Aggregated Gradient Langevin Dynamics

Chao Zhang,[1,2*] **Jiahao Xie,**[1] **Zebang Shen,**[3†] **Peilin Zhao,**[2] **Tengfei Zhou,**[1] **Hui Qian**[1]
[1]College of Computer Science and Technology, Zhejiang University
[2]Tencent AI Lab, [3]University of Pennsylvania
{zczju, xiejh, zhoutengfei, qianhui}@zju.edu.cn, zebang@seas.upenn.edu, masonzhao@tencent.com

Abstract

In this paper, we explore a general Aggregated Gradient Langevin Dynamics framework (AGLD) for the Markov Chain Monte Carlo (MCMC) sampling. We investigate the nonasymptotic convergence of AGLD with a *unified* analysis for different data accessing (e.g. *random access*, *cyclic access* and *random reshuffle*) and snapshot updating strategies, under convex and nonconvex settings respectively. It is the first time that bounds for I/O friendly strategies such as *cyclic access* and *random reshuffle* have been established in the MCMC literature. The theoretic results also indicate that methods in AGLD possess the merits of both the low per-iteration computational complexity and the short mixture time. Empirical studies demonstrate that our framework allows to derive novel schemes to generate high-quality samples for large-scale Bayesian posterior learning tasks.

1 Introduction

We focus on the Langevin dynamics based Markov Chain Monte Carlo (MCMC) methods for sampling the parameter vector $\theta \in \mathbb{R}^d$ from a target posterior distribution

$$p^* \triangleq p(\theta|\{z_i\}_{i=1}^N) \propto p(\theta) \prod_{i=1}^N p(z_i|\theta), \tag{1}$$

where $p(\theta)$ is some prior of θ, z_i's are the data points observed, and $p(z_i|\theta)$ is the likelihood function. The Langevin dynamics Monte Carlo method (LMC) adopts the *gradient* of log-posterior in an iterative manner to drive the distribution of samples to the target distribution efficiently (Roberts and Stramer 2002; Roberts, Tweedie, and others 1996; Mattingly, Stuart, and Higham 2002). To reduce the computational complexity for large-scale posterior learning tasks, the Stochastic Gradient Langevin Dynamics method (SGLD), which replaces the expensive full gradient with the stochastic gradient, has been proposed (Welling and Teh 2011). While such scheme enjoys a significantly reduced per-iteration cost, the mixture time, i.e., the total number of iterations required to achieve the correction from an out-of-equilibrium configuration to the target posterior distribution, is increased, due to the extra variance introduced by the approximation (Dalalyan and Karagulyan 2017; Dalalyan 2017b).

In recent years, efforts are made to design variance-control strategies to circumvent this slow convergence issue in the SGLD In particular, borrowing ideas from variance reduction methods in the optimization literature (Johnson and Zhang 2013; Defazio, Bach, and Lacoste-Julien 2014; Lei and Jordan 2017), the variance-reduced SGLD variants exploit the high correlations between consecutive iterates to construct *unbiased aggregated gradient* approximations with less variance, which leads to better mixture time guarantees (Dubey et al. 2016; Zou, Xu, and Gu 2018b). Among these methods, SAGA-LD and SVRG-LD (Dubey et al. 2016) are proved to be the most effective ones when high-quality samples are required (Chatterji et al. 2018; Zou, Xu, and Gu 2019). While the nonasymptotic convergence guarantees for SVRG-LD and SAGA-LD have been established, it is difficult to seamlessly extend these analyses to cover other Langevin dynamics based MCMC methods with different efficient gradient approximations.

- First of all, different delicate Lyapunov functions are designed for SVRG-LD and SAGA-LD to prove the nonasymptotic convergence to the stationary distribution. Due to the different targets of optimization and MCMC, the mixture-time analysis is not a simple transition of the convergence rate analysis in optimization. The lack of a unified perspective of these variance-reduced SGLD algorithms makes it difficult to effectively explore other variance-reduced estimators used in optimization (e.g., HSAG (Reddi et al. 2015)) for Langevin dynamics based MCMC sampling. In particular, customized Lyapunov functions need to be designed if new variance-reduced estimators are adopted.
- Second, existing theoretical analysis relies heavily on the randomness of the data accessing strategy to construct an unbiased estimator of the true gradient. In practice, the random access strategy entails heavy I/O cost, i.e., lots of

[*] This work is done when Chao Zhang works as an intern in Tencent AI Lab.

[†] Corresponding Author

data swap between the memory and the disk, when the dataset is too large to fit into memory, thereby renders existing incremental Langevin dynamics based MCMC algorithms heavily impractical for sampling tasks in the big data scenario. While other data accessing strategies such as *cyclic access* and *random reshuffle* are known to be disk I/O friendly (Xie et al. 2018), existing analysis can not be directly extended to algorithms with these strategies.

Contributions Motivated by such imperatives, we propose a general framework named Aggregated Gradient Langevin Dynamics (AGLD), which maintains a historical snapshot set of the gradient to construct more accurate gradient approximations than SGLD. AGLD possesses a three-step structure: *Data-Accessing*, *Sample-Searching*, and *Snapshot-Updating*. Different Data-Accessing (e.g. *random access*, *cyclic access* and *random reshuffle*) and Snapshot-Updating strategies can be utilized in this framework. By appropriately implementing these two steps, we can obtain several practical gradient approximations, including those used in existing methods like SVRG-LD and SAGA-LD. Under mild assumptions, a unified mixture-time analysis of AGLD is established, which holds as long as each component of the snapshot set is updated at least once in a fixed duration. We list our main contributions as follows.

- We first analyze the mixture time of AGLD under the assumptions that the negative log-posterior $f(x)$ is smooth and strongly convex and then extend the analysis to the general convex case. We also provide theoretical analysis for nonconvex $f(x)$. These results indicate that AGLD has similar mixture time bounds as LMC under similar assumptions, while the per-iteration computation is much less than that of LMC. Moreover, the analysis provides a unified bound for a wide class of algorithms with no need to further design dedicated Lyapunov functions for different Data-Accessing and Snapshot-Updating strategies.

