta_t}(\omega_1, \omega_2)}{\Pi(\theta_t) \mathbb{P}_{\theta_t, \theta'}(\omega_1, \omega_2)} \cdot \frac{q_{\omega_1}(\theta', \theta_t)}{q_{\omega_1}(\theta_t, \theta')}$$
- 7: with probability $\min\{1, r\}$, set $\theta_{t+1} \leftarrow \theta'$; otherwise, $\theta_{t+1} \leftarrow \theta_t$
 - 8: **end for**
-

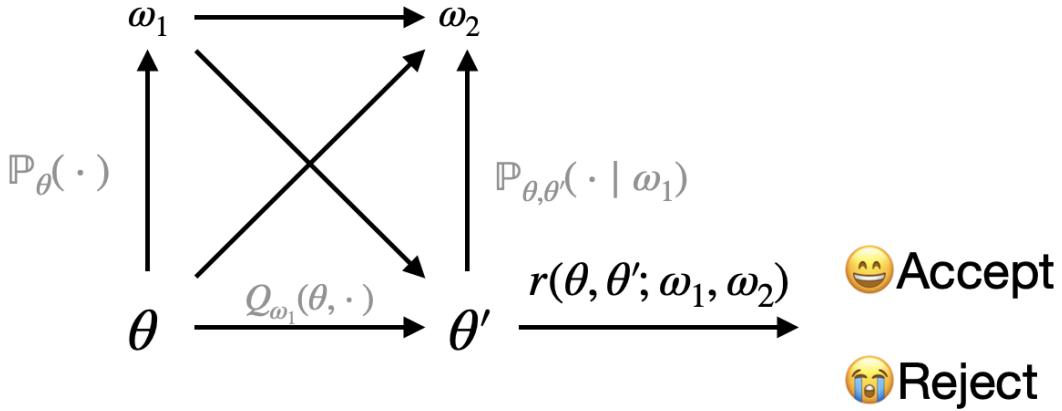


Figure 1: The workflow of Algorithm 4

The validity of this algorithm can be proven by checking the detailed balance equation:

Proposition 2. Let $\mathbb{P}_{\text{aux}}(\cdot, \cdot)$ be the transition kernel of the Markov chain defined in Algorithm 4. We have $\mathbb{P}_{\text{aux}}(\cdot, \cdot)$ is reversible respect to Π . ¹

¹The ‘accept-rejection’ mechanism ($\min\{1, r\}$) in Step 7) can be generalized to accept θ' with a probability of $a(r)$, where $a : [0, \infty) \rightarrow [0, 1]$ is any function satisfying $a(t) = ta(1/t)$. The reversibility claimed here still holds. See Appendix A.5 for the proof.

Proof of Proposition 2. For each $\theta \in \Theta$, the probability distribution $\mathbb{P}_{\text{aux}}(\theta, \cdot)$ can be decomposed as a point-mass at θ and a density on Θ , namely:

$$\mathbb{P}_{\text{aux}}(\theta, d\theta') = R(\theta)\delta_\theta(d\theta') + p_{\text{aux}}(\theta, \theta')\lambda(d\theta'),$$

where λ is the base measure on Θ defined earlier. Our goal is to show $\pi(\theta)p_{\text{aux}}(\theta, \theta') = \pi(\theta')p_{\text{aux}}(\theta', \theta)$ for every pair (θ, θ') satisfying $\theta \neq \theta'$. Reversibility will directly result from this equality. The term $p_{\text{aux}}(\theta, \theta')$ describes the density of successfully moving from θ to θ' , including two steps: proposing θ' and then accepting the transition. Therefore we have the following expression of $p_{\text{aux}}(\theta, \theta')$:

$$p_{\text{aux}}(\theta, \theta') = \int_{\Omega_1} \mathbb{P}_\theta(\omega_1) q_{\omega_1}(\theta, \theta') \int_{\Omega_2} \left(\min \left\{ 1, \frac{\Pi(\theta') \mathbb{P}_{\theta', \theta}(\omega_1, \omega_2)}{\Pi(\theta) \mathbb{P}_{\theta, \theta'}(\omega_1, \omega_2)} \cdot \frac{q_{\omega_1}(\theta', \theta)}{q_{\omega_1}(\theta, \theta')} \right\} \mathbb{P}_{\theta, \theta'}(\omega_2 | \omega_1) \lambda_2(d\omega_2) \right) \lambda_1(d\omega_1).$$

Therefore we have

$$\begin{aligned} \Pi(\theta)p_{\text{aux}}(\theta, \theta') &= \\ &\int \min \left\{ \Pi(\theta) \mathbb{P}_{\theta, \theta'}(\omega_1, \omega_2) q_{\omega_1}(\theta, \theta'), \Pi(\theta') \mathbb{P}_{\theta', \theta}(\omega_1, \omega_2) q_{\omega_1}(\theta', \theta) \right\} \lambda_1 \otimes \lambda_2(d\omega_1 d\omega_2), \end{aligned}$$

which is symmetric over θ and θ' . This shows $\Pi(\theta)p_{\text{aux}}(\theta, \theta') = \Pi(\theta')p_{\text{aux}}(\theta', \theta)$. \square

Algorithm 4 is a meta-algorithm, requiring users to define the process for generating ω_1 and ω_2 (or opting not to generate them), and selecting the proposal distribution. The immediate advantage of Algorithm 4, together with Proposition 2, lies in its ability to provide a general approach for incorporating auxiliary variables in algorithm design while preserving the stationary distribution. Meanwhile, the flexibility offered by the auxiliary variables (ω_1 and ω_2 can be independent or arbitrarily correlated) allows users to create creative methods according to their preferences.

From now on, we will always assume Π is the posterior $\pi(\cdot | x)$ without further specification. We will now briefly discuss a few such possibilities, including algorithms we've reviewed and illustrating how they align with our framework. In the next section, we will see how this framework gives rise to new algorithms.

Case 1 (No auxiliary variable). When both $\Omega_1 = \Omega_2 = \text{NULL}$, then no auxiliary variable is

generated. Algorithm 4 reduces to the standard Metropolis–Hastings type algorithm.

Case 2 (Without ω_1). If $\Omega_1 = \text{NULL}$, Algorithm 4 does not utilize any auxiliary variable for designing the proposal distribution. However, ω_2 can still be generated for estimating the target ratio. In this case, the acceptance ratio in Step 6 of Algorithm 4 simplifies to

$$\frac{\pi(\theta' | x)\mathbb{P}_{\theta',\theta_t}(\omega_2)}{\pi(\theta_t | x)\mathbb{P}_{\theta_t,\theta'}(\omega_2)} \cdot \frac{q(\theta', \theta_t)}{q(\theta_t, \theta')}.$$

This scenario recovers the previous common procedure discussed in Section 2, which in turn includes the exchange algorithm, PoissonMH, and TunaMH (Algorithm 1, 2, 3) as special cases.

Case 3 ($\omega_1 = \omega_2$). Another interesting case is when $\omega_1 = \omega_2$, i.e., the two auxiliary variables are perfectly correlated. In this case, the acceptance ratio reduces to

$$\frac{\pi(\theta' | x)\mathbb{P}_{\theta'}(\omega_1)}{\pi(\theta_t | x)\mathbb{P}_{\theta_t}(\omega_1)} \cdot \frac{q_{\omega_1}(\theta', \theta_t)}{q_{\omega_1}(\theta_t, \theta')}.$$

In this case ω_1 can clearly be used to help with designing the proposal distribution. In certain instances, this approach can help estimate the target ratio and decrease the per-iteration cost through careful design, as we will explore in Section 3.3.1. This scenario includes the auxiliary Metropolis–Hastings sampler (Section 2.1 of [65]) as a special case. However, a potential limitation is that the auxiliary ω_1 is generated solely based on the current state θ_t , and does not use the information from the proposed θ' . Incorporating information from θ' could potentially further improve the target ratio estimator.

Case 4 (ω_1 independent with ω_2). When ω_1 is independent with ω_2 , the acceptance ratio can be written as

$$\frac{\pi(\theta' | x)\mathbb{P}_{\theta'}(\omega_1)\mathbb{P}_{\theta',\theta_t}(\omega_2)}{\pi(\theta_t | x)\mathbb{P}_{\theta_t}(\omega_1)\mathbb{P}_{\theta_t,\theta'}(\omega_2)} \cdot \frac{q_{\omega_1}(\theta', \theta_t)}{q_{\omega_1}(\theta_t, \theta')}.$$

Methodologically, this approach can be viewed as separately addressing the issues of ‘designing the proposal’ and ‘estimating the ratio’. For instance, users might generate one minibatch to estimate the log-gradient of the target evaluated at the current state, which then informs the proposal. Then, another independent minibatch could be used to estimate the target ratio without directly evaluating it. We will discuss this strategy in more detail in Section 3.3.2. Particularly, when ω_1 follows a fixed distribution which does not depend on θ , the acceptance ratio simplifies to

$$\frac{\pi(\theta' | x)\mathbb{P}_{\theta',\theta_t}(\omega_2)}{\pi(\theta_t | x)\mathbb{P}_{\theta_t,\theta'}(\omega_2)} \cdot \frac{q_{\omega_1}(\theta', \theta_t)}{q_{\omega_1}(\theta_t, \theta')}.$$

3.3 New algorithms

We now introduce new gradient-based minibatch algorithms within our framework (Algorithm 4). For concreteness, we focus on sampling from a tall dataset where $\pi(\theta \mid x) \propto \pi(\theta)p_\theta(x) = \pi(\theta)\prod_{i=1}^N p_\theta(x_i)$. Recall that our objective is to use gradient information in the proposal design while maintaining both the correct stationary distribution and minimal additional overhead compared to existing methods, ensuring that each iteration only requires a minibatch of data. Our high-level idea involves introducing an auxiliary variable (ω_1 in Algorithm 4) to estimate the full gradient with a small cost. Subsequently, we use a second auxiliary variable, as described in Section 2, to estimate the target ratio. Depending on the specific application, these two variables might be either independent or correlated.

3.3.1 PoissonMH with locally balanced proposal

In this section, we will adhere to all the technical assumptions specified in PoissonMH (Algorithm 2). We aim to improve its performance by choosing a gradient-based proposal that still operates within a minibatch of the dataset per iteration.

Locally balanced proposal: We first briefly review the idea of the locally balanced proposal. The idea is initially introduced in [74] for discrete samplers and later extended in [44, 67]. It has proven to be a convenient framework for integrating traditional algorithms and developing new algorithms.

The original Barker’s proposal in [74] takes the form

$$Q^g(\theta, d\theta') \propto g\left(\frac{\pi(\theta' \mid x)}{\pi(\theta \mid x)}\right) K(\theta, d\theta')$$

where $g : \mathbb{R}^+ \rightarrow \mathbb{R}^+$ is a *balancing function* satisfying $g(t) = tg(1/t)$. Here, K represents a symmetric kernel, often thought of as the ‘uninformed proposal’. Users can choose the balancing function, with recommended options including $g(t) = \sqrt{t}$ and $g(t) = t/(1+t)$. Thus, the proposed distribution is a weighted version of K , with the weight function g selected to balance ‘proposing long moves’ and ‘high acceptance probability’. Furthermore, the function g is an increasing function evaluating the target ratio, indicating it favors movement toward points with higher probability mass/density. Experiments on various discrete distributions show that its performance is competitive compared to alternative MCMC methods.

When θ is in a continuous state space, the proposal $Q^g(\theta, \cdot)$ is not feasible for implementation. However, one can perform a first-order Taylor expansion of $\log(\pi(\theta' \mid x)) - \log(\pi(\theta \mid x))$ with respect to θ , and utilize the so-called *first-order* locally balanced proposal. It has the

form:

$$Q^{(g)}(\theta, d\theta') = \prod_{i=1}^d Q_i^{(g)}(\theta, d\theta'_i). \quad (2)$$

Here $Q_i^{(g)}$ is a one-dimensional kernel of the form

$$Q_i^{(g)}(\theta, d\theta'_i) = Z_i^{-1}(\theta) g\left(e^{\partial_{\theta_i} \log \pi(\theta|x)(\theta'_i - \theta_i)}\right) \mu_i(\theta'_i - \theta_i) d\theta'_i \quad (3)$$

where $\mu_i(\cdot)$ represents a symmetric density in \mathbb{R} , such as a centered Gaussian. The authors in [44] note that selecting $g(t) = \sqrt{t}$ corresponds to MALA [13]. Furthermore, they suggest using $g(t) = t/(1+t)$, referred to as 'Barker's proposal', inspired by [11]. This choice allows exact calculation of $Z_i(\theta)$ (which is a constant 0.5), and in turn implies an efficient algorithm for the proposal distribution $Q^{(g)}(\theta, \cdot)$. The authors in [44] find that Barker's proposal still incorporates gradient information and tends to be more robust compared to MALA.

Our proposal: First, we introduce our proposed algorithm and then explain the concept behind it. We retain all promises and notations used in Algorithm 2. We also use the notation $\omega_1 := (s_1, s_2, \dots, s_N)$ and $\mathbb{P}_\theta(\cdot) := \otimes_{i=1}^N \text{Poi}\left(\frac{\lambda M_i}{L} + \phi_i(\theta; x)\right)$, as defined in Example 2. For every balancing function g , we also define the Markov transition kernels with density

$$q_{\omega_1}^{(g)}(\theta, \theta') := \prod_{i=1}^d q_{\omega_1, i}^{(g)}(\theta, \theta'_i). \quad (4)$$

Here $q_{\omega_1, i}^{(g)}$ is a one-dimensional density proportional to

$$g\left(e^{\partial_{\theta_i} \log(\pi(\theta|x)\mathbb{P}_\theta(\omega_1))(\theta'_i - \theta_i)}\right) \mu_i(\theta'_i - \theta_i), \quad (5)$$

where μ_i is again a symmetric density on \mathbb{R} .

Algorithm 5 corresponds to Case 3 of our meta-algorithm 4. Consequently, its validity is directly proven by Proposition 2. Moreover, several points are worth discussing.

- (Comparison with the original locally-balanced proposal:) The proposed method is an auxiliary-variable-based variant of the locally-balanced proposal. Our proposal distribution $q_{\omega_1}^{(g)}$ closely resembles the first-order locally balanced proposal, with one key modification. The term $\partial_{\theta_i} \log \pi(\theta | x)$ from formula (2) is substituted by $\partial_{\theta_i} \log (\pi(\theta | x)\mathbb{P}_\theta(\omega_1))$ in formula (4). Therefore, instead of calculating the gradient of the log-posterior, we compute $\nabla \log_\theta (\pi(\theta | x)\mathbb{P}_\theta(\omega_1))$, which depends on ω_1 .

Algorithm 5 Locally Balanced PoissonMH

- 1: **Initialize:** Initial state θ_0 ; balancing function g ; number of iterations T
- 2: **for** $t = 0$ to $T - 1$ **do**
- 3: sample $\omega_1 = (s_1, s_2, \dots, s_N) \sim \mathbb{P}_{\theta_t}(\cdot)$, form minibatch $S = \{i \mid s_i > 0\}$
- 4: propose $\theta' \sim q_{\omega_1}^{(g)}(\theta_t, \cdot)$
- 5: compute
$$r \leftarrow \frac{\pi(\theta' \mid x)\mathbb{P}_{\theta'}(\omega_1)}{\pi(\theta_t \mid x)\mathbb{P}_{\theta_t}(\omega_1)} \cdot \frac{q_{\omega_1}^{(g)}(\theta', \theta_t)}{q_{\omega_1}^{(g)}(\theta_t, \theta')}.$$
- 6: with probability $\min\{1, r\}$, set $\theta_{t+1} \leftarrow \theta'$; otherwise, $\theta_{t+1} \leftarrow \theta_t$
- 7: **end for**

- (Design motivation:) As discussed in Section 3.1, both MALA and the Barker proposal are unsuitable for scenarios involving tall data because they require the full gradient. To obtain a ‘cheap’ estimate for $\nabla \log \pi(\theta \mid x)$, our key observation is the auxiliary variable ω_1 depends only on the current state θ . Therefore, one may as well first generate the minibatch and then propose the new step, i.e. swap the order of Steps 4 and 5 in Algorithm 2. Then, $\pi(\theta \mid x)\mathbb{P}_{\theta}(\omega_1)$ is employed as a computationally efficient approximation for the gradient, with the detailed calculations provided below. Therefore, the auxiliary variable ω_1 is simultaneously used for estimating both the gradient and the target ratio.
- (Computation efficiency:) Recall the definition of $\mathbb{P}_{\theta}(\omega_1)$ and $\pi(\theta \mid x)$ in Example 2, we have

$$\begin{aligned} \pi(\theta \mid x) \cdot \mathbb{P}_{\theta}(\omega_1) &= \frac{\exp(\sum_{i=1}^N \phi_i(\theta; x))}{Z(x)} \prod_{i=1}^N \mathbb{P}_{\theta}(s_i) \\ &= \frac{1}{Z(x)} \prod_{i=1}^N e^{\phi_i(\theta; x)} \cdot e^{-\lambda M_i L^{-1}} \frac{e^{-\phi_i(\theta; x)} (\lambda M_i L^{-1} + \phi_i(\theta; x))^{s_i}}{s_i!} \\ &= \frac{1}{Z(x)} e^{-\lambda} \prod_{i:s_i>0} \frac{(\lambda M_i L^{-1} + \phi_i(\theta; x))^{s_i}}{s_i!}. \end{aligned}$$

This indicates that the proxy function $\pi(\theta \mid x) \cdot \mathbb{P}_{\theta}(\omega_1)$ only depends on **the minibatch** S in Step 2 of Algorithm 5. Consequently, the gradient of the log proxy function can be computed much more efficiently than $\nabla_{\theta} \log(\pi(\theta \mid x))$. The cost of computing the gradient is at the same level as calculating the acceptance ratio in PoissonMH, thus not significantly increasing the cost per step.