- For the first time, mixture time guarantee for cyclic access and random reshuffle Data-Accessing strategies is provided in the Langevin dynamics based MCMC literature. This fills the gap of practical use and theoretical analyses, since cyclic access is I/O friendly and often used as a practical substitute for random access when the dataset is too large to fit into memory.

- We develop a novel Snapshot-Updating strategy, named Time-based Mixture Updating (TMU), which enjoys the advantages of both the Snapshot-Updating strategies used in SVRG-LD and SAGA-LD: it always updates components in the snapshot set to newly computed ones as in SAGA-LD and also periodically updates the whole snapshot set to rule out the out-of-date ones as in SVRG-LD. Plugging TMU into AGLD, we derive novel algorithms to generate high-quality samples for Bayesian learning tasks.

Simulated and real-world experiments are conducted to validate our analysis. Empirical results demonstrate the advantages of proposed variants over the state-of-the-art.

2 Preliminaries

2.1 Wasserstein Distance and Mixture Time

We use the 2-Wasserstein ($\mathcal{W}_2$) distance to evaluate the effectiveness of our methods. Specifically, the $\mathcal{W}_2$ distance between two probability measures ρ and ν is defined as

$$\mathcal{W}_2^2(\rho,\nu) = \inf_{\pi\in\Gamma(\rho,\nu)} \{\int \|x-y\|_2^2 \mathbf{d}\pi(x,y)\}.$$

Here, (x, y) are random variables with distribution density π and $\Gamma(\rho,\nu)$ denotes the collection of joint distributions with ρ and ν as its marginals. In this paper, we say K is the *ϵ-mixture time* of a Monte Carlo sampling procedure if for every $k \geq K$, the distribution $p^{(k)}$ of the sample generated in the k-th iteration satisfies $\mathcal{W}_2(p^{(k)}, p^*) \leq \epsilon$.

2.2 Stochastic Langevin Dynamics

By using the discretization of certain dynamics, dynamics based MCMC methods allow us to efficiently sample from the target distribution. A large portion of such works are based on the Langevin Dynamics (Parisi 1981)

$$\mathbf{d}\theta(t) = -\nabla_\theta f(\theta(t))\mathbf{d}t + \sqrt{2}\mathbf{d}B(t), \tag{2}$$

where ∇f is called the drift term, $B(t)$ is a d-dimensional Brownian Motion and $\theta(t) \in \mathbb{R}^d$ is the state variable.

The classic Langevin dynamics Monte Carlo method (LMC) generates samples $\{x^{(k)}\}$ in the following manner:

$$x^{(k+1)} = x^{(k)} - \eta\nabla f(x^{(k)}) + \sqrt{2\eta}\xi^{(k)}, \tag{3}$$

where $x^{(k)}$ is the time discretization of the continuous time dynamics $\theta(t)$, η is the stepsize and $\xi^{(k)} \sim \mathbf{N}(0, \mathbf{I}_{d\times d})$ is the d-dimensional Gaussian variable. The distribution $p^{(k)}$ of sample $x^{(k)}$ is shown to converge weakly to the target distribution p^* (Dalalyan 2017a; Raginsky, Rakhlin, and Telgarsky 2017).

To alleviate the expensive full gradient computation in LMC, the Stochastic Gradient Langevin Dynamics (SGLD) replaces $\nabla f(x^{(k)})$ in (3) by the stochastic approximation

$$g^{(k)} = \frac{N}{n}\sum_{i\in I_k} \nabla f_i(x^{(k)}), \tag{4}$$

where I_k is the set of n indices drawn from $[N]$ i.i.d. in iteration k and each $f_i(\theta) = -\log p(\theta|z_i) - \log p(\theta)/N$, for $i \in [N]$. Although the $g^{(k)}$ is always an unbiased estimator of the full gradient, the non-diminishing variance results in the inefficiency of sample-space exploration and slows down the convergence to the target distribution.

To overcome such difficulty, SVRG-LD and SAGA-LD (Dubey et al. 2016; Chatterji et al. 2018; Zou, Xu, and Gu 2019) use the two different variance-reduced gradient estimators of $\nabla f(x)$, which utilize the component gradient information of the past samples. While possessing similar low per-iteration component gradient computation as in SGLD, the mixture time bound of SVRG-LD and SAGA-LD are shown to be similar to that of LMC (Chatterji et al. 2018).

Algorithm 1 Aggregated Gradient Langevin Dynamics

```
Require: initial iterate $x^{(0)}$, stepsize $\eta$, Data-Accessing strategy, and Snapshot-Updating strategy.
Initialize Snapshot set $\mathcal{A}^{(0)} = \{\alpha_i^{(0)}\}_{i=1}^N$, where $\alpha_i^{(0)} = \nabla f_i(x^{(0)})$.
for $k = 0$ to $K-1$ do
    $S_k$ = Data-Accessing(k).
    Sample-Searching: find $x^{(k+1)}$ according to (5).
    $\mathcal{A}^{(k+1)} =$ Snapshot-Updating($\mathcal{A}^{(k)}, x^{(k)}, k, S_k$).
end for
```

3 Aggregated Gradient Langevin Dynamics

In this section, we present our general framework named Aggregated Gradient Langevin Dynamics (AGLD). Specifically, AGLD maintains a snapshot set consisting of component gradients evaluated in historical iterates. The information in the snapshot set is used in each iteration to construct a gradient approximation which helps to generate the next iterate. Note that iterates generated during the procedure are samples of random variables, whose distributions converge to the target distribution. At the end of each iteration, the entries in the snapshot set are updated according to some strategy. By customizing the steps in AGLD with different strategies, we can derive different algorithms. Concretely, AGLD is comprised of the following three steps, where the first and third steps can accept different strategies as inputs.

i **Data-Accessing:** select a subset of indices S_k from $[N]$ according to the input strategy.

ii **Sample-Searching:** construct the aggregated gradient approximation $g^{(k)}$ using the data points indexed by S_k and the historical snapshot set, then generate the next iterate (the new sample) by taking one step along the direction of $g^{(k)}$ with an injected Gaussian noise. Specifically, the $(k+1)$-th sample is obtained in the following manner

$$x^{(k+1)} = x^{(k)} - \eta g^{(k)} + \sqrt{2\eta}\xi^{(k)}, \tag{5}$$

where $\xi^{(k)}$ is a Gaussian noise, η is the stepsize, and

$$g^{(k)} = \sum_{i \in S_k} \frac{N}{n}(\nabla f_i(x^{(k)}) - \alpha_i^{(k)}) + \sum_{i=1}^{N} \alpha_i^{(k)}. \tag{6}$$

Here, $\alpha_i^{(k)}$'s are components in the snapshot set $\mathcal{A}^{(k)}$.

iii **Snapshot-Updating:** update historical snapshot set according to the input strategy.

We summarize AGLD in Algorithm 1. While our mixture time analyses hold as long as the input Data-Accessing and Snapshot-Updating strategies meet Requirements 1 and 2, we describe in detail several typical qualified implementations of these two steps below.

3.1 The Data-Accessing Step

We make the following requirement on the Data-Accessing step to ensure the convergence of $\mathcal{W}_2$ distance between the sample distribution $p^{(k)}$ and the target distribution p^*.

Requirement 1. *In every iteration, each point in the dataset has been visited at least once in the past C iterations, where C is some fixed positive constant.*

We note that Requirement 1 is general and covers three commonly used data accessing strategies: Random Access (RA), Random Reshuffle (RR), and Cyclic Access (CA).

RA: Select uniformly n indices from $[N]$ with replacement;

RR: Select sequentially n indices from $[N]$ with a permutation at the beginning of each data pass;

CA: Select n indices from $[N]$ in a cyclic way.

RA is widely used to construct unbiased gradient approximations in gradient-based Langevin dynamics methods, which is amenable to theoretical analysis. However, in big data scenarios when the dataset does not fit into the memory, RA is not memory-friendly, since it entails heavy data exchange between memory and disks. On the contrary, CA strategy promotes the spatial locality property significantly and therefore reduces the page fault rate when handling huge datasets using limited memory (Xie et al. 2018). RR can be considered as a trade-off between RA and CA. However, methods with either CA or RR are difficult to analyze in that the gradient approximation is commonly not an unbiased estimator of the true gradient (Shamir 2016).

It can be verified that these strategies satisfy Requirement 1, For RR, in the k-th iteration, all the data points have been accessed in the past $2N/n$ iterations. For CA, all the data points are accessed in the past N/n iterations. Note that, RA satisfies the Requirement 1 with $C = \mathcal{O}(N \log N)$ w.h.p., according to the Coupon Collector Theorem (Dawkins 1991).

3.2 The Snapshot-Updating Step

The Snapshot-Updating step maintains a snapshot set $\mathcal{A}^{(k)}$ such that in the k-th iteration, $\mathcal{A}^{(k)}$ contains N records $\alpha_i^{(k)}$ for $\nabla f_i(y_i^{(k)})$ where $y_i^{(k)}$ is some historic iterate $y_i^{(k)} = x^{(j)}$ with $j \leq k$. Additionally, for our analyses to hold, the input strategy should satisfy the following requirement.

Requirement 2. *The compents in the gradient snapshot set $\mathcal{A}^{(k)}$ should have been updated in the past D iterations, i.e. $\alpha_i^{(k)} \in \{\nabla f_i(x^{(j)})\}_{j=k-D}^k$, where D is a fixed constant.*

This requirement guarantees that $\alpha_i^{(k)}$'s are not far from the $\nabla f_i(x^{(k)})$'s and thus can be used to construct a proper approximation of $\nabla f(x^{(k)})$. The Snapshot-Updating step tries to strike a balance between the approximation accuracy and the computation cost. Specifically, in each iteration, updating a larger portion of the N entries in the snapshot set would lead to a more accurate gradient approximation at the cost of a higher computation burden. In the following, we list three feasible Snapshot-Updating strategies considered in this paper:Periodically Total Update (PTU), Per-iteration Partial Update (PPU), and Time-based Mixture Update (TMU).

PTU: This strategy operates in an epoch-wise manner: at the beginning of each epoch all the entries in the snapshot set are updated to the current component gradient $\alpha_i^{(k)} = \nabla f_i(x^{(k)})$, and in the following D–1 iterations the snapshot

Strategy 2 PTU($\mathcal{A}^{(k)}, x^{(k)}, x^{(k+1)}, k, S_k$)

```
for $i = 1$ to $N$ do
    $\alpha_i^{(k+1)} = \mathbb{I}_{\{\text{mod }(k+1,D)=0\}}\nabla f_i(x^{(k+1)}) + \mathbb{I}_{\{\text{mod }(k+1,D)\neq 0\}}\alpha_i^{(k)}$
end for
return $\mathcal{A}_{k+1}$
```

Strategy 3 PPU($\mathcal{A}^{(k)}, \theta^{(k)}, k, S_k$)

```
for $i = 1$ to $N$ do
    $\alpha_i^{(k+1)} = \mathbb{I}_{\{i\in S_k\}}\nabla f_i(x^{(k)}) + \mathbb{I}_{\{i\notin i_k\}}\alpha_i^{(k)}$
end for
return $\mathcal{A}_{k+1}$
```

Strategy 4 TMU($\mathcal{A}^{(k)}, \theta^{(k)}, k, S_k$)

```
for $i = 1$ to $N$ do
    if mod$(k+1, D) = 0$ then
        $\alpha_i^{(k+1)} = \nabla f_i(x^{(k+1)})$
    else
        $\alpha_i^{(k+1)} = \mathbb{I}_{\{i\in S_k\}}\nabla f_i(x^{(k)}) + \mathbb{I}_{\{i\notin S_k\}}\alpha_i^{(k)}$
    end if
end for
return $\mathcal{A}_{k+1}$
```

set remains unchanged (see Strategy 2). Such synchronous update to the snapshot set allows us to implement PTU in a memory efficient manner. In the k-th iteration, PTU only needs to store the iterate $\tilde{x}$ and its gradient $\nabla f(\tilde{x})$ where $\tilde{x} = x^{k-\text{mod}(k,D)}$, as we can obtain the snapshot entry $\alpha_i^{(k)}$ via a simple evaluation of the corresponding component gradient at $\tilde{x}$ in the calculation of $g^{(k)}$. Therefore the PTU strategy is preferable when storage is limited.