- (Implementation:) Following the suggestions in [44], we choose $g(t) = t/(1+t)$ and $g(t) = \sqrt{t}$ in our actual implementation. We call them Poisson-Barker and Poisson-

MALA respectively. These are the minibatch versions (covering both proposal generation and target ratio evaluation) of the Barker algorithm in [44] and MALA. Similar to [44], these choices of g allow efficient implementation, with details available in Appendix B. We examine their performances and provide comparisons with PoissonMH and other standard MCMC algorithms (that use the full dataset) in Section 5.

3.3.2 TunaMH with SGLD proposal

In this section, we will adhere to all the technical assumptions in TunaMH (Algorithm 3) as we discuss our strategy for designing a gradient-based proposal to improve it. Recall the target distribution has the form $\pi(\theta | x) \propto \exp\{-\sum_{i=1}^N U_i(\theta; x)\}$.

A key aspect of Algorithm 5 is the use of the same auxiliary variable for both estimating the gradient and estimating the target ratio, i.e., choosing $\omega_1 = \omega_2$ in our meta-algorithm Algorithm 4. However, this approach becomes infeasible if we aim to improve TunaMH. As noted at the end of Section 2, the auxiliary variable in TunaMH (Algorithm 3) depends on both the current and proposed states. Consequently, it is necessary to first propose a new state before generating this auxiliary variable, rather than the other way around.

Therefore, we adopt another cost-effective strategy for utilizing the gradient information, inspired by the SGLD algorithm in [72]. At each step, we first select a minibatch B of K data points uniformly from the entire dataset. Thus, a natural estimator of the gradient of the log-posterior, $\nabla_\theta \log \pi(\theta | x) = -\nabla_\theta \sum_{i=1}^N U_i(\theta; x)$, is $(N/K) \sum_{i \in B} \nabla_\theta U_i(\theta_t; x)$. The computation of this estimator has a cost that scales linearly with the minibatch size K , rather than with the total dataset size N . In the examples we considered, data size N is no less than 10,000 while K is often chosen as 20. Therefore, computing the estimated gradient is significantly cheaper compared to calculating the full gradient. Setting $\omega_1 := B \sim \text{Unif}\{\{1, 2, \dots, N\}, K\}$, our proposal has the form:

$$q_{\omega_1}(\theta, \cdot) \sim \mathbb{N} \left(\theta - \frac{\epsilon^2}{2} \frac{N}{K} \sum_{i \in B} \nabla_\theta U_i(\theta; x), \epsilon^2 \mathbb{I} \right),$$

where ϵ is a tuning parameter chosen by users. Next, we use the same strategy as TunaMH to select an independent minibatch from a product Poisson distribution to estimate the target ratio. Using the notations $\omega_2 := (s_1, s_2, \dots, s_N)$ and $\mathbb{P}_{\theta, \theta'}(\cdot) := \otimes_{i=1}^N \text{Poi}\left(\frac{\lambda c_i}{C} + \phi_i(\theta, \theta'; x)\right)$, our algorithm is described below:

Algorithm 6 TunaMH-SGLD

- 1: **Initialize:** Initial state θ_0 ; batch size K for the first minibatch; step size ϵ ; number of iterations T
 - 2: **for** $t = 0$ to $T - 1$ **do**
 - 3: sample the first minibatch $B \subset \{1, 2, \dots, N\}$ uniformly at random with size K
 - 4: propose $\theta' \sim q_{\omega_1}(\theta_t, \cdot)$
 - 5: sample $\omega_2 = (s_1, s_2, \dots, s_N) \sim \mathbb{P}_{\theta_t, \theta'}(\cdot)$, form the second minibatch $S = \{i \mid s_i > 0\}$
 - 6: compute
$$r \leftarrow \frac{\pi(\theta' \mid x)\mathbb{P}_{\theta', \theta_t}(\omega_2)}{\pi(\theta_t \mid x)\mathbb{P}_{\theta_t, \theta'}(\omega_2)} \cdot \frac{q_{\omega_1}(\theta', \theta_t)}{q_{\omega_1}(\theta_t, \theta')}.$$
 - 7: with probability $\min\{1, r\}$, set $\theta_{t+1} \leftarrow \theta'$; otherwise $\theta_{t+1} \leftarrow \theta_t$
 - 8: **end for**
-

Algorithm 6 corresponds to Case 4 of our meta-algorithm 4. Thus, its validity directly follows from Proposition 2. In this case, our first auxiliary variable ω_1 is independent of the second, meaning the second generator \mathcal{S}_2 no longer depends on ω_1 . Therefore, the distribution $\mathbb{P}_{\theta, \theta'}(\cdot \mid \omega_1)$ (in Step 5 of Algorithm 4) equals $\mathbb{P}_{\theta, \theta'}(\cdot)$. Additionally, the first variable $\omega_1 = B$ follows a fixed uniform distribution that does not depend on θ . Consequently, the joint distribution $\mathbb{P}_{\theta, \theta'}(\omega_1, \omega_2)$ equals $\mathbb{P}(\omega_1)\mathbb{P}_{\theta, \theta'}(\omega_2)$. As a result, the target ratio (in Step 6 of Algorithm 4) simplifies to

$$\frac{\pi(\theta' \mid x)\mathbb{P}_{\theta', \theta_t}(\omega_2)}{\pi(\theta_t \mid x)\mathbb{P}_{\theta_t, \theta'}(\omega_2)} \cdot \frac{q_{\omega_1}(\theta', \theta_t)}{q_{\omega_1}(\theta_t, \theta')},$$

which is precisely what we have in Step 6 of Algorithm 6.

For readers already familiar with the SGLD framework, another natural perspective is to view Algorithm 6 as a minibatch Metropolized version of SGLD. If each iteration of Algorithm 6 skips Steps 5–7 for the accept-reject correction, it becomes exactly the SGLD algorithm described in [72] with a fixed step size. The SGLD algorithm is a noisier but much faster version of the Unadjusted Langevin Algorithm (ULA), also known as the Langevin Monte Carlo algorithm (LMC). When $K = N$, it exactly recovers ULA. Both ULA and SGLD with a fixed step size are known to have systematic bias, as they do not converge to the target distribution $\pi(\theta \mid x)$ even after an infinite number of steps [68]. Introducing a straightforward Metropolis–Hastings acceptance-rejection step can ensure the correct stationary distribution but requires full-dataset evaluation, which undermines the primary advantage of SGLD. Therefore, deriving an acceptance-rejection version of SGLD using mini-batch data is considered an open problem. To quote from [72]: *Interesting directions of future research includes deriving a MH rejection step based on mini-batch data ...*

Algorithm 6 offers a solution to this problem under the same assumptions as Algorithm 3. It eliminates the systematic bias in SGLD by incorporating an acceptance-rejection step based on the approach used in TunaMH. Relaxing the technical assumptions or generalizing to posterior distributions with non-i.i.d. model likelihoods are intriguing open questions.

3.3.3 Additional remarks

It is natural to ask whether a similar approach could be used to leverage gradient information in the exchange algorithm. Recall the gradient of the log-posterior for doubly-intractable distributions equals:

$$\begin{aligned}\nabla_{\theta} \log(\pi(\theta)p_{\theta}(x)) &= \nabla_{\theta} \log\left(\frac{\pi(\theta)f_{\theta}(x)}{Z(\theta)}\right) \\ &= \nabla_{\theta} \log(\pi(\theta)f_{\theta}(x)) - \nabla_{\theta} \log(Z(\theta)) \\ &= \nabla_{\theta} \log(\pi(\theta)f_{\theta}(x)) - \frac{\nabla_{\theta} Z(\theta)}{Z(\theta)}.\end{aligned}$$

Assuming the model-likelihood has the form $p_{\theta}(x) = \exp\{\theta^T s(x)\}/Z(\theta)$, the authors in [2] observe that the above formula can simplified to $\nabla_{\theta} \log(\pi(\theta)p_{\theta}(x)) = \nabla_{\theta} \log(\pi(\theta)f_{\theta}(x)) - \mathbb{E}_{\omega \sim p_{\theta}(\cdot)}[s(\omega)]$. Therefore users can generate $\omega \sim p_{\theta}(\cdot)$ and use $\log(\pi(\theta)f_{\theta}(x)) - s(\omega)$ as an estimator for the gradient. Thus, a natural strategy using our meta-algorithm 4 is to set $\omega_1 \sim \mathbb{P}_{\theta}(\cdot) = p_{\theta}(\cdot)$, and $q_{\omega_1}(\theta, \cdot) = \mathbb{N}(\theta + \tau(\log(\pi(\theta)f_{\theta}(x)) - s(\omega_1)), 2\tau)$ (choosing a MALA-type proposal with an estimated gradient replacing the actual gradient), and $\omega_2 \sim \mathbb{P}_{\theta'}(\cdot)$ independent of ω_1 . This idea is similar to the MALA-exchange algorithm (Algorithm 9 in [2]), except that they recycle the auxiliary variables from the previous step and use them in the next step, while we refresh the auxiliary variables at each iteration. Their algorithm has demonstrated improved performance over the original exchange algorithm on several discrete Markov random field models.

Without assuming a specific form for $p_{\theta}(x)$, designing gradient-based proposals for the exchange algorithm remains an open problem. Additionally, the exchange algorithm requires exact samplers for $p_{\theta}(\cdot)$ for every fixed θ , which is often infeasible. In practice, users often use another approximate sampler for $p_{\theta}(\cdot)$. This leads to the modified exchange algorithm, also known as the “doubly Metropolis–Hastings sampler” [41], which involves a double loop: the outer loop updates θ_t at each iteration, while the inner loop approximately samples auxiliary variables from $p_{\theta_t}(\cdot)$ for each t . The doubly Metropolis–Hastings sampler is more practical but breaks the stationary distribution. Since this paper focuses on methods that maintain the stationary distribution, we will leave the exploration of this topic for future work.

4 Theory

This section presents our theoretical analysis. We will begin by examining the meta-algorithm 4 and then apply our findings to specific algorithms. Proposition 2 has already shown the reversibility of the proposed method. Therefore, this section will focus on understanding the convergence speed of Algorithm 4 to the target distribution.

Our agenda is to compare the Markov chain, with transition kernel denoted by \mathbb{P}_{aux} with two relevant chains $\mathbb{P}_{\text{ideal}}$ and \mathbb{P}_{MwG} . These two chains need not be practically implementable, but are easier to study as a stochastic process. Therefore, comparison results will help to determine the extent to which the properties of $\mathbb{P}_{\text{ideal}}$ or \mathbb{P}_{MwG} can be translated to \mathbb{P}_{aux} .

Now, we introduce the two new chains. Fix $\theta \in \Theta$, define the idealized proposal $Q_{\text{ideal}}(\theta, d\theta') := \mathbb{E}_{\omega_1 \sim \mathbb{P}_\theta}[Q_{\omega_1}(\theta, d\theta')]$, with density $q_{\text{ideal}}(\theta, \theta') = \int_{\Omega_1} q_{\omega_1}(\theta, \theta') \mathbb{P}_\theta(\omega_1) \lambda_1(d\omega_1)$. The idealized algorithm is described in Algorithm 7 below. It is a standard Metropolis–Hastings algorithm: its proposal distribution is an averaged version of the proposals q_{ω_1} over the auxiliary variable ω_1 , and its acceptance-rejection step is based on the entire dataset. We also notice that Q_{ideal} coincides with the actual proposal when our meta-algorithm 4 does not use ω_1 (i.e., $\Omega_1 = \text{NULL}$), including the exchange algorithm, PoissonMH, and TunaMH.

The second chain, \mathbb{P}_{MwG} , is defined as the transition kernel for Case 3. This case is part of Algorithm 4 when $\omega_1 = \omega_2$, as explained in Section 3.2. A crucial feature is that this chain (or Case 3) can equivalently be viewed as a Metropolis-within-Gibbs algorithm. More precisely, it is the marginal chain for the θ component, targeting an augmented distribution $\pi(\theta, \omega | x) := \pi(\theta | x) \mathbb{P}_\theta(\omega)$. At each step, users first generate ω_1 from $\pi(\cdot | \theta, x) = \mathbb{P}_\theta(\cdot)$. They then use the proposal $q_{\omega_1}(\theta, \cdot)$ to draw θ' , targeting $\pi(\theta' | \omega_1, x) \propto \pi(\theta' | x) \mathbb{P}_\theta(\omega_1)$, and implement an acceptance-rejection step. This perspective allows us to apply existing results on Metropolis-within-Gibbs, such as [57], in our analysis.

For each θ , all of $\mathbb{P}_{\text{ideal}}(\theta, \cdot)$, $\mathbb{P}_{\text{MwG}}(\theta, \cdot)$ and $\mathbb{P}_{\text{aux}}(\theta, \cdot)$ can be represented as a mixture of a continuous density on Θ (with respect to the base measure) and a point mass at θ . We therefore define $p_{\text{ideal}}(\theta, \cdot)$, $p_{\text{MwG}}(\theta, \cdot)$ and $p_{\text{aux}}(\theta, \cdot)$ for the density part, respectively.

Algorithm 7 Idealized MH

- 1: **Initialize:** Initial state θ_0 ; number of iterations T ; target distribution $\pi(\theta | x)$
 - 2: **for** $t = 0$ to $T - 1$ **do**
 - 3: propose $\theta' \sim Q_{\text{ideal}}(\theta, \cdot)$
 - 4: compute the acceptance ratio
- $$r \leftarrow \frac{\pi(\theta' | x)}{\pi(\theta_t | x)} \cdot \frac{q_{\text{ideal}}(\theta', \theta_t)}{q_{\text{ideal}}(\theta_t, \theta')}$$
- 5: with probability $\min\{1, r\}$, set $\theta_{t+1} \leftarrow \theta'$; otherwise, $\theta_{t+1} \leftarrow \theta_t$
 - 6: **end for**
-

4.1 Peskun's ordering

One side of the comparison is easy. The following result generalizes existing results that use only one auxiliary variable, such as Lemma 1 in [70], Section 2.3 in [53], and Section 2.1 in [3]. Nevertheless, the proof idea is very similar and is given in Appendix 4.1.

Lemma 1. *For each pair (θ, θ') such that $\theta \neq \theta'$, we have*

$$p_{\text{aux}}(\theta, \theta') \leq p_{\text{MWG}}(\theta, \theta') \leq p_{\text{ideal}}(\theta, \theta'). \quad (6)$$

This lemma demonstrates that Algorithm 4 is always less likely to move from any state to another compared to Algorithm 7 and \mathbb{P}_{MWG} .

Lemma 1 implies that $\mathbb{P}_{\text{aux}} \prec \mathbb{P}_{\text{MWG}} \prec \mathbb{P}_{\text{ideal}}$ according to Peskun's ordering [55]. Dominance in Peskun's ordering has several implications, including that $\mathbb{P}_{\text{MWG}} - \mathbb{P}_{\text{aux}}$ and $\mathbb{P}_{\text{ideal}} - \mathbb{P}_{\text{MWG}}$ are non-negative operators, and the asymptotic variance for any f under $\mathbb{P}_{\text{ideal}}$ is no greater than that under \mathbb{P}_{aux} , among others. For further results, we refer readers to [64].

4.2 Comparing the transition density

Section 4.1 states that Algorithm 4 is always less efficient than \mathbb{P}_{MWG} and $\mathbb{P}_{\text{ideal}}$. This inefficiency can be attributed to the “price” paid for the extra auxiliary variables and the computational cost savings. On the other hand, it is perhaps more interesting to understand the reverse direction, examining when Algorithm 4 retains most of the efficiency of its idealized counterparts.

We use the shorthand notation $d_{\text{TV}}(\theta, \theta', \omega_1) := d_{\text{TV}}(\mathbb{P}_{\theta, \theta'}(\omega_2 | \omega_1), \mathbb{P}_{\theta', \theta}(\omega_2 | \omega_1))$. We also define $d_{\text{TV}}(\theta, \theta') = \sup_{\omega_1} d_{\text{TV}}(\theta, \theta' | \omega_1)$. Moreover, define the largest KL-divergence

$\tilde{d}_{\text{KL}}(\theta, \theta') := \sup_{\omega_1} d_{\text{KL}}(\mathbb{P}_{\theta, \theta'}(\omega_2 | \omega_1) || \mathbb{P}_{\theta', \theta}(\omega_2 | \omega_1))$, and its symmetrized version

$$d_{\text{KL}}(\theta, \theta') := 0.5 \sup_{\omega_1} (d_{\text{KL}}(\mathbb{P}_{\theta, \theta'}(\omega_2 | \omega_1) || \mathbb{P}_{\theta', \theta}(\omega_2 | \omega_1)) + d_{\text{KL}}(\mathbb{P}_{\theta', \theta}(\omega_2 | \omega_1) || \mathbb{P}_{\theta, \theta'}(\omega_2 | \omega_1))).$$

We have the following comparison result on transition density between p_{aux} and p_{MWG} .