PPU: This strategy substitutes $\alpha_i^{(k)}$ by $\nabla f_i(x^{(k)})$ for $i \in S_k$ in the k-th iteration (see Strategy 3). This partial substitution strategy together with Requirement 1 can ensure the Requirement 2. The downside of PPU is the extra $\mathcal{O}(d\cdot N)$ memory used to keep the snapshot set $\mathcal{A}^{(k)}$. Fortunately, in many applications of interests, $\nabla f_i(x)$ is actually the product of a scalar and the data point z_i, which implies that only $\mathcal{O}(N)$ extra storage is needed to store N scalars.

TMU: This strategy updates the whole $\mathcal{A}$ once every D iterations and substitutes $\alpha_i^{(k)}$ by $\nabla f_i(x^{(k)})$ in the k-th iteration (see Strategy 4). TMU possesses the merits of both PPU and PTU: it updates components of gradient snapshot set in S_k to newly computed one in each iteration as PPU, and also periodically updates the whole snapshot set as PTU in case that there exist indices unselected for a long time.

Remark 1. *PPU is the Snapshot-Updating strategy used in SAGA-LD and PTU is the strategy used in SVRG-LD (Dubey et al. 2016). To the best of our knowledge, TMU has never been proposed in the MCMC literature before. Note that the HSAG Snapshot-Updating strategy proposed by Reddi et al. (2015) also satisfies our requirement, and we omit the discussion of it due to the limit of space.*

3.3 Derived Algorithms

By plugging the aforementioned Data-Accessing and Snapshot-Updating strategies into AGLD, we derive several practical algorithms. We name the algorithms by "Snapshot-updating - Data-Accessing", e.g. TMU-RA uses TMU as the Snapshot-Updating strategy and RA as the Data-Accessing strategy. Note that we recover SAGA-LD and SVRG-LD with PPU-RA and PTU-RA, respectively. In the following section, we provide unified analyses for all derived algorithms under different regularity conditions. We emphasize that, in the absence of the unbiasedness of the gradient approximation, our mixture time analyses are the first to cover algorithms with the I/O friendly cyclic data accessing scheme.

4 Theoretical Analysis

In this section, we provide the mixture time analysis for AGLD. The detailed proofs of the theorems are postponed to the long version of this paper due to the limit of space.

4.1 Analysis for AGLD with strongly convex $f(x)$

We first investigate the $\mathcal{W}_2$ distance between the sample distribution $p^{(k)}$ of the iterate $x^{(k)}$ and the target distribution p^* under the smoothness and strong convexity assumptions.

Assumption 1 (Smoothness)**.** *Each individual f_i is $\tilde{M}$-smooth. That is, f_i is twice differentiable and there exists a constant $\tilde{M} > 0$ such that for all $x, y \in \mathbb{R}^d$*

$$f_i(y) \le f_i(x) + \langle \nabla f_i(x), y - x\rangle + \frac{\tilde{M}}{2}\|x - y\|_2^2. \qquad (7)$$

Accordingly, we can verify that the summation f of $f_i's$ is M-smooth with $M = \tilde{M}N$.

Assumption 2 (Strong Convexity)**.** *The sum f is μ-strongly convex. That is, there exists a constant $\mu > 0$ such that for all $x, y \in \mathbb{R}^d$,*

$$f(y) \ge f(x) + \langle \nabla f(x), y - x\rangle + \frac{\mu}{2}\|x - y\|_2^2. \qquad (8)$$

Note that these assumptions are satisfied by many Bayesian sampling models such as Bayesian ridge regression, Bayesian logistic regression and Bayesian Independent Component Analysis, and they are used in many existing analyses of Langevin dynamics based MCMC methods (Dalalyan 2017b; Baker et al. 2017; Zou, Xu, and Gu 2018b; Chatterji et al. 2018).

Theorem 1. *Under Assumption 1, 2 and Requirement 2, AGLD outputs sample $\mathbf{x}^{(k)}$ with its distribution $p^{(k)}$ satisfying $\mathcal{W}_2(p^{(k)}, p^*) \le \epsilon$ for any $k \ge K = \tilde{\mathcal{O}}(\epsilon^{-2})$ with $\eta = \mathcal{O}(\epsilon^2)$.*

Remark 2. *Under this assumption, the ϵ-mixture time K of AGLD has the same dependency on ϵ as that of LMC (Dalalyan 2017b). Note that we hide the dependency of other regularity parameters such as μ, L and N in the $\mathcal{O}(\cdot)$ for simplicity. Actually, AGLD methods with CA/RR have a worse dependency on these parameters than algorithms with RA. However, when the dataset does not fit into the memory, the sequential data accessing nature of CA enjoys less I/O cost than random data accessing, which makes CA based AGLD methods have a better time efficiency than the RA based ones.*

The bound of the mixture time for AGLD with RA can be improved under the Lipschitz-continuous Hessian condition.

Assumption 3. *[Lipschitz-continuous Hessian] There exists a constant $L > 0$ such that for all $x, y \in \mathbb{R}^d$*

$$\|\nabla^2 f(x) - \nabla^2 f(y)\| \leq L\|x - y\|_2.$$

Theorem 2. *Under Assumption 1, 2, 3 and Requirement 2, AGLD methods with RA output sample $\mathbf{x}^{(k)}$ with its distribution $p^{(k)}$ satisfying $\mathcal{W}_2(p^{(k)}, p^*) \leq \epsilon$ for any $k \geq K = \mathcal{O}(\log(1/\epsilon)/\epsilon)$ by setting $\eta = \mathcal{O}(\epsilon)$.*

Additionally, when we adopt the random data accessing scheme, the mixture time of the newly proposed TMU-RA method can be written in a more concrete form, which is established in the following theorem.