Theorem 1. *For each pair (θ, θ') such that $\theta \neq \theta'$, we have*

$$\begin{aligned} p_{\text{aux}}(\theta, \theta') &\geq (1 - d_{\text{TV}}(\theta, \theta')) p_{\text{MWG}}(\theta, \theta') \\ &\geq e^{-1/e} \exp\{-\min\{d_{\text{KL}}(\theta, \theta'), \tilde{d}_{\text{KL}}(\theta, \theta'), \tilde{d}_{\text{KL}}(\theta', \theta)\}\} p_{\text{MWG}}(\theta, \theta'). \end{aligned}$$

Proof of Theorem 1. We can write down the transition density as:

$$\begin{aligned} p_{\text{aux}}(\theta, \theta') &= \int_{\Omega_1 \times \Omega_2} \mathbb{P}_\theta(\omega_1) q_{\omega_1}(\theta, \theta') \mathbb{P}_{\theta, \theta'}(\omega_2 | \omega_1) r(\theta, \theta'; \omega_1, \omega_2) \lambda_1(d\omega_1 d\omega_2) \\ &= \int_{\Omega_1} \mathbb{P}_\theta(\omega_1) q_{\omega_1}(\theta, \theta') \left(\mathbb{E}_{\omega_2 | \omega_1, \theta, \theta'} \left[\min \left\{ \frac{\pi(\theta' | x) \mathbb{P}_{\theta', \theta}(\omega_1, \omega_2)}{\pi(\theta | x) \mathbb{P}_{\theta, \theta'}(\omega_1, \omega_2)} \cdot \frac{q_{\omega_1}(\theta', \theta)}{q_{\omega_1}(\theta, \theta')}, 1 \right\} \right] \right) \lambda_1(d\omega_1) \\ &\geq \int_{\Omega_1} \mathbb{P}_\theta(\omega_1) q_{\omega_1}(\theta, \theta') \left(\min \left\{ \frac{\pi(\theta' | x) q_{\omega_1}(\theta', \theta)}{\pi(\theta | x) q_{\omega_1}(\theta, \theta')} \frac{\mathbb{P}_{\theta'}(\omega_1)}{\mathbb{P}_\theta(\omega_1)}, 1 \right\} \right) \\ &\quad \times \left(\int_{\Omega_2} \min \{ \mathbb{P}_{\theta, \theta'}(\omega_2 | \omega_1), \mathbb{P}_{\theta', \theta}(\omega_2 | \omega_1) \} \lambda_2(d\omega_2) \right) \lambda_1(d\omega_1) \\ &= \int_{\Omega_1} \mathbb{P}_\theta(\omega_1) q_{\omega_1}(\theta, \theta') \left(\min \left\{ \frac{\pi(\theta' | x) q_{\omega_1}(\theta', \theta)}{\pi(\theta | x) q_{\omega_1}(\theta, \theta')} \frac{\mathbb{P}_{\theta'}(\omega_1)}{\mathbb{P}_\theta(\omega_1)}, 1 \right\} \right) \\ &\quad \times (1 - d_{\text{TV}}(\theta, \theta', \omega_1)) \lambda_1(d\omega_1) \\ &\geq (1 - d_{\text{TV}}(\theta, \theta')) \int_{\Omega_1} \mathbb{P}_\theta(\omega_1) q_{\omega_1}(\theta, \theta') \left(\min \left\{ \frac{\pi(\theta' | x) q_{\omega_1}(\theta', \theta)}{\pi(\theta | x) q_{\omega_1}(\theta, \theta')} \frac{\mathbb{P}_{\theta'}(\omega_1)}{\mathbb{P}_\theta(\omega_1)}, 1 \right\} \right) \\ &= (1 - d_{\text{TV}}(\theta, \theta')) p_{\text{MWG}}(\theta, \theta'). \end{aligned}$$

The first inequality follows from the fact $\min\{ab, cd\} \geq \min\{a, c\} \min\{b, d\}$ for $a, b, c, d \geq 0$. For the second inequality in the statement of Theorem 1, first recall the improved version of Bretagnolle–Huber inequality in Section 8.3 of [32], which states:

$$1 - d_{\text{TV}}(P, Q) \geq e^{-1/e} \exp\{-d_{\text{KL}}(P || Q)\}.$$

The original Bretagnolle–Huber inequality [18, 21] is a slightly weaker version where the constant $e^{-1/e}$ is replaced by $1/2$. Replacing P, Q by $\mathbb{P}_{\theta, \theta'}(\omega_2 | \omega_1), \mathbb{P}_{\theta', \theta}(\omega_2 | \omega_1)$, and taking

the supremum (for d_{TV}) over ω_1 , we get:

$$p_{\text{aux}}(\theta, \theta') \geq e^{-1/e} \exp\{\tilde{d}_{\text{KL}}(\theta, \theta')\} p_{\text{MwG}}(\theta, \theta').$$

Meanwhile, reversing the order between P, Q in the Bretagnolle–Huber inequality, we have

$$1 - d_{\text{TV}}(Q, P) \geq e^{-1/e} \exp\{-d_{\text{KL}}(Q||P)\}.$$

This implies

$$p_{\text{aux}}(\theta, \theta') \geq e^{-1/e} \exp\{\tilde{d}_{\text{KL}}(\theta', \theta)\} p_{\text{MwG}}(\theta, \theta').$$

Finally, multiplying the improved Bretagnolle–Huber inequality for (P, Q) and (Q, P) together and then taking the square root, we obtain:

$$1 - d_{\text{TV}}(P, Q) \geq e^{-1/e} \exp\{-0.5(d_{\text{KL}}(P||Q) + d_{\text{KL}}(Q||P))\},$$

which further shows

$$1 - d_{\text{TV}}(\theta, \theta') \geq e^{-1/e} \exp\{-d_{\text{KL}}(\theta, \theta')\}.$$

This concludes the proof of Theorem 1. \square

Theorem 1 holds for any algorithm that fits into our meta-algorithm 4. The following corollaries may be useful when working on specific algorithms.

Corollary 1. *In the following categories, we have:*

1. When $\omega_1 = \text{Null}$, we have $\mathbb{P}_{\text{MwG}} = \mathbb{P}_{\text{ideal}}$, the inequalities in Theorem 1 reads:

$$p_{\text{aux}}(\theta, \theta') \geq (1 - d_{\text{TV}}(\theta, \theta')) p_{\text{ideal}}(\theta, \theta') \geq e^{-1/e} \exp\{-d_{\text{KL}}(\theta, \theta')\} p_{\text{ideal}}(\theta, \theta').$$

2. When $\omega_1 = \omega_2$, then $d_{\text{TV}}(\theta, \theta') = 0$, and $\mathbb{P}_{\text{aux}} = \mathbb{P}_{\text{MwG}}$. The first inequality in Theorem 1 is saturated.
3. When ω_2 is independent of ω_1 , the distribution $\mathbb{P}_{\theta', \theta}(\omega_2 \mid \omega_1)$ simplifies to $\mathbb{P}_{\theta', \theta}(\omega_2)$. Consequently, $d_{\text{TV}}(\theta, \theta', \omega_1)$ is constant with respect to ω_1 and equals $d_{\text{TV}}(\theta, \theta')$.

As specific examples, Algorithms 1, 2, and 3 all fall under the first category. The newly introduced Locally Balanced PoissonMH (Algorithm 5) belongs to the second category, while TunaMH-SGLD (Algorithm 6) belongs to the third category.

4.3 Spectral gap and Dirichlet form

Theorem 1 is useful for comparing the spectral gaps of \mathbb{P}_{aux} and \mathbb{P}_{MwG} . Let's begin by reviewing some key concepts. Consider P as the transition kernel for a reversible Markov chain with stationary distribution Π . The linear space $L^2(\Pi)$ includes all functions f satisfying $\int f^2(x)\Pi(dx) < \infty$. Within this space, $L_0^2(\Pi)$ is the subspace of functions $f \in L^2(\Pi)$ satisfying $\int f(x)\Pi(dx) = 0$, i.e., all random variables with zero mean and finite variance. It is known that both $L^2(\Pi)$ and $L_0^2(\Pi)$ are Hilbert spaces with the inner product $\langle f, g \rangle_\Pi := \int (fg)d\Pi = \mathbb{E}_\Pi[fg]$. The norm of any element $f \in L^2(\Pi)$ is defined as $\|f\|_\Pi := \sqrt{\langle f, f \rangle_\Pi}$. The Markov transition kernel P operates as a linear operator on $L^2(\Pi)$, acting on f by $(Pf)(x) := \int P(x, dy)f(y)$. Reversibility of P implies that it is an adjoint operator, meaning $\langle Pf, g \rangle_\Pi = \langle f, Pg \rangle_\Pi$. It is also known that P maps $L_0^2(\Pi)$ to $L_0^2(\Pi)$, i.e., $Pf \in L_0^2(\Pi)$ if $f \in L_0^2(\Pi)$. The operator norm of P on $L^2(\Pi)$ (and $L_0^2(\Pi)$, respectively) is defined as $\|P\|_{L^2(\Pi)} := \sup_{f \in L^2(\Pi), \|f\|_\Pi=1} \|Pf\|_\Pi$ (and $\|P\|_{L_0^2(\Pi)} := \sup_{f \in L_0^2(\Pi), \|f\|_\Pi=1} \|Pf\|_\Pi$, respectively). It is known $\|P\|_{L^2(\Pi)} = 1$ (as $P1 = 1$), and $\|P\|_{L_0^2(\Pi)} \leq 1$. The spectral gap of P , denoted as $\text{Gap}(P)$, is defined as $1 - \|P\|_{L_0^2(\Pi)}$. The Dirichlet form of a function f with respect to P is defined as $\mathcal{E}(P, f) := \|(I - P)f, f\|_\Pi = 0.5 \int \int (f(x) - f(y))^2 \Pi(dx)P(x, dy)$. When P is reversible, it is known that $\text{Gap}(P) = \inf_{f \in L_0^2(\Pi), f \neq 0} \mathcal{E}(P, f) / \|f\|_\Pi^2$.

Theorem 1 can be directly used to compare the Dirichlet form of \mathbb{P}_{MwG} and \mathbb{P}_{aux} . The next proposition follows directly from Theorem 1, and therefore we omit the proof.

Proposition 3. *Let $\Pi = \pi(\theta | x)$ be the common stationary distribution of \mathbb{P}_{MwG} and \mathbb{P}_{aux} . Let f be any function in $L_0^2(\Pi)$, we have:*

1. *The Dirichlet form*

$$\begin{aligned} \mathcal{E}(\mathbb{P}_{\text{aux}}, f) &\geq \int \int (1 - d_{\text{TV}}(\theta, \theta')) (f(\theta) - f(\theta'))^2 \mathbb{P}_{\text{MwG}}(\theta, d\theta') \pi(d\theta | x) \\ &\geq \left(1 - \sup_{\theta, \theta'} d_{\text{TV}}(\theta, \theta')\right) \mathcal{E}(\mathbb{P}_{\text{MwG}}, f). \end{aligned}$$

Therefore $\text{Gap}(\mathbb{P}_{\text{aux}}) \geq (1 - \sup_{\theta, \theta'} d_{\text{TV}}(\theta, \theta')) \text{Gap}(\mathbb{P}_{\text{MwG}})$.

2. *Similarly,*

$$\begin{aligned} \mathcal{E}(\mathbb{P}_{\text{aux}}, f) &\geq \int \int e^{-1/e} \exp\{-d_{\text{KL}}(\theta, \theta')\} (f(\theta) - f(\theta'))^2 \mathbb{P}_{\text{MwG}}(\theta, d\theta') \pi(d\theta | x) \\ &\geq \left(e^{-1/e} \inf_{\theta, \theta'} \exp\{-d_{\text{KL}}(\theta, \theta')\}\right) \mathcal{E}(\mathbb{P}_{\text{MwG}}, f). \end{aligned}$$

Therefore $\text{Gap}(\mathbb{P}_{\text{aux}}) \geq (e^{-1/e} \inf_{\theta, \theta'} \exp\{-d_{\text{KL}}(\theta, \theta')\}) \text{Gap}(\mathbb{P}_{\text{MwG}})$.

The spectral gap implies the mixing time of a Markov chain. Consider an initial distribution π_0 that is β -warm with respect to the stationary distribution Π , meaning $\pi_0(A)/\Pi(A) \leq \beta$ for any measurable set A . A Markov chain P starting from a β -warm distribution reaches within ϵ of the stationary distribution in terms of both total-variation and L^2 distance after $\mathcal{O}(\log(\beta\epsilon^{-1})/\text{Gap}(P))$ iterations, as shown in Theorem 2.1 of [60]. Thus, Proposition 3 roughly says that \mathbb{P}_{aux} is at most $1/(1 - \sup_{\theta,\theta'} d_{\text{TV}}(\theta, \theta'))$ times slower than \mathbb{P}_{MwG} in terms of iteration count. Conversely, if the per-iteration cost of \mathbb{P}_{aux} is less than $1/(1 - \sup_{\theta,\theta'} d_{\text{TV}}(\theta, \theta'))$ times the cost of \mathbb{P}_{MwG} , then the \mathbb{P}_{aux} is (provably) more efficient than \mathbb{P}_{MwG} in terms of total cost.

Bounding $\text{Gap}(\mathbb{P}_{\text{MwG}})$: Proposition 3 establishes a lower bound for $\text{Gap}(\mathbb{P}_{\text{aux}})$ in relation to $\text{Gap}(\mathbb{P}_{\text{MwG}})$. The next step is to bound $\text{Gap}(\mathbb{P}_{\text{MwG}})$. In this section, we discuss recent findings from [57] that examine $\text{Gap}(\mathbb{P}_{\text{MwG}})$. In each iteration, the chain \mathbb{P}_{MwG} first draws $\omega_1 \sim \mathbb{P}_\theta(\cdot)$, followed by a Metropolis–Hastings algorithm to sample from $\pi(\theta | \omega_1, x) \propto \pi(\theta | x)\mathbb{P}_\theta(\omega_1)$ using the proposal distribution $q_{\omega_1}(\theta, \cdot)$. We use $\mathbb{P}_{\text{MHG},\omega_1}$ to denote the transition kernel of the second step. We also use $\mathbb{P}_{\text{Gibbs}}$ to denote the “ θ -chain” of the standard Gibbs sampler targeting $\pi(\theta | x)\mathbb{P}_\theta(\omega_1)$. This means we first draw $\omega_1 \sim \mathbb{P}_\theta(\cdot)$, and then draw θ directly from $\pi(\theta | \omega_1, x)$.

Proposition 4 (Translated from Corollary 14 and Theorem 15 of [57]). *We have:*

1. $\text{Gap}(\mathbb{P}_{\text{MwG}}) \geq \text{Gap}(\mathbb{P}_{\text{Gibbs}}) \times \inf_{\omega_1} \text{Gap}(\mathbb{P}_{\text{MHG},\omega_1})$
2. Suppose there is a measurable function $\gamma_1 : \Omega_1 \rightarrow [0, 1]$ satisfying

$$\|\mathbb{P}_{\text{MHG},\omega_1}\|_{L_0^2(\pi(\cdot | \omega_1, x))} \leq \gamma_1(\omega_1)$$

for every ω_1 . Meanwhile

$$\int \gamma_1(\omega_1)^t \mathbb{P}_\theta(d\omega_1) \leq \alpha_t$$

for every θ and some even t . Then

$$\text{Gap}(\mathbb{P}_{\text{MwG}}) \geq \frac{\text{Gap}(\mathbb{P}_{\text{Gibbs}}) - \alpha_t}{t}.$$

The first result is also known in [6]. The second result, which relaxes the assumptions of the first, generalizes the results of [40]. These results have been applied in the analysis of proximal samplers for log-concave target distributions and hybrid slice samplers. See Section 5 of [57] for details.

4.4 Applications to existing algorithms

We will explain how Theorem 1 and Proposition 3 can be applied to study current algorithms.

Exchange algorithm: Since the exchange algorithm (Algorithm 2) has $\omega_1 = \text{Null}$, we know $\mathbb{P}_{\text{MwG}} = \mathbb{P}_{\text{ideal}}$, which is the plain Metropolis–Hastings algorithm. Our next proposition recovers Theorem 5 in [70].

Proposition 5. *Let \mathbb{P}_{aux} be the transition kernel of the exchange algorithm (Algorithm 1), then*

$$p_{\text{aux}}(\theta, \theta') \geq \left(1 - \sup_{\theta, \theta'} d_{\text{TV}}(p_\theta, p_{\theta'})\right) p_{\text{ideal}}(\theta, \theta').$$

In particular, let $A_{\theta, \theta'}(s) := \{\omega : p_\theta(\omega) > s p_{\theta'}(\omega)\}$. If there exists $\epsilon, \delta \in (0, 1)$ such that $\mathbb{P}_{p_{\theta'}}(A_{\theta, \theta'}(\delta)) > \epsilon$ uniformly over θ, θ' , then $1 - \sup_{\theta, \theta'} d_{\text{TV}}(p_\theta, p_{\theta'}) \geq \epsilon\delta$, thus $\text{Gap}(\mathbb{P}_{\text{aux}}) \geq \epsilon\delta \text{Gap}(\mathbb{P}_{\text{ideal}})$.