Theorem 3. *Under Assumption 1, 2, 3 and denote $\kappa = M/\mu$. TMU-RA outputs sample $\mathbf{x}^{(k)}$ with its distribution $p^{(k)}$ satisfying $\mathcal{W}_2(p^{(k)}, p^*) \leq \epsilon$ for any $k \geq K = \tilde{\mathcal{O}}(\kappa^{3/2}\sqrt{d}/(n\epsilon))$ if we set $\eta < \epsilon n\sqrt{\mu}/(M\sqrt{dN})$, $n \geq 9$, and $D = N$.*

Remark 3. *Note that the component gradient complexity to achieve $\mathcal{W}_2(p^{(k)}, p^*) \leq \epsilon$ in TMU-RA is $T_g = \tilde{\mathcal{O}}(N + \kappa^{3/2}\sqrt{d}/\epsilon)$, which is the same as those of SAGA-LD (Chatterji et al. 2018) and SVRG-LD (Zou, Xu, and Gu 2018b). Practically, in our experiments, TMU based variants always have a better empirically performance than the PPU based and PTU based counterparts as the entries in the snapshot set maintained by TMU is more up-to-date.*

4.2 Extension to general convex $f(x)$

Following a similar idea from (Zou, Xu, and Gu 2018a), we can extend AGLD to drawing samples from densities with general convex $f(x)$. Firstly, we construct the following strongly convex approximation $\hat{f}(x)$ of $f(x)$,

$$\hat{f}(x) = f(x) + \lambda\|x\|^2/2.$$

Then, we run AGLD to generate samples with $\hat{f}(x)$ until the sample distribution $p^{(K)}$ satisfies $\mathcal{W}_2(p^{(K)}, \hat{p}^*) \leq \epsilon/2$ where $\hat{p}^* \propto e^{-\hat{f}(x)}$ denotes stationary distribution of Langevin Dynamics with the drift term $\nabla\hat{f}$ (check 2 for definition). If we choose a proper λ to make $\mathcal{W}_2(\hat{p}^*, p^*) \leq \epsilon/2$, then by the triangle inequality of the $\mathcal{W}_2$ distance, we have $\mathcal{W}_2(p^{(K)}, p^*) \leq \mathcal{W}_2(p^{(K)}, p^*) + \mathcal{W}_2(\hat{p}^*, p^*) \leq \epsilon$. Thus, we have the following theorem.

Theorem 4. *Suppose the assumptions in Theorem 1 hold and further assume the target distribution $p^* \propto e^{-f}$ has bounded forth order moment, i.e. $\mathbb{E}_{p^*}[\|x\|_2^4] \leq \hat{U}d^2$. If we choose $\lambda = 4\epsilon^2/(\hat{U}d^2)$ and run the AGLD algorithm with $\hat{f}(x) = f(x) + \lambda\|x\|^2/2$, we have $\mathcal{W}_2(p^{(k)}, p^*) \leq \epsilon$ for any $k \geq K = \tilde{\mathcal{O}}(\epsilon^{-8})$. If we further assume that f has Lipschitz-continuous Hessian, then SVRG-LD, SAGA-LD, and TMU-RA can achieve $\mathcal{W}_2(p^{(K)}, p^*) \leq \epsilon$ in $K = \tilde{\mathcal{O}}(\epsilon^{-3})$ iterations.*

4.3 Theoretical results for nonconvex $f(x)$

In this subsection, we characterize the ϵ-mixture time of AGLD for sampling from densities with nonconvex $f(x)$. The following assumption is necessary for our theory.

Assumption 4. *[Dissipative] There exists constants $a, b > 0$ such that for all $x \in \mathbb{R}^d$, the sum f satisfies*

$$\langle \nabla f(x), x \rangle \geq b\|x\|_2^2 - a.$$

This assumption is typical for the ergodicity analysis of stochastic differential equations and diffusion approximations. It indicates that, starting from a position that is sufficiently far from the origin, the Langevin dynamics (2) moves towards the origin on average. With this assumption, we establish the following theorem on the nonasymptotic convergence of AGLD for nonconvex $f(x)$.

Theorem 5. *Under Assumption 1, 4, and Requirement 2, AGLD outputs sample $\mathbf{x}^{(k)}$ with distribution $p^{(k)}$ satisfying $\mathcal{W}_2(p^{(k)}, p^*) \leq \epsilon$ for any $k \geq K = \tilde{\mathcal{O}}(\epsilon^{-4})$ with $\eta = \mathcal{O}(\epsilon^4)$.*

Remark 4. *This $\tilde{\mathcal{O}}(\epsilon^{-4})$ result is similar to the bound for LMC sampling from nonconvex $f(x)$ (Raginsky, Rakhlin, and Telgarsky 2017). Note that, as pointed out by (Raginsky, Rakhlin, and Telgarsky 2017), vanilla SGLD fails to converge in this setting.*

5 Related Work

In this section, we briefly review the literature of Langevin dynamics based MCMC algorithms.

By directly discretizing the Langevin dynamics (2), Roberts, Tweedie, and others (1996) proposed to use LMC (3) to generate samples of the target distribution. The first nonasymptotic analysis of LMC was established by Dalalyan (2017b), which analyzed the error of approximating the target distribution with strongly convex $f(x)$ in the total variational distance. This result was soon improved by Durmus, Moulines, and others (2017). Later, Durmus and Moulines (2016) and Cheng and Bartlett (2018) established the convergence of LMC in the 2-Wasserstein distance and KL-divergence, respectively. While the former works focus on sampling from distribution with (strongly-)convex $f(x)$, Raginsky, Rakhlin, and Telgarsky (2017) investigated the nonasymptotic convergence of LMC in the 2-Wasserstein distance when $f(x)$ is nonconvex.

With the increasing amount of data size in modern machine learning tasks, SGLD method(Welling and Teh 2011), which replaces the full gradient in LMC with a stochastic gradient (Robbins and Monro 1951), has received much attention. Vollmer, Zygalakis, and others (2015) analyzed the nonasymptotic bias and variance of SGLD using Poisson equations, and Dalalyan and Karagulyan (2017) proved the convergence of SGLD in the 2-Wasserstein distance when the target distribution is strongly log-concave. Despite the great success of SGLD, the large variance of stochastic gradients may lead to unavoidable bias (Baker et al. 2017; Betancourt 2015; Brosse, Durmus, and Moulines 2018). To overcome this, Teh, Thiery, and Vollmer (2016) proposed to decrease the step size to alleviate the bias and proved the asymptotic rate of SGLD in terms of Mean Square Error (MSE). Dang et al. (2019) utilized an approximate MH correction step, which only uses part of the whole data set, to decrease the influence of variance.

Another way to reduce the variance of stochastic gradients and save gradient computation is to apply variance-reduction

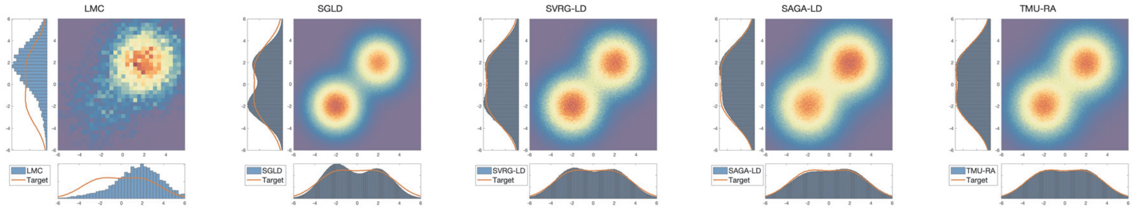

Figure 1: Gaussian Mixture Model. The red line denotes the projection of the target distribution p^*.

Table 1: Statistics of datasets used in our experiments.

DATASET	DIMENSION	DATASIZE
YEARPREDICTIONMSD	90	515,345
SLICELOACTION	384	53500
CRITEO	999,999	45,840,617
KDD12	54,686,45	149,639,105

techniques. Dubey et al. (2016) used two different variance-reduced gradient estimators of $\nabla f(x)$, which utilize the component gradient information of the past samples, and devised SVRG-LD and SAGA-LD algorithms. They proved that these two algorithms improve the MSE upon SGLD. Chatterji et al. (2018) and Zou, Xu, and Gu (2019) studied the nonasymptotic convergence of these methods in the 2-Wasserstein distance when sampling from densities with strongly convex and nonconvex $f(x)$, respectively. Their results show that SVRG-LD and SAGA-LD can achieve similar ϵ-mixture time bound as LMC w.r.t. ϵ, while the per-iteration computational cost is similar to that of SGLD. There is another research line which uses the mode of the log-posterior to construct control-variate estimates of full gradients (Baker et al. 2017; Bierkens, Fearnhead, and Roberts 2016; Nagapetyan et al. 2017; Chatterji et al. 2018; Brosse, Durmus, and Moulines 2018). However, calculating the mode is intractable for large-scale problems, rendering these methods impractical for real-world Bayesian learning tasks.

6 Experiments

We follow the experiment settings in the literature (Zou, Xu, and Gu 2018b; Dubey et al. 2016; Chatterji et al. 2018; Welling and Teh 2011; Zou, Xu, and Gu 2019) and conduct empirical studies on two simulated experiments (sampling from distribution with convex and nonconvex f, respectively) and two real-world applications (Bayesian Logistic Regression and Bayesian Ridge Regression). Nine instances of AGLD are considered, including SVRG-LD (PTU-RA), PTU-RR, PTU-CA, SAGA-LD (PPU-RA), PPU-RR, PPU-CA, TMU-RA, TMU-RR, and TMU-CA. We also include LMC, SGLD, SVGR-LD+ (Zou, Xu, and Gu 2018b), SVRG-RR+ and SVRG-CA+[1] as baselines. Due to the limit of space, we put the experiment sampling from distribution with convex f in our long version. The statistics of datasets are listed in Table 1.

6.1 Sampling for Gaussian Mixture Distribution

In this simulated experiment, we consider sampling from distribution $p^* \propto \exp(-f(x)) = \exp(-\sum_{i=1}^N f_i(x)/N)$, where each component $\exp(-f_i(x))$ is defined as $\exp(-f_i(x)) = e^{-\|x-a_i\|_2^2/2} + e^{-\|x+a_i\|2/2}, a_i \in \mathbb{R}^d$. It can be verified that $\exp(-f_i(x))$ is proportional to the PDF of a Gaussian mixture distribution. According to (Dalalyan 2017b), when the parameter a_i is chosen such that $\|a_i\|^2 \geq 1$, $f_i(x)$ is nonconvex. We set the sample size $N = 500$ and dimension $d = 10$, and randomly generate parameters $a_i \sim \mathbf{N}(\mu, \Sigma)$ with $\mu = (2, \cdots, 2)^T$ and $\Sigma = \mathbf{I}_{d\times d}$.

In this experiment, we fix the Data-Accessing strategy to RA in AGLD and compare the performance of LMC, SGLD, SVRG-LD, SAGA-LD and TMU-RA algorithms. We run all algorithms for 2×10^4 data passes, and make use of the iterates in the last 10^4 data passes to visualize distributions.

In Figure 1, we report the 2D projection of the densities of random samples generated by each algorithm. It can be observed that all three AGLD methods, i.e., SVRG-LD, SAGA-LD and TMU-RA, can well approximate the target distribution in 2×10^4 data passes, while the distributions generated by LMC and SGLD have obvious deviation from the true one. Moreover, the results show that the sample probability of TMU-RA approximates the target distribution best among the three AGLD methods.

6.2 Bayesian Ridge Regression

Bayesian ridge regression aims to predict the response y according to the covariate x, given the dataset $\mathbf{Z} = \{x_i, y_i\}_{i=1}^N$. The response y is modeled as a random variable sampled from a conditional Gaussian distribution $p(y|x, w) = \mathbf{N}(w^T x, \lambda)$, where w denotes the weight variable and has a Gaussian prior $p(w) = \mathbf{N}(0, \lambda \mathbf{I}_{d\times d})$. By the Bayesian rule, one can infer w from the posterior $p(w|\mathbf{Z})$ and use it to make the prediction. Two publicly available benchmark datasets are used for evaluation: YearPredictionMSD and SliceLocation[2].