Proof of Proposition 5. Our first result is derived directly by recognizing that $\mathbb{P}_{\theta, \theta'}$ within our general framework corresponds to $p_{\theta'}$ in the exchange algorithm. The second result follows from the observation:

$$1 - d_{\text{TV}}(p_\theta, p_{\theta'}) = \mathbb{E}_{p_{\theta'}} \left[\min \left\{ 1, \frac{p_\theta}{p_{\theta'}} \right\} \right] \geq \delta \mathbb{P}_{p_{\theta'}}(A_{\theta, \theta'}(\delta)) = \epsilon\delta.$$

□

PoissonMH: It is useful to recall a few facts, which we provide proofs for completeness in Appendix A.4: for univariate Poisson random variables with parameter λ_1, λ_2 , their KL divergence equals $d_{\text{KL}}(\text{Poi}(\lambda_1) \parallel \text{Poi}(\lambda_2)) = \lambda_1 \log(\lambda_1/\lambda_2) + \lambda_2 - \lambda_1$. Meanwhile, the KL divergence $d_{\text{KL}}(\text{Poi}(\lambda_1) \parallel \text{Poi}(\lambda_2))$ satisfying $m \leq \lambda_1 \leq \lambda_2 \leq M$ is maximized when $\lambda_1 = m$ and $\lambda_2 = M$. Furthermore, KL divergence satisfies the ‘tensorization property’, i.e., $d_{\text{KL}}(\otimes_i P_i \parallel \otimes_i Q_i) = \sum_i d_{\text{KL}}(P_i \parallel Q_i)$, while $1 - d_{\text{TV}}(\otimes_i P_i, \otimes_i Q_i) \geq \prod_i (1 - d_{\text{TV}}(P_i, Q_i))$. Using Proposition 3 allows us to give a (arguably) simpler proof for the theoretical analysis of PoissonMH, with slightly stronger results.

Proposition 6. *With all the notations used in PoissonMH (Algorithm 2), let \mathbb{P}_{aux} be the transition kernel of Algorithm 2. Then we have $\text{Gap}(\mathbb{P}_{\text{aux}}) \geq e^{-1/e} \exp \left\{ -\frac{L^2}{\max\{\lambda+L, 2\lambda\}} \right\} \text{Gap}(\mathbb{P}_{\text{ideal}})$.*

Proof of Proposition 6. Let $\tilde{\lambda}_i(\theta) := \lambda M_i / L + \phi_i(\theta; x)$. We recognize that $\mathbb{P}_{\theta, \theta'} = \otimes_{i=1}^N \text{Poi}(\tilde{\lambda}_i(\theta))$ in PoissonMH. We have $1 - d_{\text{TV}}(\mathbb{P}_{\theta, \theta'}, \mathbb{P}_{\theta', \theta}) \geq \prod_i \left(1 - d_{\text{TV}}(\text{Poi}(\tilde{\lambda}_i(\theta)), \text{Poi}(\tilde{\lambda}_i(\theta')))\right)$. For each

term in the product, assuming without loss of generality that $\tilde{\lambda}_i(\theta') > \tilde{\lambda}_i(\theta)$:

$$1 - d_{\text{TV}}(\text{Poi}(\tilde{\lambda}_i(\theta)), \text{Poi}(\tilde{\lambda}_i(\theta'))) \geq e^{-1/e} \exp \left\{ -d_{\text{KL}} \left(\text{Poi}(\tilde{\lambda}_i(\theta)) \parallel \text{Poi}(\tilde{\lambda}_i(\theta')) \right) \right\}.$$

The KL divergence satisfies:

$$\begin{aligned} d_{\text{KL}} \left(\text{Poi}(\tilde{\lambda}_i(\theta)) \parallel \text{Poi}(\tilde{\lambda}_i(\theta')) \right) &\leq \tilde{\lambda}_i(\theta) \log \left(\tilde{\lambda}_i(\theta)/\tilde{\lambda}_i(\theta') \right) + \tilde{\lambda}_i(\theta') - \tilde{\lambda}_i(\theta) \\ &\leq (\lambda M_i/L) \frac{-M_i}{\lambda M_i/L + M_i} + M_i = \frac{LM_i}{\lambda + L}. \end{aligned}$$

Here the second inequality uses the fact $\tilde{\lambda}_i(\theta) \in [\lambda M_i/L, \lambda M_i/L + M_i]$ for every θ . Putting these together shows

$$1 - d_{\text{TV}}(\mathbb{P}_{\theta, \theta'}, \mathbb{P}_{\theta', \theta}) \geq e^{-1/e} \exp \left\{ - \sum_i \frac{LM_i}{\lambda + L} \right\} = e^{-1/e} \exp \left\{ - \frac{L^2}{\lambda + L} \right\}.$$

Thus $\text{Gap}(\mathbb{P}_{\text{aux}}) \geq e^{-1/e} \exp \left\{ - \frac{L^2}{\lambda + L} \right\} \text{Gap}(\mathbb{P}_{\text{ideal}})$.

We can also use the second part of Proposition 3, and calculate the symmetrized KL:

$$\begin{aligned} &d_{\text{KL}}(\mathbb{P}_{\theta, \theta'} \parallel \mathbb{P}_{\theta', \theta}) + d_{\text{KL}}(\mathbb{P}_{\theta', \theta} \parallel \mathbb{P}_{\theta, \theta'}) \\ &= \sum_{i=1}^N \left(\tilde{\lambda}_i(\theta) \log \frac{\tilde{\lambda}_i(\theta)}{\tilde{\lambda}_i(\theta')} + \cancel{\tilde{\lambda}_i(\theta')} - \cancel{\tilde{\lambda}_i(\theta)} \right) + \sum_{i=1}^N \left(\tilde{\lambda}_i(\theta') \log \frac{\tilde{\lambda}_i(\theta')}{\tilde{\lambda}_i(\theta)} + \cancel{\tilde{\lambda}_i(\theta)} - \cancel{\tilde{\lambda}_i(\theta')} \right) \\ &\leq \log \left(1 + \frac{L}{\lambda} \right) \sum_{i=1}^N |\tilde{\lambda}_i(\theta') - \tilde{\lambda}_i(\theta)| \leq \log \left(1 + \frac{L}{\lambda} \right) \sum_{i=1}^N M_i = L \log \left(1 + \frac{L}{\lambda} \right). \end{aligned}$$

Therefore, applying Proposition 3, we get:

$$\text{Gap}(\mathbb{P}_{\text{aux}}) \geq e^{-1/e} \left(1 + \frac{L}{\lambda} \right)^{-L/2} \text{Gap}(\mathbb{P}_{\text{ideal}}) \geq e^{-1/e} \exp \left\{ \frac{-L^2}{2\lambda} \right\} \text{Gap}(\mathbb{P}_{\text{ideal}}).$$

□

In Theorem 2 of the original paper [76], the authors establish the bound $\text{Gap}(\mathbb{P}_{\text{aux}}) \geq 0.5 \exp \left\{ \frac{-L^2}{\lambda + L} \right\} \text{Gap}(\mathbb{P}_{\text{ideal}})$ as a theoretical guarantee for the PoissonMH algorithm. Our exponent term $\frac{-L^2}{\max\{\lambda + L, 2\lambda\}}$ is always at least as good as the original bound, and is strictly sharper if $\lambda > L$. For example, the suggested choice of λ is of the order L^2 (with L being approximately of the order N) in the original paper, which satisfies this requirement. Our constant $e^{-1/e} \approx 0.69$ is also slightly sharper compared to their 0.5.

TunaMH: As another application of Proposition 3, we provide a (again, arguably) simpler proof for TunaMH, achieving stronger results. The proof is in Appendix A.3

Proposition 7. *With all the notations used in TunaMH (Algorithm 3), let \mathbb{P}_{aux} be the transition kernel of Algorithm 3. Then we have $\text{Gap}(\mathbb{P}_{\text{aux}}) \geq e^{-1/e} \exp\left\{-\frac{1}{2\chi}\right\} \text{Gap}(\mathbb{P}_{\text{ideal}})$.*

Theorem 2 of [75] establishes the bound $\text{Gap}(\mathbb{P}_{\text{aux}}) \geq \exp\{-1/\chi - 2\sqrt{(\log 2)/\chi}\} \text{Gap}(\mathbb{P}_{\text{ideal}})$. Hence, our rate $\exp\{-1/2\chi\}$ is strictly sharper than the existing rate $\exp\{-1/\chi - 2\sqrt{(\log 2)/\chi}\}$, with the improvement becoming more significant as χ decreases. Although our constant $e^{-1/e}$ is slightly worse than the original constant by a factor of 30%, it is important to note that the tuning parameter χ , which controls the expected batch size, is often chosen to be a very small constant (e.g., 10^{-4} or 10^{-5} as discussed in Section 5 of [75]). In those cases, our bound is approximately $\exp(10^4)$ or $\exp(10^5)$ times larger than the previous bound.

4.5 Future directions: from algorithms to a more powerful comparison theory, and back

As demonstrated in Section 4.4, Proposition 3 provides useful results for existing algorithms. However, it still has significant limitations due to its assumptions. Proposition 3 provides non-trivial lower bounds on the spectral gap only when $\sup_{\theta, \theta'} d_{\text{TV}}(\theta, \theta') < 1$. This condition is satisfied for PoissonMH and TunaMH (implicitly implied by their assumptions), but it may not hold in many practical situations. On the other hand, the ‘pointwise’ comparison (Theorem 1) is much more robust. Therefore, an appealing direction of research is to establish a weaker version of Proposition 3 but under relaxed assumptions. Promising approaches towards this goal include the use of approximate conductance [28, 9] and weak Poincaré inequalities [4, 5]. Both methods have recently demonstrated successful applications in the analysis of MCMC algorithms. For example, weak Poincaré inequalities can provide useful bounds on the mixing time for pseudo-marginal MCMC when the spectral gap of the idealized chain is either unknown or nonexistent.

If a more powerful comparison theory can be established, it could also lead to the development of new algorithms with theoretical guarantees. As demonstrated in Section 4.4, the existing assumptions in [76] and [75] both implicitly imply the (potentially stringent) condition $\sup_{\theta, \theta'} d_{\text{TV}}(\theta, \theta') < 1$. Therefore, it is reasonable to conjecture that developing a new theory with relaxed assumptions might lead to new algorithms with broader applicability.

5 Numerical experiments

This section presents numerical experiments to compare our gradient-based minibatch algorithms with existing methods. In the first two experiments, we examine two simulated examples: heterogeneous truncated Gaussian and robust linear regression. We compare our Poisson-Barker and Poisson-MALA algorithms with several alternatives, including their predecessor PoissonMH and generic full batch algorithms like random-walk Metropolis, MALA, and HMC. In the third experiment, we consider Bayesian logistic regression on the MNIST handwritten digits dataset. We compare TunaMH-SGLD with TunaMH, random-walk Metropolis, and MALA. Our experiments suggest that minibatch algorithms generally outperform full batch algorithms when normalized by wall-clock time. In particular, our gradient-based minibatch method significantly improves existing minibatch methods.

We implement all methods in **Julia**. The codes to reproduce the experimental results can be found at <https://github.com/ywwes26/MCMC-Auxiliary>. In all the figures in this section, the random-walk Metropolis is labeled as MH, while all other methods are labeled by their respective names. Different minibatch algorithms require prior knowledge of different bounds on the model likelihood (refer to the “promise” block in Algorithm 2 and 3 for details). They are derived in Appendix C.

5.1 Heterogeneous truncated Gaussian

We begin with an illustration example. Suppose $\theta \in \mathbb{R}^d$ is our parameter of interest, the data $\{y_i\}_{i=1}^N$ is generated i.i.d. from $y_i \mid \theta \sim \mathcal{N}(\theta, \Sigma)$ with $\theta = (0, 0, \dots, 0)$. In our experiment, we set $d = 20$, $N = 10^5$, $\beta = 10^{-5}$ and $\Sigma = \text{Diagonal}(1, 0.95, 0.90, \dots, 0.05)$. Therefore, our data follows a multidimensional heterogeneous Gaussian, a common setting in sampling tasks such as [51]. Following [76, 75], we truncate the parameter θ to a hypercube $[-3, 3]^d$ and apply a flat prior. As in [62, 75], we aim to sample from the tempered posterior: $\pi(\theta \mid \{y_i\}_{i=1}^N) \propto \exp\left\{-\frac{1}{2}\beta \sum_{i=1}^N (\theta - y_i)^\top \Sigma^{-1} (\theta - y_i)\right\}$ with $\beta = 10^{-5}$.

We test PoissonMH, Poisson-MALA, Poisson-Barker, (full batch) random-walk Metropolis, and (full batch) MALA in this problem. Following the suggestion in [76], the hyperparameter λ for all three minibatch algorithms is set to $\lambda = 0.0005L^2$, resulting in a batch size of around 6000 (6% of the data points). The initial θ_0 is drawn from $\mathcal{N}(0, \mathbb{I}_d)$, and the step size of each algorithm is tuned through several pilot runs to achieve target acceptance rates of 0.25, 0.4, and 0.55. We use the following two metrics to compare the methods:

1. Mean Squared Error (MSE) in estimating the posterior mean and posterior variance.
The true values can be calculated numerically.

2. Effective Sample Size (ESS), a metric using autocorrelation to quantify the dependence of MCMC samples. In an MCMC run, each dimension corresponds to an ESS. We record and report the minimum, median, and maximum ESS across all dimensions for all methods.

We compare the clock-time performances of all five methods mentioned above. Figure 2 reports the MSE of estimating posterior mean/variance as a function of time, across different acceptance rates. We have three interesting findings:

1. Poisson-Barker significantly improves the convergence speed of PoissonMH, random-walk Metropolis, and MALA across all acceptance rates, both in terms of posterior mean and variance. The improvement seems to be more notable when the acceptance rate increases.
2. All three minibatch algorithms (PoissonMH, Poisson-MALA, Poisson-Barker) consistently outperform random-walk Metropolis and MALA, demonstrating the effectiveness of these minibatch MCMC algorithms in the tall data regime.
3. Poisson-MALA achieves nearly the same best performance as Poisson-Barker at acceptance rates of 0.4 and 0.55. However, in the low acceptance rate (0.25) regime, Poisson-MALA has some fluctuations, with overall performance slightly worse than PoissonMH. This aligns with the intuition in [44] that MALA can be sensitive to tuning parameters, while Barker’s proposal is more robust.

Table 1 presents a comparison of the ESS per second (ESS/s) across various acceptance rates. Our gradient-based minibatch methods (Poisson-Barker and Poisson-MALA) consistently rank as the top two methods across all combinations of acceptance rates: (0.25, 0.4, 0.55) and metrics: (min, median, max) ESS/s. They have demonstrated an improvement of 1.24 – 7.02 times over PoissonMH, 7.61 – 12.33 times over MALA, and 13.00 – 67.24 times over random-walk Metropolis. At a 0.25 acceptance rate, Poisson-Barker has the highest performance, closely followed by Poisson-MALA. At a 0.55 acceptance rate, Poisson-MALA achieves the best performance, which is also the highest overall performance when all methods are optimally tuned. At a 0.4 acceptance rate, Poisson-Barker and Poisson-MALA are comparable. Therefore, despite the additional evaluations for approximating the gradient at each step, our gradient-based minibatch algorithms have demonstrated substantial benefits from selecting a more efficient proposal.

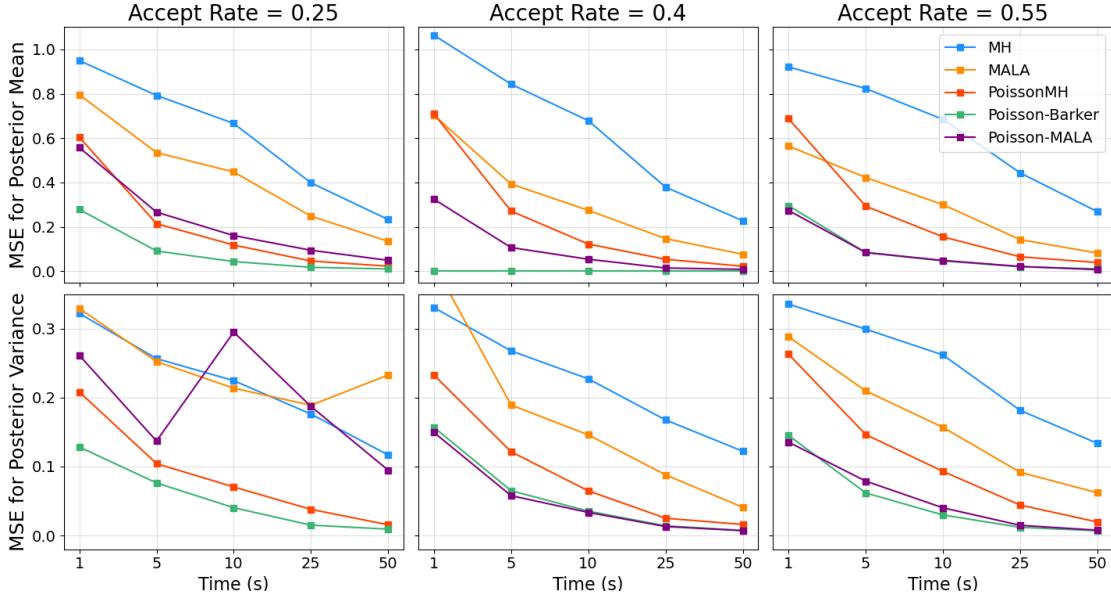


Figure 2: Clock-wise MSE comparison of random-walk Metropolis, MALA, PoissonMH, Poisson-Barker and Poisson-MALA. First row: MSE of posterior mean as a function of time for different acceptance rates; Second row: MSE of posterior variance as a function of time for different acceptance rates. The results are averaged over 10 runs for all five methods.

Method	ESS/s: (Min, Median, Max)				Best
	acceptance rate=0.25	acceptance rate=0.4	acceptance rate=0.55		
MH	(0.04, 0.06, 0.40)	(0.03, 0.04, 0.41)	(0.03, 0.04, 0.33)		(0.04, 0.06, 0.41)
MALA	(0.06, 0.11, 0.68)	(0.09, 0.18, 1.57)	(0.09, 0.15, 2.57)		(0.09, 0.18, 2.57)
PoissonMH	(0.37, 0.64, 4.55)	(0.31, 0.48, 3.99)	(0.25, 0.39, 3.16)		(0.37, 0.64, 4.55)
Poisson-Barker	(0.74, 1.32, 5.77)	(0.74, 1.40, 8.38)	(0.79, 1.45, 10.68)		(0.79, 1.45, 10.68)
Poisson-MALA	(0.52, 1.09, 5.64)	(0.73, 1.37, 12.87)	(0.81, 1.53, 22.19)		(0.81, 1.53, 22.19)

Table 1: ESS/s of random-walk Metropolis, MALA, PoissonMH, Poisson-Barker and Poisson-MALA for the heterogeneous truncated Gaussian experiment. Here (Min, Median, Max) refer to the minimum, median and maximum of ESS/s across all dimensions. The column ‘‘Best’’ reports the best ESS/s across all acceptance rates. The results are averaged over 10 runs for all five methods.

5.2 Robust linear regression

We then compare Poisson-Barker, Poisson-MALA, PoissonMH, random-walk Metropolis, MALA, and HMC in a robust linear regression example, which is also considered in [23, 75, 46]. The data is generated following the simulation settings in [23, 75]. Specifically, for $i = 1, 2, \dots, N$, the covariates $x_i \in \mathbb{R}^d$ are independently generated from $\mathbb{N}(0, \mathbb{I}_d)$ and $y_i = \sum_{j=1}^d x_{ij} + \epsilon_i$ with $\epsilon_i \sim \mathbb{N}(0, 1)$. The likelihood is modeled as $p(y_i | \theta, x_i) = \text{Student}(y_i - \theta^\top x_i | v)$, where $\text{Student}(\cdot | v)$ is the density of a Student’s t distribution with v degrees of freedom. The model is ‘‘robust’’ because the likelihood function has a heavier tail [46]. Unlike

the Truncated Gaussian example, we cannot easily estimate the statistics of the posterior, so we must use MCMC. With a flat prior, the tempered posterior will be:

$$\pi(\theta \mid \{y_i, x_i\}_{i=1}^N) \propto \exp \left\{ -\beta \cdot \frac{v+1}{2} \sum_{i=1}^N \log \left(1 + \frac{(y_i - \theta^\top x_i)^2}{v} \right) \right\}$$

When the data is appropriately re-scaled (e.g., each covariate has a mean of 0 and variance of 1), the regression coefficient is typically restricted to a certain range. Therefore, we follow the common practice [31] of constraining the coefficients within a high-dimensional sphere $\{\theta \in \mathbb{R}^d \mid \|\theta\|_2 \leq R\}$, which has a similar effect to l^2 -regularization.

We use $v = 4$ as in [23, 75] and set $d = 10$, $N = 10^5$, $\beta = 10^{-4}$, $R = 15$, and test all six methods mentioned above. PoissonMH, Poisson-MALA and Poisson-Barker use $\lambda = 0.01L^2$. Following common practice, the number of leapfrog steps in HMC is set to 10. The initial θ_0 is drawn from $\mathcal{N}(0, \mathbb{I}_d)$, and the step size of each algorithm is tuned to achieve acceptance rates of 0.25, 0.4, and 0.55. Our true parameter is $\theta^* = (1, 1, \dots, 1)$.

The convergence speed comparison of all methods is summarized in Figure 3. By plotting the MSE of estimating the true coefficients as a function of time, we can observe that Poisson-MALA and Poisson-Barker are much more efficient than PoissonMH, and all of them outperform full-batch methods. Poisson-MALA and Poisson-Barker adapt to the posterior a lot faster than PoissonMH at an early stage due to the benefit of gradient information. For instance, in part (a) of Figure 3, the MSE of PoissonMH is approximately 1.0 at 0.025 seconds, whereas the MSE of Poisson-MALA and Poisson-Barker is around 0.2 at the same time. Full-batch methods experience slow adaptation over time because they require a complete scan of the entire dataset, despite having a faster convergence rate per iteration. Full-batch gradient-based methods overall perform better than the random-walk Metropolis due to the benefit of gradient information, when their acceptance rates are tuned properly. It should also be noted that HMC performs worse than MALA and sometimes even worse than random-walk Metropolis, as each iteration involves 10 leapfrog steps, making it approximately 20 times more costly than the other two. To compare the per-step convergence rate of each method, we also present their MSE as a function of the number of iterations in Figure 4. Our results confirm our intuition that random-walk Metropolis outperforms PoissonMH per step, and MALA outperforms both Poisson-MALA and PoissonMH.

We also present the ESS/s comparisons in Table 2. The results have a similar trend to those discussed in 5.1. Poisson-MALA and Poisson-Barker are the top two methods across all metrics, showing improvements ranging from 2.11 to 8.23 times over PoissonMH, and achieving nearly or more than 100 times the performance of full-batch methods.

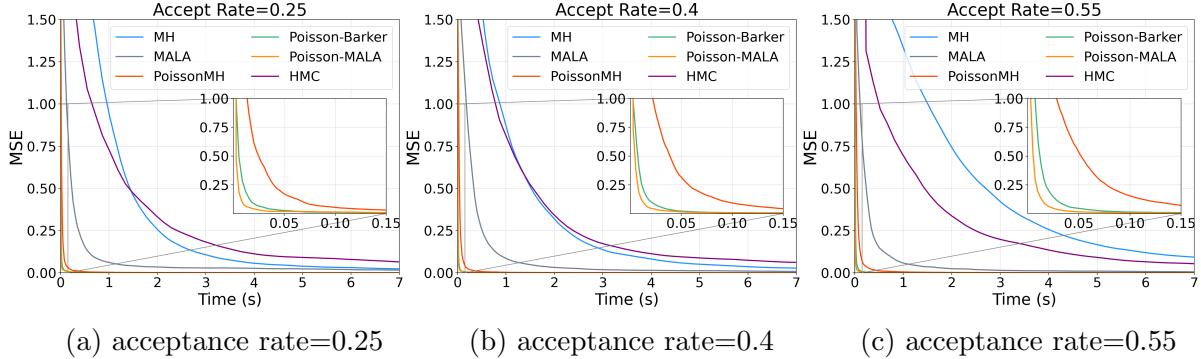


Figure 3: Estimating θ^* in Robust Linear Regression: MSE as a function of time across different acceptance rates. The 3 large plots report the performances of all methods in the first 7 seconds. The 3 inside plots are zoomed-in proportions of the large plots, focusing on the comparison of PoissonMH, Poisson-Barker and Poisson-MALA in the first 0.15 seconds. All results are averaged over 10 independent repeated simulations.

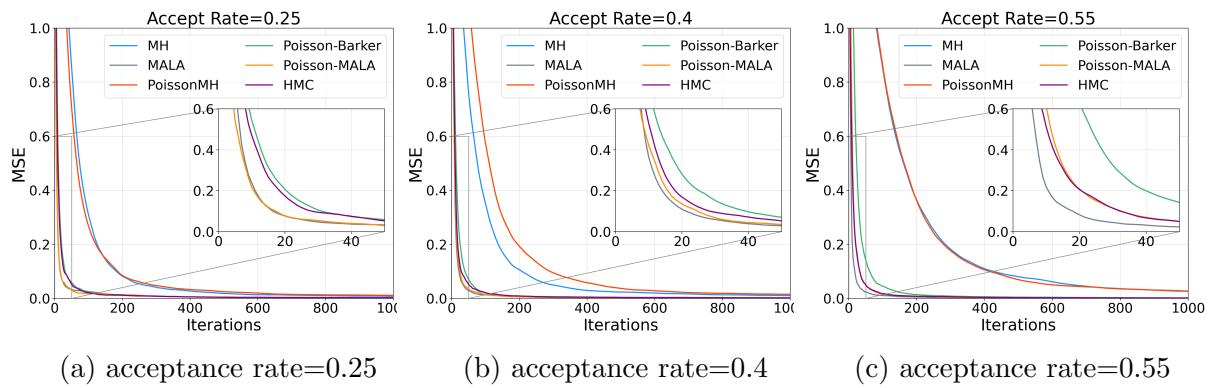


Figure 4: Estimating θ^* in Robust Linear Regression: MSE as a function of the number of iterations across different acceptance rates. The 3 large plots report the performances of all methods in the first 1000 steps. The 3 inside plots are zoomed-in proportions of the large plots, focusing on the comparison of Poisson-Barker, Poisson-MALA, MALA and HMC in the first 50 steps. All results are averaged over 10 independent repeated simulations.

5.3 Bayesian Logistic Regression

Finally, we apply TunaMH and TunaMH-SGLD to a Bayesian logistic regression task using the MNIST handwritten digits dataset. We focus on classifying handwritten 3s and 5s, which can often appear visually similar. The training dataset consists of 11,552 samples, while the test dataset contains 1,902 samples. Figure 5 shows some examples of the original data. Additional experiments for classifying 7s and 9s and the results can be found in Appendix D.1. Similar to [72, 75], we use the first 50 principal components of the 28 by 28 features as covariates.

Method	ESS/s: (Min, Median, Max)			
	acceptance rate=0.25	acceptance rate=0.4	acceptance rate=0.55	Best
MH	(1.7, 1.9, 2.3)	(1.39, 1.68, 1.85)	(1.1, 1.2, 1.4)	(1.7, 1.9, 2.3)
MALA	(2.1, 2.3, 2.5)	(3.88, 4.19, 4.65)	(4.9, 5.4, 5.8)	(4.9, 5.4, 5.8)
HMC	(0.7, 0.8, 0.9)	(0.87, 0.95, 0.99)	(1.0, 1.0, 1.1)	(1.0, 1.0, 1.1)
PoissonMH	(81.3, 86.0, 90.5)	(79.6, 81.3, 84.1)	(56.2, 57.5, 60.1)	(81.3, 86.0, 90.5)
PoissonMH-Barker	(175.3, 184.3, 191.4)	(234.0, 240.6, 249.6)	(257.9, 263.3, 271.7)	(257.9, 263.3, 271.7)
PoissonMH-MALA	(214.2, 220.7, 226.8)	(363.9, 371.4, 381.9)	(458.6, 473.0, 481.7)	(458.6, 473.0, 481.7)

Table 2: ESS/s of random-walk Metropolis, MALA, HMC, PoissonMH, Poisson-Barker and Poisson-MALA for the robust linear regression experiment. Here (Min, Median, Max) refer to the minimum, median and maximum of ESS/s across all dimensions. The column “Best” reports the best ESS/s across all acceptance rates. The results are averaged over 10 runs for all six methods.



Figure 5

For TunaMH and TunaMH-SGLD, we follow [75] and set the hyperparameter $\chi = 10^{-5}$. The batch size for estimating the gradient in TunaMH-SGLD is set as $K = 20$, so the computation overhead is minimal. We also clip the gradient norm to 2. The original study [75] uses a step size of 0.0008 for TunaMH and 0.004 for the random-walk Metropolis algorithm. To provide a comprehensive comparison, we test various step sizes $\{0.0008, 0.001, 0.002, 0.004\}$ across all algorithms.

The test accuracy results over time are presented in Figure 6. TunaMH-SGLD improves the test accuracy of TunaMH across all step size configurations, particularly with larger step sizes. Both TunaMH-SGLD and TunaMH significantly outperform full batch methods, similar to the trends observed in Sections 5.1 and 5.2. By utilizing an additional minibatch for estimating the gradient, TunaMH-SGLD converges to the posterior faster in clock time.

For completeness, we also report the test errors as a function of the number of iterations in Figure 7.

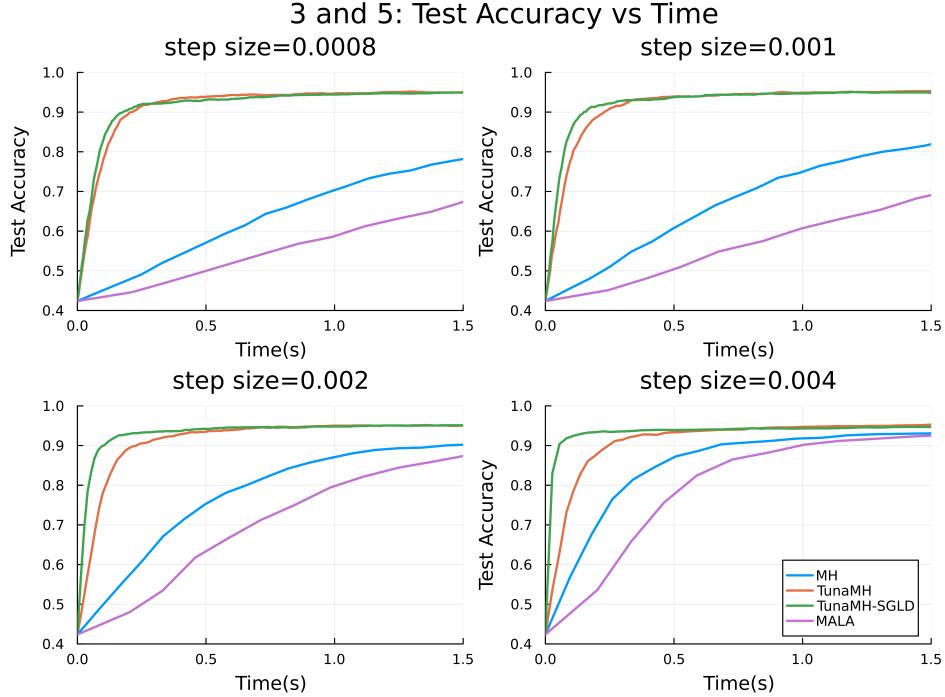


Figure 6: Test accuracy as a function of time for classifying 3s and 5s in MNIST. Step sizes for all methods are varied across $\{0.0008, 0.001, 0.002, 0.004\}$. Each subplot reports the curve of test accuracy in the first 1.5 seconds.

6 Future directions

Our framework opens new directions for the development of MCMC algorithms. When considering minibatch MCMC algorithms, it is crucial for future research to expand the technical assumptions for posterior distributions beyond i.i.d. data. Under i.i.d. assumptions, the cost per iteration scales linearly with the minibatch size. Therefore a minibatch containing 5% of the data results in approximately 5% of the full batch cost per step. However, for non-i.i.d. models, evaluating the unnormalized full-posterior often scales quadratically or cubically with the sample size, especially when determinant evaluation is necessary. Therefore, evaluating the likelihood over a minibatch with 5% of the data may cost only 0.25% or less compared to full batch methods. Thus, designing minibatch MCMC algorithms for these cases, such as spatial models and independent component analysis, becomes more urgent.

Another intriguing question is understanding the source of the performance gains in minibatch MCMC algorithms, including both our methods and existing ones. Empirically,

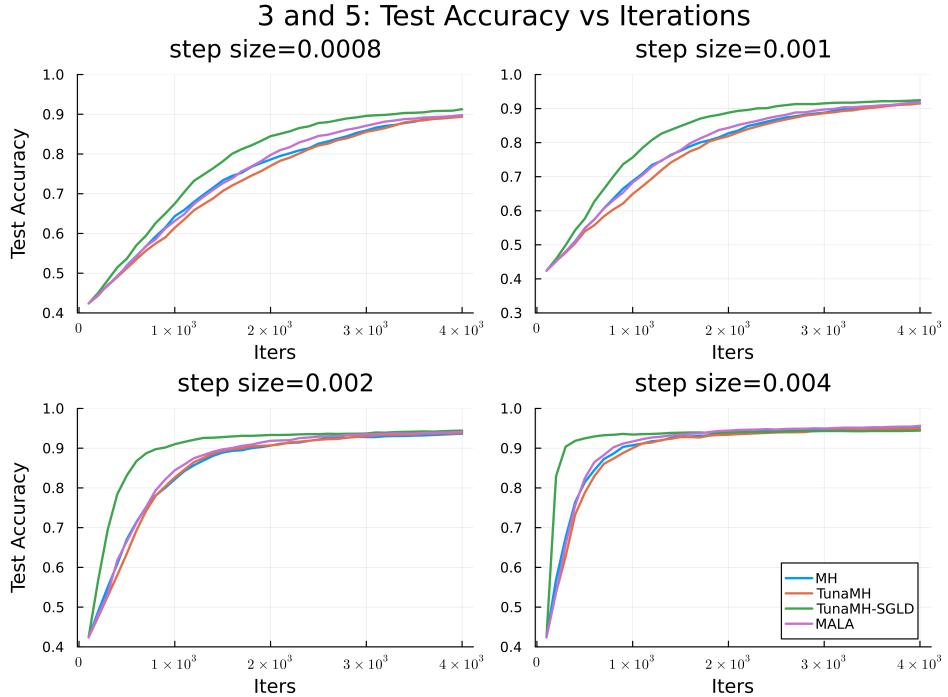


Figure 7: Test accuracy as a function of the number of iterations for classifying 3s and 5s in MNIST. Step sizes for methods are varied across $\{0.0008, 0.001, 0.002, 0.004\}$. Each subplot reports the curve of test accuracy in the first 4,000 steps.

these algorithms demonstrate significant improvements, often achieving ESS/s values 10-100 times greater than those of MCMC algorithms running on the entire dataset. However, it has been proven in [38] that several minibatch algorithms, including [46] but not those considered in this paper, will not significantly improve performance for realistic problems such as generalized linear models. Understanding this discrepancy may also be closely related to the concept of “coresets,” as discussed in [36, 19, 20].