In this task, we fix the Data-Accessing strategy to RA and compare the performance of different Snapshot-Updating strategies. To have a better understanding of the newly-proposed TMU Snapshot-Updating strategy, we also investigate the performance of TMU type methods with different Data-Accessing strategies.

[1] SVRG-RR+ is the random reshuffle variant of SVRG-LD+, and SVRG-CA+ is the cyclic access variant of SVRG-LD+.

[2] https://archive.ics.uci.edu/ml/index.php

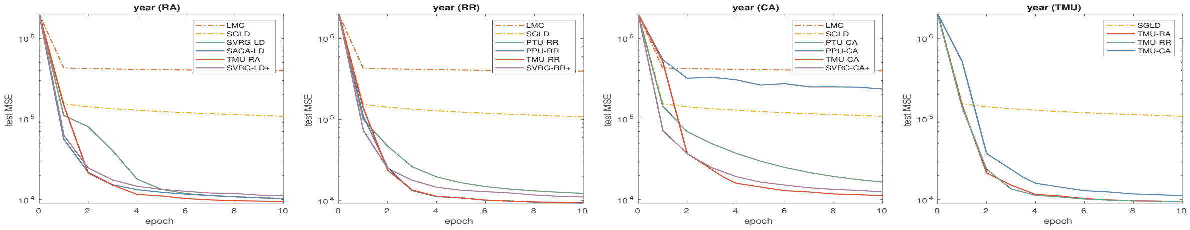

Figure 2: Bayesian Ridge Regression.

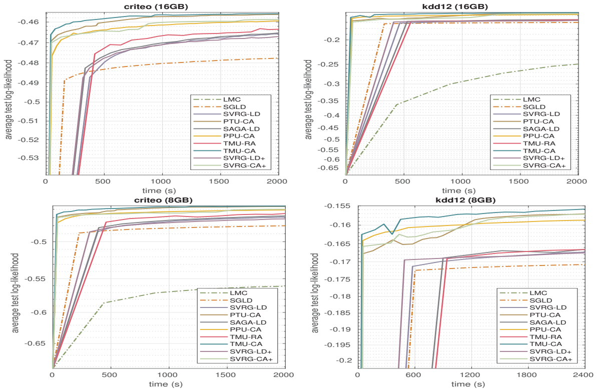

Figure 3: Bayesian Logistic Regression.

By randomly partitioning the dataset into training $(4/5)$ and testing $(1/5)$ sets, we report the test Mean Square Error (MSE) of the compared methods on YearPredictionMSD in Fig. 2. The results for SliceLocation are similar to that of YearPredictionMSD, and are postponed to the Appendix due to the limit of space. We use the number of effective passes (epoch) of the dataset as the x-axis, which is proportional to the CPU time. From the first three columns of the figure, we can see that (i) TMU-type methods have the best performance among all the methods with the same Data-Accessing strategy, (ii) SVRG+ and PPU type methods constantly outperform LMC, SGLD, and PTU type methods. These results validate the advantage of TMU strategy over PPU and PTU. The last column of Figure 2 shows that TMU-RA outperforms TMU-CA/TMU-RR, when the dataset is fitted to the memory. These results imply that the TMU-RA is the best choice if we have enough memory.

6.3 Bayesian Logistic Regression

Bayesian Logistic Regression is a robust binary classification task. Let $\mathbf{Z} = \{x_i, y_i\}_{i=1}^N$ be a dataset with $y_i \in \{-1, 1\}$ denoting the sample label and $x_i \in \mathbb{R}^d$ denoting the sample covariate vector. The conditional distribution of label y is modeled by $p(y|x, w) = \phi(y_i w^T x_i)$, where $\phi(\cdot)$ is the sigmoid function and the prior of w is $p(w) = \mathbf{N}(0, \lambda \mathbf{I}_{d \times d})$.

We focus on the big data setting, where the physical memory is insufficient to load the entire dataset. Specifically, two large-scale datasets criteo (27.32GB) and kdd12 (26.76GB) are used [3] and we manually restrict the available physical memory to 16 GB and 8 GB for simulation.

We demonstrate that CA strategy is advantageous in such setting by comparing 6 AGLD methods with either CA or RA in the experiment, namely, SVRG-LD, PTU-CA, SAGA-LD, PPU-CA, TMU-RA, and TMU-CA. We also include LMC, SGLD, SVRG-LD+, and SVRG-CA+ as baseline. Methods with the RR strategy have almost identical performance as their RA counterparts and are hence omitted. The average test log-likelihood versus execution time are reported in Fig. 3. The empirical results show that methods with CA outperform their RA counterparts. As the amount of physical memory gets smaller (from 16 GB to 8GB), the time efficiency of CA becomes more apparent. The results also show that TMU has better performance than other Snapshot-Updating strategies with the same Data-Accessing strategy.

7 Conclusion and Future Work

In this paper, we proposed a general framework called Aggregated Gradient Langevin Dynamics (AGLD) for Bayesian posterior sampling. A unified analysis for AGLD is provided without the need to design different Lyapunov functions for different methods individually. In particular, we establish the first theoretical guarantees for cyclic access and random reshuffle based methods. By introducing the new Snapshot-Updating strategy TMU, we derive some new methods under AGLD. Empirical results validate the efficiency and effectiveness of the proposed TMU in both simulated and real-world tasks. The theoretical analysis and empirical results indicate that TMU-RA would be the best choice if the memory is sufficient and TMU-CA would be used, otherwise.

Acknowledgements

This work is supported by Zhejiang Provincial Natural Science Foundation of China under Grant No. LZ18F020002, and National Natural Science Foundation of China (Grant No: 61672376, 61751209, 61472347).

References

Baker, J.; Fearnhead, P.; Fox, E. B.; and Nemeth, C. 2017. Control variates for stochastic gradient mcmc. *arXiv preprint arXiv:1706.05439*.