Acknowledgement

The second author was partially supported by the National Science Foundation through grant DMS-2210849 and an Adobe Data Science Research Award. The authors thank Qian Qin and Pierre Jacob for helpful discussions.

References

- [1] Sungjin Ahn. *Stochastic gradient MCMC: Algorithms and applications*. University of California, Irvine, 2015.
- [2] Pierre Alquier, Nial Friel, Richard Everitt, and Aidan Boland. Noisy monte carlo: Convergence of markov chains with approximate transition kernels. *Statistics and Computing*, 26(1-2):29–47, 2016.
- [3] Christophe Andrieu, Arnaud Doucet, Sinan Yıldırım, and Nicolas Chopin. On the utility of Metropolis-Hastings with asymmetric acceptance ratio. *arXiv preprint arXiv:1803.09527*, 2018.
- [4] Christophe Andrieu, Anthony Lee, Sam Power, and Andi Q Wang. Comparison of Markov chains via weak Poincaré inequalities with application to pseudo-marginal MCMC. *The Annals of Statistics*, 50(6):3592–3618, 2022.
- [5] Christophe Andrieu, Anthony Lee, Sam Power, and Andi Q Wang. Weak Poincaré Inequalities for Markov chains: theory and applications. *arXiv preprint arXiv:2312.11689*, 2023.
- [6] Christophe Andrieu, Anthony Lee, and Matti Vihola. Uniform ergodicity of the iterated conditional SMC and geometric ergodicity of particle Gibbs samplers. *Bernoulli*, 24(2), 2018.
- [7] Christophe Andrieu and Gareth O Roberts. The Pseudo-Marginal Approach for Efficient Monte Carlo Computations. *The Annals of Statistics*, pages 697–725, 2009.
- [8] Christophe Andrieu and Matti Vihola. Convergence properties of pseudo-marginal Markov chain Monte Carlo algorithms. *The Annals of Applied Probability*, 25(2):1030–1077, 2015.
- [9] Filippo Ascolani, Gareth O Roberts, and Giacomo Zanella. Scalability of Metropolis-within-Gibbs schemes for high-dimensional Bayesian models. *arXiv preprint arXiv:2403.09416*, 2024.
- [10] Rémi Bardenet, Arnaud Doucet, and Chris Holmes. On Markov chain Monte Carlo methods for tall data. *Journal of Machine Learning Research*, 18(47):1–43, 2017.
- [11] Anthony Alfred Barker. Monte Carlo calculations of the radial distribution functions for a proton electron plasma. *Australian Journal of Physics*, 18(2):119–134, 1965.

- [12] Mark A Beaumont. Estimation of population growth or decline in genetically monitored populations. *Genetics*, 164(3):1139–1160, 2003.
- [13] Julian Besag. Spatial interaction and the statistical analysis of lattice systems. *Journal of the Royal Statistical Society: Series B (Methodological)*, 36(2):192–225, 1974.
- [14] Julian Besag. On the statistical analysis of dirty pictures. *Journal of the Royal Statistical Society Series B: Statistical Methodology*, 48(3):259–279, 1986.
- [15] Julian Besag. Comments on “Representations of knowledge in complex systems” by U. Grenander and MI Miller. *J. Roy. Statist. Soc. Ser. B*, 56(591-592):4, 1994.
- [16] GNJC Bierkens, Paul Fearnhead, and Gareth Roberts. The Zig-Zag process and super-efficient sampling for Bayesian analysis of big data. *Annals of Statistics*, 47(3), 2019.
- [17] Alexandre Bouchard-Côté, Sebastian J Vollmer, and Arnaud Doucet. The bouncy particle sampler: A nonreversible rejection-free Markov chain Monte Carlo method. *Journal of the American Statistical Association*, 113(522):855–867, 2018.
- [18] Jean Bretagnolle and Catherine Huber. Estimation des densités: risque minimax. *Zeitschrift für Wahrscheinlichkeitstheorie und verwandte Gebiete*, 47:119–137, 1979.
- [19] Trevor Campbell and Tamara Broderick. Bayesian coreset construction via greedy iterative geodesic ascent. In *International Conference on Machine Learning*, pages 698–706. PMLR, 2018.
- [20] Trevor Campbell and Tamara Broderick. Automated scalable bayesian inference via hilbert coresets. *Journal of Machine Learning Research*, 20(15):1–38, 2019.
- [21] Clément L Canonne. A short note on an inequality between kl and tv. *arXiv preprint arXiv:2202.07198*, 2022.
- [22] Tianqi Chen, Emily Fox, and Carlos Guestrin. Stochastic gradient hamiltonian monte carlo. In *International Conference on Machine Learning*, pages 1683–1691. PMLR, 2014.
- [23] Rob Cornish, Paul Vanetti, Alexandre Bouchard-Côté, George Deligiannidis, and Arnaud Doucet. Scalable metropolis-hastings for exact bayesian inference with large datasets. In *International Conference on Machine Learning*, pages 1351–1360. PMLR, 2019.

- [24] Khue-Dung Dang, Matias Quiroz, Robert Kohn, Minh-Ngoc Tran, and Mattias Villani. Hamiltonian Monte Carlo with energy conserving subsampling. *Journal of Machine Learning Research*, 20(100):1–31, 2019.
- [25] Persi Diaconis and Laurent Saloff-Coste. Comparison theorems for reversible Markov chains. *The Annals of Applied Probability*, 3(3):696–730, 1993.
- [26] Persi Diaconis and Guanyang Wang. Bayesian goodness of fit tests: a conversation for david mumford. *Annals of Mathematical Sciences and Applications*, 3(1):287–308, 2018.
- [27] Simon Duane, Anthony D Kennedy, Brian J Pendleton, and Duncan Roweth. Hybrid monte carlo. *Physics letters B*, 195(2):216–222, 1987.
- [28] Raaz Dwivedi, Yuansi Chen, Martin J Wainwright, and Bin Yu. Log-concave sampling: Metropolis-Hastings algorithms are fast. *Journal of Machine Learning Research*, 20(183):1–42, 2019.
- [29] Martin Dyer, Leslie Ann Goldberg, Mark Jerrum, and Russell Martin. Markov chain comparison. *Probability Surveys*, 3:89–111, 2006.
- [30] Paul Fearnhead, Omiros Papaspiliopoulos, Gareth O Roberts, and Andrew Stuart. Random-weight particle filtering of continuous time processes. *Journal of the Royal Statistical Society Series B: Statistical Methodology*, 72(4):497–512, 2010.
- [31] Andrew Gelman, Aleks Jakulin, Maria Grazia Pittau, and Yu-Sung Su. A weakly informative default prior distribution for logistic and other regression models. *The Annals of Applied Statistics*, 2(4):1360 – 1383, 2008.
- [32] Sébastien Gerchinovitz, Pierre Ménard, and Gilles Stoltz. Fano’s inequality for random variables. *Statistical Science*, 35(2):178–201, 2020.
- [33] WK Hastings. Monte Carlo sampling methods using Markov chains and their applications. *Biometrika*, 57(1):97–109, 1970.
- [34] Peter D Hoff. Simulation of the matrix bingham–von mises–fisher distribution, with applications to multivariate and relational data. *Journal of Computational and Graphical Statistics*, 18(2):438–456, 2009.
- [35] Zaijing Huang and Andrew Gelman. Sampling for bayesian computation with large datasets. *Available at SSRN 1010107*, 2005.

- [36] Jonathan Huggins, Trevor Campbell, and Tamara Broderick. Coresets for scalable Bayesian logistic regression. *Advances in neural information processing systems*, 29, 2016.
- [37] Pierre E Jacob and Alexandre H Thiery. On nonnegative unbiased estimators. *The Annals of Statistics*, pages 769–784, 2015.
- [38] James E Johndrow, Natesh S Pillai, and Aaron Smith. No free lunch for approximate mcmc. *arXiv preprint arXiv:2010.12514*, 2020.
- [39] Anoop Korattikara, Yutian Chen, and Max Welling. Austerity in MCMC land: Cutting the Metropolis-Hastings budget. In *International conference on machine learning*, pages 181–189. PMLR, 2014.
- [40] Krzysztof Łatuszyński and Daniel Rudolf. Convergence of hybrid slice sampling via spectral gap. *arXiv preprint arXiv:1409.2709*, 2014.
- [41] Faming Liang. A double Metropolis–Hastings sampler for spatial models with intractable normalizing constants. *Journal of Statistical Computation and Simulation*, 80(9):1007–1022, 2010.
- [42] Faming Liang, Ick Hoon Jin, Qifan Song, and Jun S Liu. An adaptive exchange algorithm for sampling from distributions with intractable normalizing constants. *Journal of the American Statistical Association*, 111(513):377–393, 2016.
- [43] L Lin, KF Liu, and J Sloan. A noisy Monte Carlo algorithm. *Physical Review D*, 61(7):074505, 2000.
- [44] Samuel Livingstone and Giacomo Zanella. The Barker Proposal: Combining Robustness and Efficiency in Gradient-Based MCMC. *Journal of the Royal Statistical Society Series B: Statistical Methodology*, 84(2):496–523, 01 2022.
- [45] Anne-Marie Lyne, Mark Girolami, Yves Atchadé, Heiko Strathmann, and Daniel Simpson. On russia roulette estimates for bayesian inference with doubly-intractable likelihoods. *Statistical Science*, 30(4):443–467, 2015.
- [46] Dougal Maclaurin and Ryan P. Adams. Firefly monte carlo: Exact mcmc with subsets of data. In *Conference on Uncertainty in Artificial Intelligence*, 2014.
- [47] Gael M. Martin, David T. Frazier, and Christian P. Robert. Computing Bayes: From Then ‘Til Now. *Statistical Science*, 39(1):3 – 19, 2024.

- [48] Nicholas Metropolis, Arianna W Rosenbluth, Marshall N Rosenbluth, Augusta H Teller, and Edward Teller. Equation of state calculations by fast computing machines. *The Journal of Chemical Physics*, 21(6):1087–1092, 1953.
- [49] Jesper Møller, Anthony N Pettitt, Robert Reeves, and Kasper K Berthelsen. An efficient Markov chain Monte Carlo method for distributions with intractable normalising constants. *Biometrika*, 93(2):451–458, 2006.
- [50] Iain Murray, Zoubin Ghahramani, and David JC MacKay. Mcmc for doubly-intractable distributions. In *Proceedings of the Twenty-Second Conference on Uncertainty in Artificial Intelligence*, pages 359–366, 2006.
- [51] Radford M Neal et al. MCMC using Hamiltonian dynamics. *Handbook of markov chain monte carlo*, 2(11):2, 2011.
- [52] Willie Neiswanger, Chong Wang, and Eric P Xing. Asymptotically exact, embarrassingly parallel mcmc. In *Proceedings of the Thirtieth Conference on Uncertainty in Artificial Intelligence*, pages 623–632, 2014.
- [53] Geoff K Nicholls, Colin Fox, and Alexis Muir Watt. Coupled MCMC with a randomized acceptance probability. *arXiv preprint arXiv:1205.6857*, 2012.
- [54] Omiros Papaspiliopoulos. *Monte Carlo probabilistic inference for diffusion processes: a methodological framework*, page 82–103. Cambridge University Press, 2011.
- [55] P. H. Peskun. Optimum monte-carlo sampling using markov chains. *Biometrika*, 60(3):607–612, 1973.
- [56] Sam Power, Daniel Rudolf, Björn Sprungk, and Andi Q Wang. Weak Poincaré inequality comparisons for ideal and hybrid slice sampling. *arXiv preprint arXiv:2402.13678*, 2024.
- [57] Qian Qin, Nianqiao Ju, and Guanyang Wang. Spectral gap bounds for reversible hybrid Gibbs chains. *arXiv preprint arXiv:2312.12782*, 2023.
- [58] Matias Quiroz, Robert Kohn, Mattias Villani, and Minh-Ngoc Tran. Speeding up MCMC by efficient data subsampling. *Journal of the American Statistical Association*, 2018.
- [59] Vinayak Rao, Lizhen Lin, and David B Dunson. Data augmentation for models based on rejection sampling. *Biometrika*, 103(2):319–335, 2016.

- [60] Gareth O Roberts and Jeffrey S Rosenthal. Geometric ergodicity and hybrid Markov chains. *Electronic Communications in Probability [electronic only]*, 2:13–25, 1997.
- [61] Garry Robins, Pip Pattison, Yuval Kalish, and Dean Lusher. An introduction to exponential random graph (p^*) models for social networks. *Social networks*, 29(2):173–191, 2007.
- [62] Daniel Seita, Xinlei Pan, Haoyu Chen, and John Canny. An efficient minibatch acceptance test for metropolis-hastings. In *Proceedings of the 27th International Joint Conference on Artificial Intelligence*, pages 5359–5363, 2018.
- [63] David J Strauss. A model for clustering. *Biometrika*, 62(2):467–475, 1975.
- [64] Luke Tierney. A note on metropolis-hastings kernels for general state spaces. *Annals of Applied Probability*, pages 1–9, 1998.
- [65] Michalis K Titsias and Omiros Papaspiliopoulos. Auxiliary gradient-based sampling algorithms. *Journal of the Royal Statistical Society Series B: Statistical Methodology*, 80(4):749–767, 2018.
- [66] Valeria Vitelli, Øystein Sørensen, Marta Crispino, Arnoldo Frigessi, and Elja Arjas. Probabilistic preference learning with the Mallows rank model. *Journal of Machine Learning Research*, 18(158):1–49, 2018.
- [67] Jure Vogrinc, Samuel Livingstone, and Giacomo Zanella. Optimal design of the barker proposal and other locally balanced Metropolis–Hastings algorithms. *Biometrika*, 110(3):579–595, 2023.
- [68] Sebastian J Vollmer, Konstantinos C Zygalakis, and Yee Whye Teh. Exploration of the (non-) asymptotic bias and variance of stochastic gradient langevin dynamics. *Journal of Machine Learning Research*, 17(159):1–48, 2016.
- [69] Wolfgang Wagner. Unbiased Monte Carlo evaluation of certain functional integrals. *Journal of Computational Physics*, 71(1):21–33, 1987.
- [70] Guanyang Wang. On the theoretical properties of the exchange algorithm. *Bernoulli*, 28(3):1935 – 1960, 2022.
- [71] Xiangyu Wang and David B Dunson. Parallelizing MCMC via Weierstrass sampler. *arXiv preprint arXiv:1312.4605*, 2013.

- [72] Max Welling and Yee W Teh. Bayesian learning via stochastic gradient langevin dynamics. In *Proceedings of the 28th international conference on machine learning (ICML-11)*, pages 681–688, 2011.
- [73] Tung-Yu Wu, YX Rachel Wang, and Wing H Wong. Mini-batch Metropolis–Hastings with reversible SGLD proposal. *Journal of the American Statistical Association*, 117(537):386–394, 2022.
- [74] Giacomo Zanella. Informed proposals for local mcmc in discrete spaces. *Journal of the American Statistical Association*, 115(530):852–865, 2020.
- [75] Ruqi Zhang, A Feder Cooper, and Christopher M De Sa. Asymptotically optimal exact minibatch Metropolis–Hastings. *Advances in Neural Information Processing Systems*, 33:19500–19510, 2020.
- [76] Ruqi Zhang and Christopher M De Sa. Poisson-minibatching for gibbs sampling with convergence rate guarantees. *Advances in Neural Information Processing Systems*, 32, 2019.

A Additional proofs

A.1 Verification of Example 3

To verify equation (1), first recall $\pi(\theta \mid x) = \exp\left(-\sum_{i=1}^N U_i(\theta; x)\right)/Z(x)$, and the definition

$$\phi_i(\theta, \theta'; x) = \frac{U_i(\theta'; x) - U_i(\theta; x)}{2} + \frac{c_i}{2} M(\theta, \theta').$$

It is useful to observe:

$$U_i(\theta; x) + \phi(\theta, \theta'; x) = \frac{U_i(\theta'; x) + U_i(\theta; x)}{2} + \frac{c_i M(\theta, \theta')}{2} = U_i(\theta'; x) + \phi(\theta', \theta; x).$$

We have :

$$\begin{aligned} R_{\theta \rightarrow \theta'}(\omega) \cdot \pi(\theta \mid x) \cdot P_{\theta \rightarrow \theta'}(\omega) &= R_{\theta \rightarrow \theta'}(\omega) \frac{\exp\left(-\sum_{i=1}^N U_i(\theta; x)\right)}{Z(x)} \prod_{i=1}^N P_{\theta \rightarrow \theta'}(s_i) \\ &= \frac{1}{Z(x)} \prod_{i=1}^N \left(\frac{\lambda c_i C^{-1} + \phi_i(\theta', \theta; x)}{\lambda c_i C^{-1} + \phi_i(\theta, \theta'; x)} \right)^{s_i} \cdot e^{-\phi_i(\theta, \theta'; x)} \cdot e^{-\lambda c_i C^{-1}} e^{-U_i(\theta; x)} \frac{(\lambda c_i C^{-1} + \phi_i(\theta, \theta'; x))^{s_i}}{s_i!} \\ &= \frac{1}{Z(x)} \prod_{i=1}^N \left(\frac{\lambda c_i C^{-1} + \phi_i(\theta', \theta; x)}{\lambda c_i C^{-1} + \phi_i(\theta, \theta'; x)} \right)^{s_i} \cdot e^{-\phi_i(\theta', \theta; x)} \cdot e^{-\lambda c_i C^{-1}} e^{-U_i(\theta'; x)} \frac{(\lambda M_i L^{-1} + \phi_i(\theta; x))^{s_i}}{s_i!} \\ &= \frac{1}{Z(x)} \prod_{i=1}^N \frac{(\lambda c_i C^{-1} + \phi_i(\theta', \theta; x))^{s_i}}{s_i!} (e^{-\lambda c_i C^{-1}} e^{-\phi_i(\theta', \theta; x)}) e^{-U_i(\theta'; x)} \\ &= \pi(\theta' \mid x) \prod_{i=1}^N P_{\theta' \rightarrow \theta}(s_i) \\ &= \pi(\theta' \mid x) P_{\theta' \rightarrow \theta}(\omega). \end{aligned}$$

A.2 Proof of Lemma 1

Proof of Lemma 1. We write $p_{\text{aux}}(\theta, \theta')$ in its integral form, and upper bound it as follows:

$$\begin{aligned}
p_{\text{aux}}(\theta, \theta') &= \int_{\Omega_1 \times \Omega_2} \mathbb{P}_\theta(\omega_1) q_{\omega_1}(\theta, \theta') \mathbb{P}_{\theta, \theta'}(\omega_2 | \omega_1) r(\theta, \theta'; \omega_1, \omega_2) \lambda_1 \otimes \lambda_2(d\omega_1 d\omega_2) \\
&= \int_{\Omega_1} \mathbb{P}_\theta(\omega_1) q_{\omega_1}(\theta, \theta') \left(\mathbb{E}_{\omega_2 | \omega_1, \theta, \theta'} \left[\min \left\{ \frac{\pi(\theta' | x) \mathbb{P}_{\theta', \theta}(\omega_1, \omega_2)}{\pi(\theta | x) \mathbb{P}_{\theta, \theta'}(\omega_1, \omega_2)} \cdot \frac{q_{\omega_1}(\theta', \theta)}{q_{\omega_1}(\theta, \theta')}, 1 \right\} \right] \right) \lambda_1(d\omega_1) \\
&\leq \int_{\Omega_1} \mathbb{P}_\theta(\omega_1) q_{\omega_1}(\theta, \theta') \left(\min \left\{ \frac{\pi(\theta' | x) q_{\omega_1}(\theta', \theta)}{\pi(\theta | x) q_{\omega_1}(\theta, \theta')} \mathbb{E}_{\omega_2 | \omega_1, \theta, \theta'} \left[\frac{\mathbb{P}_{\theta', \theta}(\omega_1, \omega_2)}{\mathbb{P}_{\theta, \theta'}(\omega_1, \omega_2)} \right], 1 \right\} \right) \lambda_1(d\omega_1) \\
&= \int_{\Omega_1} \mathbb{P}_\theta(\omega_1) q_{\omega_1}(\theta, \theta') \left(\min \left\{ \frac{\pi(\theta' | x) q_{\omega_1}(\theta', \theta)}{\pi(\theta | x) q_{\omega_1}(\theta, \theta')} \frac{\mathbb{P}_{\theta'}(\omega_1)}{\mathbb{P}_\theta(\omega_1)}, 1 \right\} \right) \lambda_1(d\omega_1) \\
&= p_{\text{MwG}}(\theta, \theta')
\end{aligned}$$

The first inequality follows from $\mathbb{E}[\min\{ah(X), c\}] \leq \min\{a\mathbb{E}[h(X)], c\}$, treating ω_2 as a random variable, and using the fact that

$$\begin{aligned}
\mathbb{E}_{\omega_2 | \omega_1, \theta, \theta'} \left[\frac{\mathbb{P}_{\theta', \theta}(\omega_1, \omega_2)}{\mathbb{P}_{\theta, \theta'}(\omega_1, \omega_2)} \right] &= \int_{\Omega_2} \frac{\mathbb{P}_{\theta'}(\omega_1) \mathbb{P}_{\theta', \theta}(\omega_2 | \omega_1)}{\mathbb{P}_\theta(\omega_1) \mathbb{P}_{\theta, \theta'}(\omega_2 | \omega_1)} \mathbb{P}_{\theta, \theta'}(\omega_2 | \omega_1) \lambda_2(d\omega_2) \\
&= \frac{\mathbb{P}_{\theta'}(\omega_1)}{\mathbb{P}_\theta(\omega_1)}.
\end{aligned}$$

The last equality follows from the calculations in Case 3.

Furthermore

$$\begin{aligned}
p_{\text{MwG}}(\theta, \theta') &\leq \min \left\{ \int \frac{\pi(\theta' | x)}{\pi(\theta | x)} q_{\omega_1}(\theta', \theta) \mathbb{P}_{\theta'}(\omega_1) \lambda_1(d\omega_1), \int q_{\omega_1}(\theta, \theta') \mathbb{P}_\theta(\omega_1) \lambda_1(d\omega_1) \right\} \\
&= \min \left\{ \frac{\pi(\theta' | x)}{\pi(\theta | x)} q_{\text{ideal}}(\theta', \theta), q_{\text{ideal}}(\theta, \theta') \right\} \\
&= q_{\text{ideal}}(\theta, \theta') \min \left\{ \frac{\pi(\theta' | x)}{\pi(\theta | x)} \frac{q_{\text{ideal}}(\theta', \theta)}{q_{\text{ideal}}(\theta, \theta')}, 1 \right\} \\
&= p_{\text{ideal}}(\theta, \theta').
\end{aligned}$$

Here, the inequality uses the fact $\int \min\{f, g\} \leq \min\{\int f, \int g\}$. \square

A.3 Proof of Proposition 7

Proof of Proposition 7. Let $\tilde{\phi}_i(\theta, \theta') := \lambda c_i/C + \phi_i(\theta, \theta'; x)$. We have $\mathbb{P}_{\theta, \theta'} = \otimes_{i=1}^N \text{Poi}(\tilde{\phi}_i(\theta, \theta'))$ for TunaMH. We can calculate the symmetrized KL divergence

$$\begin{aligned}
& d_{\text{KL}}(\mathbb{P}_{\theta, \theta'} || \mathbb{P}_{\theta', \theta}) + d_{\text{KL}}(\mathbb{P}_{\theta', \theta} || \mathbb{P}_{\theta, \theta'}) \\
&= \sum_{i=1}^N \left(\tilde{\phi}_i(\theta, \theta') \log \frac{\tilde{\phi}_i(\theta, \theta')}{\tilde{\phi}_i(\theta', \theta)} + \cancel{\tilde{\phi}_i(\theta', \theta)} - \cancel{\tilde{\phi}_i(\theta, \theta')} \right) + \sum_{i=1}^N \left(\tilde{\phi}_i(\theta', \theta) \log \frac{\tilde{\phi}_i(\theta', \theta)}{\tilde{\phi}_i(\theta, \theta')} + \cancel{\tilde{\phi}_i(\theta, \theta')} - \cancel{\tilde{\phi}_i(\theta', \theta)} \right) \\
&= \sum_{i=1}^N \left(\log \left(\tilde{\phi}_i(\theta, \theta') / \tilde{\phi}_i(\theta', \theta) \right) \right) \left(\tilde{\phi}_i(\theta, \theta') - \tilde{\phi}_i(\theta', \theta) \right) \\
&\leq \sum_{i=1}^N c_i M(\theta, \theta') \log \left(\frac{\lambda c_i / C + c_i M}{\lambda c_i / C} \right) \\
&\leq \sum_{i=1}^N c_i M(\theta, \theta') \frac{C M(\theta, \theta')}{\lambda} = \frac{C^2 M^2(\theta, \theta')}{\chi C^2 M^2(\theta, \theta')} = \frac{1}{\chi}.
\end{aligned}$$

Therefore, applying Proposition 3, we get:

$$\text{Gap}(\mathbb{P}_{\text{aux}}) \geq e^{-1/e} \exp\left\{-\frac{1}{2\chi}\right\} \text{Gap}(\mathbb{P}_{\text{ideal}}).$$

□

A.4 Proof of auxiliary results

Here we prove several auxiliary results mentioned in Section 4.4. Even though many of these results are well-known, we include proofs here for clarity and completeness.

Proposition 8. *For univariate Poisson random variables with parameter λ_1, λ_2 , their KL divergence equals $d_{\text{KL}}(\text{Poi}(\lambda_1) || \text{Poi}(\lambda_2)) = \lambda_1 \log(\lambda_1/\lambda_2) + \lambda_2 - \lambda_1$.*

Proof of Proposition 8. Let

$$P_\lambda(n) := \exp\{-\lambda\} \frac{\lambda^n}{n!}$$

be the probability mass function of a $\text{Poi}(\lambda)$ random variable evaluating at $n \in \mathbb{N}$. By definition, we have:

$$d_{\text{KL}}(\text{Poi}(\lambda_1) || \text{Poi}(\lambda_2)) = \mathbb{E}_{X \sim \text{Poi}(\lambda_1)} \left[\log \left(\frac{P_{\lambda_1}(X)}{P_{\lambda_2}(X)} \right) \right].$$

Therefore,

$$\begin{aligned}
d_{\text{KL}}(\text{Poi}(\lambda_1) \parallel \text{Poi}(\lambda_2)) &= \mathbb{E}_{X \sim \text{Poi}(\lambda_1)} \left[-\lambda_1 + \lambda_2 + X \log \left(\frac{\lambda_1}{\lambda_2} \right) \right] \\
&= \lambda_2 - \lambda_1 + \log \left(\frac{\lambda_1}{\lambda_2} \right) \mathbb{E}_{X \sim \text{Poi}(\lambda_1)} [X] \\
&= \lambda_1 \log \left(\frac{\lambda_1}{\lambda_2} \right) + \lambda_2 - \lambda_1.
\end{aligned}$$

□

Proposition 9. *For any λ_1, λ_2 satisfying $0 \leq m \leq \lambda_1 \leq \lambda_2 \leq M$, we have*

$$d_{\text{KL}}(\text{Poi}(\lambda_1) \parallel \text{Poi}(\lambda_2)) \leq d_{\text{KL}}(\text{Poi}(m) \parallel \text{Poi}(M)).$$

Proof of Proposition 9. The proposition is equivalent to proving that the function $g(\lambda_1, \lambda_2) := \lambda_1 \log(\lambda_1/\lambda_2) + \lambda_2 - \lambda_1$, constrained within the region $m \leq \lambda_1 \leq \lambda_2 \leq M$, is maximized at $\lambda_1 = m$ and $\lambda_2 = M$. We first fix λ_1 , and take derivative with respect to λ_2 :

$$\partial_{\lambda_2} g(\lambda_1, \lambda_2) = -\frac{\lambda_1}{\lambda_2} + 1.$$

The derivative is non-negative as $\lambda_2 \geq \lambda_1$. Therefore, $g(\lambda_1, \lambda_2) \leq g(\lambda_1, M)$. Then we arbitrarily fix λ_2 , and take derivative with respect to λ_1 :

$$\partial_{\lambda_1} g(\lambda_1, \lambda_2) = \log(\lambda_1) + 1 - \log(\lambda_2) - 1 = \log(\lambda_1) - \log(\lambda_2) \leq 0.$$

Therefore, $g(\lambda_1, \lambda_2) \leq g(m, \lambda_2)$. Putting these two inequalities together, we know:

$$d_{\text{KL}}(\text{Poi}(\lambda_1) \parallel \text{Poi}(\lambda_2)) = g(\lambda_1, \lambda_2) \leq g(m, M) = d_{\text{KL}}(\text{Poi}(m) \parallel \text{Poi}(M)).$$

□

Proposition 10. *Let P_i, Q_i be two probability measures on the same measurable space $(\Omega_i, \mathcal{F}_i)$ for $i \in \{1, 2, \dots, M\}$. Then we have*

$$d_{\text{KL}}(\otimes_{i=1}^M P_i \parallel \otimes_i Q_i) = \sum_{i=1}^M d_{\text{KL}}(P_i \parallel Q_i)$$

and

$$1 - d_{\text{TV}}(\otimes_{i=1}^M P_i, \otimes_i Q_i) \geq \prod_{i=1}^M (1 - d_{\text{TV}}(P_i, Q_i)).$$

Proof. Let $P := \otimes_{i=1}^M P_i$ and $Q := \otimes_{i=1}^M Q_i$, and $X = (X_1, \dots, X_M) \sim P$. Then

$$\begin{aligned} d_{\text{KL}}(P||Q) &= \mathbb{E}_{X \sim P} \left[\log \left(\frac{P(X)}{Q(X)} \right) \right] \\ &= \mathbb{E}_{X \sim P} \left[\sum_{i=1}^M \log \left(\frac{P(X_i)}{Q(X_i)} \right) \right] \\ &= \sum_{i=1}^M \mathbb{E}_{X_i \sim P_i} \left[\log \left(\frac{P(X_i)}{Q(X_i)} \right) \right] \\ &= \sum_{i=1}^M d_{\text{KL}}(P_i||Q_i), \end{aligned}$$

which proves the first equality.

For the second equality, recall the fact $d_{\text{TV}}(\mu, \nu) = 1 - \sup_{(X,Y) \in \Gamma(\mu,\nu)} \mathbb{P}(X = Y)$. Here, $\Gamma(\mu, \nu)$ denotes all the couplings of μ and ν (that is, all the joint distributions with the marginal distributions μ and ν). Now fix any $\epsilon > 0$, for each $i \in \{1, \dots, M\}$, choose $\Pi_i \in \Gamma(P_i, Q_i)$ such that

$$\mathbb{P}_{(X_i, Y_i) \sim \Pi_i}(X = Y) \geq 1 - d_{\text{TV}}(P_i, Q_i) - \epsilon.$$

Consider the $2M$ -dimensional vector $(X_1, Y_1, X_2, Y_2, \dots, X_M, Y_M) \sim \prod_{i=1}^M \Pi_i$, then the joint distribution of $X = (X_1, \dots, X_M)$ and $Y = (Y_1, \dots, Y_M)$ belongs to the $\Gamma(P, Q)$. Hence

$$1 - d_{\text{TV}}(P, Q) \geq \mathbb{P}(X = Y) = \prod_{i=1}^n \mathbb{P}(X_i = Y_i) \geq \prod_{i=1}^n (1 - d_{\text{TV}}(P_i, Q_i) - \epsilon).$$

Since ϵ is an arbitrary positive constant, we can let $\epsilon \rightarrow 0$ and get

$$1 - d_{\text{TV}}(\otimes_{i=1}^M P_i, \otimes_i Q_i) \geq \prod_{i=1}^M (1 - d_{\text{TV}}(P_i, Q_i)).$$

□

A.5 Additional properties

Proposition 11. *Let $a : [0, \infty) \rightarrow [0, 1]$ be any fixed function satisfying $a(t) = ta(1/t)$. Let $\tilde{\mathbb{P}}_{\text{aux}}(\cdot, \cdot)$ denote the transition kernel of the Markov chain as defined in Algorithm 4, with the exception that Step 7 is modified to: “with probability $a(r)$, set $\theta_{t+1} \leftarrow \theta'$; otherwise, set $\theta_{t+1} \leftarrow \theta_t$ ”. Then $\tilde{\mathbb{P}}_{\text{aux}}(\cdot, \cdot)$ remains reversible with respect to $\pi(\theta | x)$.*

Proof of Proposition 11. Let \tilde{p}_{aux} denote the density part of \tilde{P}_{aux} . We have the following expression of $\tilde{p}_{\text{aux}}(\theta, \theta')$:

$$\begin{aligned}\tilde{p}_{\text{aux}}(\theta, \theta') &= \int_{\Omega_1} \mathbb{P}_\theta(\omega_1) q_{\omega_1}(\theta, \theta') \int_{\Omega_2} \left(a \left(\frac{\Pi(\theta') \mathbb{P}_{\theta', \theta}(\omega_1, \omega_2)}{\Pi(\theta) \mathbb{P}_{\theta, \theta'}(\omega_1, \omega_2)} \cdot \frac{q_{\omega_1}(\theta', \theta)}{q_{\omega_1}(\theta, \theta')} \right) \mathbb{P}_{\theta, \theta'}(\omega_2 | \omega_1) \lambda_2(d\omega_2) \right) \lambda_1(d\omega_1) \\ &= \frac{\Pi(\theta')}{\Pi(\theta)} \int_{\Omega_1} \int_{\Omega_2} \mathbb{P}_{\theta', \theta}(\omega_1, \omega_2) q_{\omega_1}(\theta', \theta) a \left(\frac{\Pi(\theta) \mathbb{P}_{\theta, \theta'}(\omega_1, \omega_2)}{\Pi(\theta') \mathbb{P}_{\theta', \theta}(\omega_1, \omega_2)} \cdot \frac{q_{\omega_1}(\theta, \theta')}{q_{\omega_1}(\theta', \theta)} \right) \lambda_2(d\omega_2) \lambda_1(d\omega_1) \\ &= \frac{\Pi(\theta')}{\Pi(\theta)} \tilde{p}_{\text{aux}}(\theta', \theta),\end{aligned}$$

where the second equality uses $a(r) = ra(1/r)$. Therefore we have

$$\Pi(\theta) \tilde{p}_{\text{aux}}(\theta, \theta') = \Pi(\theta') \tilde{p}_{\text{aux}}(\theta', \theta),$$

as desired. \square

B Implementation details of locally balanced PoissonMH

According to [44], there are two different implementation versions of the locally balanced proposal on \mathbb{R}^d . The first approach is to treat each dimension of the parameter space separately. We define the proposal for dimension j as:

$$q_{\omega_1, j}^{(g)}(\theta, \theta'_j) := Z^{-1}(\theta, \omega_1) g \left(e^{\partial_{\theta_j} \log(\pi(\theta|x) \mathbb{P}_\theta(\omega_1)) (\theta'_j - \theta_j)} \right) \mu_j(\theta'_j - \theta_j),$$

where μ_j is a univariate symmetric density. The proposal on \mathbb{R}^d is then defined as the product of proposals on each dimension:

$$q_{\omega_1}^{(g)}(\theta, \theta') = \prod_{j=1}^d q_{\omega_1, j}^{(g)}(\theta, \theta'_j)$$

The second approach is to derive a multivariate version of the locally balanced proposal. Proposition 3.3 and the experiments in [44] show that the first approach is more efficient. As a result, we develop the auxiliary version of the first approach for both Poisson-Barker and Poisson-MALA.