Betancourt, M. 2015. The fundamental incompatibility of scalable hamiltonian monte carlo and naive data subsampling. In *International Conference on Machine Learning*, 533–540.

Bierkens, J.; Fearnhead, P.; and Roberts, G. 2016. The zig-zag process and super-efficient sampling for bayesian analysis of big data. *arXiv preprint arXiv:1607.03188*.

[3] https://www.csie.ntu.edu.tw/~cjlin/libsvmtools/datasets

Brosse, N.; Durmus, A.; and Moulines, E. 2018. The promises and pitfalls of stochastic gradient langevin dynamics. In *Advances in Neural Information Processing Systems*, 8268–8278.

Chatterji, N. S.; Flammarion, N.; Ma, Y.-A.; Bartlett, P. L.; and Jordan, M. I. 2018. On the theory of variance reduction for stochastic gradient monte carlo. *arXiv preprint arXiv:1802.05431*.

Cheng, X., and Bartlett, P. L. 2018. Convergence of langevin mcmc in kl-divergence. *PMLR 83* (83):186–211.

Dalalyan, A. S., and Karagulyan, A. G. 2017. User-friendly guarantees for the langevin monte carlo with inaccurate gradient. *arXiv preprint arXiv:1710.00095*.

Dalalyan, A. S. 2017a. Further and stronger analogy between sampling and optimization: Langevin monte carlo and gradient descent. *arXiv preprint arXiv:1704.04752*.

Dalalyan, A. S. 2017b. Theoretical guarantees for approximate sampling from smooth and log-concave densities. *Journal of the Royal Statistical Society: Series B (Statistical Methodology)* 79(3):651–676.

Dang, K.-D.; Quiroz, M.; Kohn, R.; Tran, M.-N.; and Villani, M. 2019. Hamiltonian monte carlo with energy conserving subsampling. *Journal of machine learning research* 20(100):1–31.

Dawkins, B. 1991. Siobhan's problem: the coupon collector revisited. *The American Statistician* 45(1):76–82.

Defazio, A.; Bach, F.; and Lacoste-Julien, S. 2014. Saga: A fast incremental gradient method with support for non-strongly convex composite objectives. In *NIPS*, 1646–1654.

Dubey, K. A.; Reddi, S. J.; Williamson, S. A.; Poczos, B.; Smola, A. J.; and Xing, E. P. 2016. Variance reduction in stochastic gradient langevin dynamics. In *Advances in Neural Information Processing Systems*, 1154–1162.

Durmus, A., and Moulines, E. 2016. High-dimensional bayesian inference via the unadjusted langevin algorithm. *arXiv preprint arXiv:1605.01559*.

Durmus, A.; Moulines, E.; et al. 2017. Nonasymptotic convergence analysis for the unadjusted langevin algorithm. *The Annals of Applied Probability* 27(3):1551–1587.

Johnson, R., and Zhang, T. 2013. Accelerating stochastic gradient descent using predictive variance reduction. In *NIPS*, 315–323.

Lei, L., and Jordan, M. 2017. Less than a single pass: Stochastically controlled stochastic gradient. In *Artificial Intelligence and Statistics*, 148–156.

Mattingly, J. C.; Stuart, A. M.; and Higham, D. J. 2002. Ergodicity for sdes and approximations: locally lipschitz vector fields and degenerate noise. *Stochastic processes and their applications* 101(2):185–232.

Nagapetyan, T.; Duncan, A. B.; Hasenclever, L.; Vollmer, S. J.; Szpruch, L.; and Zygalakis, K. 2017. The true cost of stochastic gradient langevin dynamics. *arXiv preprint arXiv:1706.02692*.

Parisi, G. 1981. Correlation functions and computer simulations. *Nuclear Physics B* 180(3):378–384.

Raginsky, M.; Rakhlin, A.; and Telgarsky, M. 2017. Non-convex learning via stochastic gradient langevin dynamics: a nonasymptotic analysis. *arXiv preprint arXiv:1702.03849*.

Reddi, S. J.; Hefny, A.; Sra, S.; Poczos, B.; and Smola, A. J. 2015. On variance reduction in stochastic gradient descent and its asynchronous variants. In *NIPS*, 2647–2655.

Robbins, H., and Monro, S. 1951. A stochastic approximation method. *The annals of mathematical statistics* 400–407.

Roberts, G. O., and Stramer, O. 2002. Langevin diffusions and metropolis-hastings algorithms. *Methodology and computing in applied probability* 4(4):337–357.

Roberts, G. O.; Tweedie, R. L.; et al. 1996. Exponential convergence of langevin distributions and their discrete approximations. *Bernoulli* 2(4):341–363.

Shamir, O. 2016. Without-replacement sampling for stochastic gradient methods. In *NeurIPS*, 46–54.

Teh, Y. W.; Thiery, A. H.; and Vollmer, S. J. 2016. Consistency and fluctuations for stochastic gradient langevin dynamics. *Journal of Machine Learning Research* 17:1–33.

Vollmer, S. J.; Zygalakis, K. C.; et al. 2015. (non-) asymptotic properties of stochastic gradient langevin dynamics. *arXiv preprint arXiv:1501.00438*.

Welling, M., and Teh, Y. W. 2011. Bayesian learning via stochastic gradient langevin dynamics. In *ICML*, 681–688.

Xie, J.; Qian, H.; Shen, Z.; and Zhang, C. 2018. Towards memory-friendly deterministic incremental gradient method. In *International Conference on Artificial Intelligence and Statistics*, 1147–1156.

Zou, D.; Xu, P.; and Gu, Q. 2018a. Stochastic variance-reduced hamilton monte carlo methods. *arXiv preprint arXiv:1802.04791*.

Zou, D.; Xu, P.; and Gu, Q. 2018b. Subsampled stochastic variance-reduced gradient langevin dynamics. *UAI*.

Zou, D.; Xu, P.; and Gu, Q. 2019. Sampling from non-log-concave distributions via variance-reduced gradient langevin dynamics. In *AISTATS*, 2936–2945.