Now, we provide implementation details of Poisson-MALA and Poisson-Barker. First, by taking $g(t) = \sqrt{t}$ and $\mu_j \sim N(0, \sigma^2)$ for every j , we derive the Poisson-MALA proposal as:

$$q_{\omega_1}^M(\theta, \theta') = \prod_{j=1}^d q_{\omega_1,j}^M(\theta, \theta'_j).$$

The j -th term in the product can be further represented as

$$\begin{aligned} q_{\omega_1,j}^M(\theta, \theta'_j) &\propto \exp \left\{ \frac{1}{2} \partial_{\theta_j} \log (\pi(\theta | x) \mathbb{P}_\theta(\omega_1)) (\theta' - \theta_j) \right\} \exp \left\{ -\frac{1}{2\sigma^2} (\theta' - \theta_j)^2 \right\} \\ &\propto \exp \left\{ -\frac{1}{2\sigma^2} \left(\theta' - \theta_j - \frac{\sigma^2}{2} \partial_{\theta_j} \log (\pi(\theta | x) \mathbb{P}_\theta(\omega_1)) \right)^2 \right\} \end{aligned}$$

which is Gaussian centered at $\theta_j + \frac{\sigma^2}{2} \partial_{\theta_j} \log (\pi(\theta | x) \mathbb{P}_\theta(\omega_1))$ with standard deviation σ . This proposal is similar to MALA except that we substitute the gradient of $\log \pi(\theta | x)$ by the gradient of $\log \pi(\theta | x) \mathbb{P}_\theta(\omega_1)$, enabling efficient implementation. Since all the dimensions are independent, our Poisson-MALA proposal is a multivariate Gaussian with mean $\theta + 0.5\sigma^2 \nabla_\theta \log (\pi(\theta | x) \mathbb{P}_\theta(\omega_1))$ and covariance matrix $\sigma^2 \mathbb{I}_d$.

Next, when taking $g(t) = t/(1+t)$, we can similarly write our Poisson-Barker proposal as:

$$q_{\omega_1}^B(\theta, \theta') = \prod_{j=1}^d q_{\omega_1,j}^B(\theta, \theta'_j).$$

The j -th term in the product can be further represented as

$$q_{\omega_1,j}^B(\theta, \theta'_j) = Z^{-1}(\theta, \omega_1) \frac{\mu_j(\theta'_j - \theta)}{1 + e^{-\partial_{\theta_j} \log (\pi(\theta | x) \mathbb{P}_\theta(\omega_1)) (\theta'_j - \theta)}}.$$

Now we claim $Z(\theta, \omega_1)$ is a constant 0.5. To verify our claim, we first notice that $g(t) = t/(1+t) = 1 - g(t^{-1})$. Let $C_{\theta, \omega_1} := \partial_{\theta_j} \log (\pi(\theta | x) \mathbb{P}_\theta(\omega_1))$, we can integrate:

$$\begin{aligned} Z(\theta, \omega_1) &= \int_{\mathbb{R}} g(\exp\{C_{\theta, \omega_1}(\theta'_j - \theta_j)\}) \mu_j(\theta'_j - \theta_j) d\theta'_j \\ &= \int_{\mathbb{R}} g(\exp\{C_{\theta, \omega_1}t\}) \mu_j(t) dt \\ &= \int_{t>0} g(\exp\{C_{\theta, \omega_1}t\}) \mu_j(t) dt + \int_{t<0} g(\exp\{C_{\theta, \omega_1}t\}) \mu_j(t) dt \\ &= \int_{t>0} g(\exp\{C_{\theta, \omega_1}t\}) \mu_j(t) dt + \int_{t>0} g(\exp\{-C_{\theta, \omega_1}t\}) \mu_j(t) dt \quad \text{since } \mu_j(t) = \mu_j(-t) \\ &= \int_{t>0} \mu_j(t) dt \\ &= 0.5. \end{aligned}$$

The same calculation is used in [44]. Therefore our proposal has an explicit form

$$q_{\omega_1,j}^B(\theta, \theta'_j) = 2 \frac{\mu_j(\theta'_j - \theta_j)}{1 + e^{-\partial_{\theta_j} \log(\pi(\theta|x)\mathbb{P}_\theta(\omega_1))(\theta'_j - \theta_j)}}.$$

Sampling from this proposal distribution can be done perfectly by the following simple procedure: given current θ_j , one first propose $z \sim \mu_j$, then move to $\theta_j + z$ with probability $g(e^{\partial_{\theta_j} \log(\pi(\theta|x)\mathbb{P}_\theta(\omega_1))z})$, and to $\theta_j - z$ with probability $1 - g(e^{\partial_{\theta_j} \log(\pi(\theta|x)\mathbb{P}_\theta(\omega_1))z})$. This procedure samples exactly from $q_{\omega_1,j}^B(\theta, \theta'_j)$, see also Proposition 3.1 of [44] for details. Sampling from the d -dimensional proposal can be done by implementing the above strategy for each dimensional independently.

Both algorithms require evaluating $\nabla \log(\pi(\theta | x)\mathbb{P}_\theta(\omega_1))$, and we verify that this only requires evaluation on the same minibatch as PoissonMH. As derived in Section 3.3.1,

$$\pi(\theta | x)\mathbb{P}_\theta(\omega_1) = \frac{1}{Z(x)} \exp(-\lambda) \prod_{i:s_i>0} \frac{(\lambda M_i L^{-1} + \phi_i(\theta; x))^{s_i}}{s_i!}.$$

Then, we take the logarithm:

$$\log \pi(\theta | x)\mathbb{P}_\theta(\omega_1) = -\log Z(x) - \lambda - \sum_{i:s_i>0} s_i! + \sum_{i:s_i>0} s_i \log \{\lambda M_i L^{-1} + \phi_i(\theta; x)\}.$$

The gradient is:

$$\nabla_\theta \log \pi(\theta | x)\mathbb{P}_\theta(\omega_1) = \sum_{i:s_i>0} s_i \frac{\nabla_\theta \phi_i(\theta; x)}{\lambda M_i L^{-1} + \phi_i(\theta; x)}$$

As a result, we only need to evaluate the gradient on the minibatch data.

C Further details of experiments in Section 5

C.1 Heterogeneous truncated Gaussian

Recall that for fixed θ , our data $\{y_1, y_2, \dots, y_N\}$ is assumed to be i.i.d. with distribution $y_i | \theta \sim \mathcal{N}(\theta, \Sigma)$. With uniform prior on the hypercube $[-K, K]^d$, our target is the following tempered posterior:

$$\pi(\theta | y) \propto \exp \left\{ -\frac{1}{2} \beta \sum_{i=1}^N (\theta - y_i)^\top \Sigma^{-1} (\theta - y_i) \right\} \mathbb{I}(\theta \in [-K, K]^d)$$

Now we look at the function $\tilde{\phi}_i(\theta) = -\frac{1}{2}\beta(\theta - y_i)^\top \Sigma^{-1}(\theta - y_i)$. It is clear that $\tilde{\phi}_i(\theta) \leq 0$.

On the other hand:

$$\tilde{\phi}_i(\theta) \geq -\frac{1}{2}\beta\lambda_{\max}(\Sigma^{-1})\|\theta - y_i\|_2^2 \geq -\frac{1}{2}\beta\lambda_{\max}(\Sigma^{-1})\sum_{j=1}^d(|y_{ij}| + K)^2,$$

where $\lambda_{\max}(\Sigma^{-1})$ is the largest eigenvalue of Σ^{-1} . Set

$$M_i := \frac{1}{2}\beta\lambda_{\max}(\Sigma^{-1})\sum_{j=1}^d(|y_{ij}| + K)^2,$$

and define $\phi_i(\theta) := \tilde{\phi}_i(\theta) + M_i$, we have $\phi_i(\theta) \in [0, M_i]$ for each i . Meanwhile, the posterior is $\pi(\theta | y) \propto \exp\{\sum_{i=1}^N \phi_i(\theta)\}$, as adding a constant to $\tilde{\phi}_i(\theta)$ does not affect the posterior.

C.2 Robust Linear Regression

Similarly, we can derive a bound for each

$$\tilde{\phi}_i(\theta) := -\frac{v+1}{2} \log \left(1 + \frac{(y_i - \theta^\top x_i)^2}{v} \right)$$

in the robust linear regression example. Suppose we constraint θ in the region $\{\theta, \|\theta\|_2 \leq R\}$, then in order to get the upper bound M_i , we first find the following maximum:

$$\max_{\theta} (y_i - \theta^\top x_i)^2 \text{ subject to } \|\theta\|_2 \leq R$$

The above maximization problem can be solved analytically, with the solution:

$$\theta^* = \begin{cases} \frac{-x_i}{\|x_i\|_2} \cdot R & \text{when } y_i > 0 \\ \frac{x_i}{\|x_i\|_2} \cdot R & \text{when } y_i < 0. \end{cases}$$

The maximum equals $(|y_i| + \|x_i\|_2 \cdot R)^2$ for both cases. Taking

$$M_i = \frac{v+1}{2} \log \left(1 + \frac{(|y_i| + \|x_i\|_2 \cdot R)^2}{v} \right),$$

and set $\phi_i(\theta) := \tilde{\phi}_i(\theta) + M_i$, we have $\phi_i(\theta) \in [0, M_i]$ for each i . Meanwhile, the posterior is $\pi(\theta | y) \propto \exp\{\sum_{i=1}^N \phi_i(\theta)\}$.

C.3 Bayesian Logistic Regression on MNIST

For $i = 1, 2, \dots, N$, let $x_i \in \mathbb{R}^d$ be the features and $y_i \in \{0, 1\}$ be the label. The logistic regression model is $p_\theta(y_i = 1 | x) = h(x_i^\top \theta)$, where $h(z) := (1 + \exp(-z))^{-1}$ is the sigmoid function. We assume the data are i.i.d distributed. With a flat prior, the target posterior distribution is:

$$\pi(\theta | y, x) \propto \exp \left\{ - \sum_{i=1}^N -y_i \log h(x_i^\top \theta) - (1 - y_i) \log h(-x_i^\top \theta) \right\}.$$

Let us denote $U_i(\theta) = -y_i \log h(x_i^\top \theta) - (1 - y_i) \log h(-x_i^\top \theta)$. To derive the required bound for $U_i(\theta)$ in Algorithm 3 and 6, first notice that $U_i(\theta)$ is continuous and convex with gradient $\nabla_\theta U_i(\theta) = (h(x_i^\top \theta) - y_i) \cdot x_i$. Then, we have:

$$U_i(\theta') - U_i(\theta) \leq \nabla_\theta U_i(\theta)^\top (\theta' - \theta)$$

and

$$\begin{aligned} |U_i(\theta') - U_i(\theta)| &\leq \|\nabla_\theta U_i(\theta)\|_2 \cdot \|\theta' - \theta\|_2 \\ &\leq \|x_i\|_2 \cdot \|\theta' - \theta\|_2 \end{aligned}$$

where the first inequality follows from Cauchy–Schwarz inequality and the second inequality comes from $|h(x_i^\top \theta) - y_i| \leq 1$. Thus, we can take $M_i(\theta, \theta') = \|\theta' - \theta\|_2$ and $c_i = \|x_i\|_2$ for TunaMH and Tuna-SGLD.

D Additional experiments

D.1 Bayesian Logistic Regression: 7s and 9s

Similar to Section 5.3, we run TunaMH, TunaMH-SGLD, MH and MALA on Bayesian logistic regression task to classify 7s and 9s in the MNIST data set. The training set contains 12214 samples, and the test set contains 2037 samples. We adopt all implementation details from Section 5.3 and report the results in Figure 8 and 9.

We can draw similar conclusions to Section 5.3 that our proposed minibatch gradient-based algorithm performs better than its original minibatch non-gradient version in all step size configurations. And all minibatch methods significantly improve the performances of full batch methods.

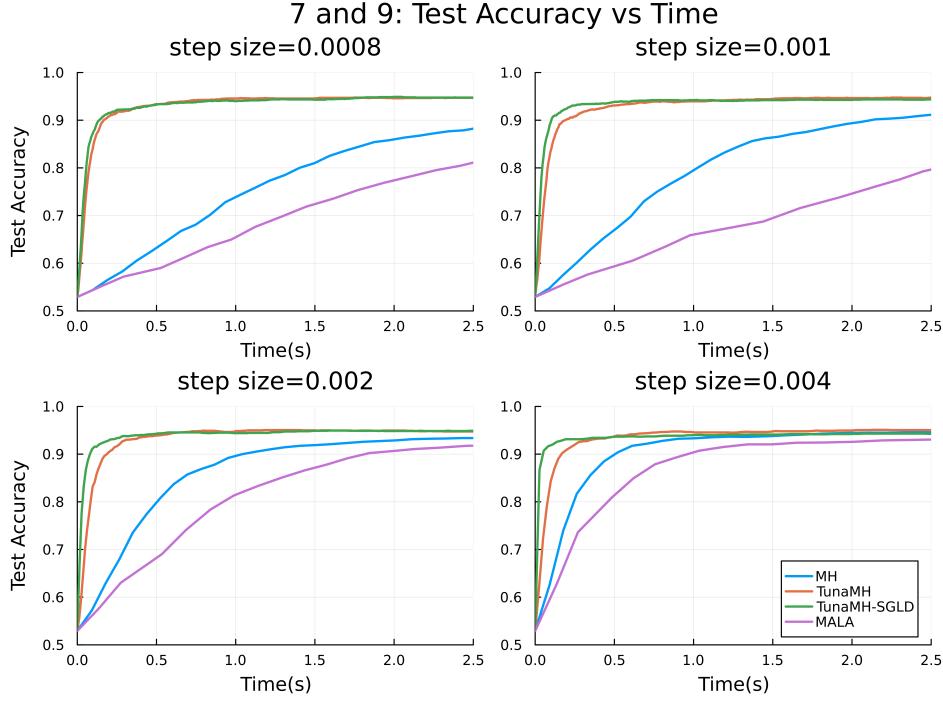


Figure 8: Test accuracy as a function of time for classifying 7s and 9s in MNIST. Step sizes for all methods are varied across $\{0.0008, 0.001, 0.002, 0.004\}$. Each subplot reports the curve of test accuracy in the first 1.5 seconds.

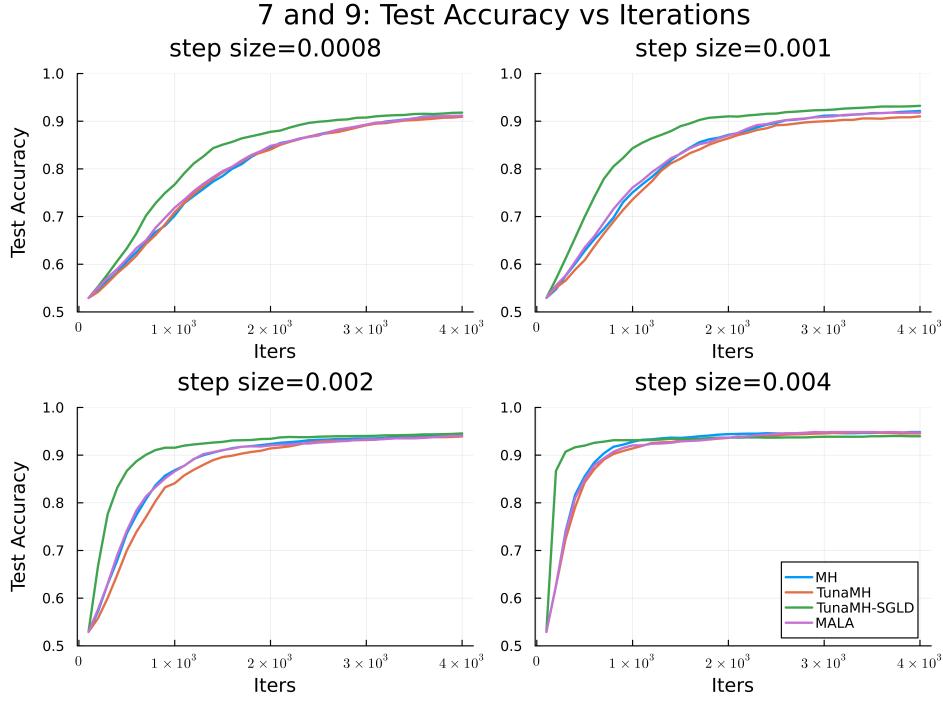


Figure 9: Test accuracy as a function of the number of iterations for classifying 7s and 9s in MNIST. Step sizes for methods are varied across $\{0.0008, 0.001, 0.002, 0.004\}$. Each subplot reports the curve of test accuracy in the first 4,000 steps.