Biometrika (2019), *pp*. 1–8
Printed in Great Britain

Extended Stochastic Gradient MCMC for Large-Scale Bayesian Variable Selection

BY QIFAN SONG, YAN SUN, MAO YE, FAMING LIANG
Department of Statistics, Purdue University,
West lafayette, IN 47906, USA
qfsong, sun748, ye207, fmliang@purdue.edu

SUMMARY

Stochastic gradient Markov chain Monte Carlo (MCMC) algorithms have received much attention in Bayesian computing for big data problems, but they are only applicable to a small class of problems for which the parameter space has a fixed dimension and the log-posterior density is differentiable with respect to the parameters. This paper proposes an extended stochastic gradient MCMC algorithm which, by introducing appropriate latent variables, can be applied to more general large-scale Bayesian computing problems, such as those involving dimension jumping and missing data. Numerical studies show that the proposed algorithm is highly scalable and much more efficient than traditional MCMC algorithms. The proposed algorithms have much alleviated the pain of Bayesian methods in big data computing.

Some key words: Dimension Jumping, Missing Data, Stochastic Gradient Langevin Dynamics, Subsampling

1. INTRODUCTION

After six decades of continual development, MCMC has proven to be a powerful and typically unique computational tool for analyzing data of complex structures. However, for large datasets, its computational cost can be prohibitive as it requires all of the data to be processed at each iteration. To tackle this difficulty, a variety of scalable algorithms have been proposed in the recent literature. According to the strategies they employed, these algorithms can be grouped into a few categories, including stochastic gradient MCMC algorithms (Welling & Teh, 2011; Ding et al., 2014; Ahn et al., 2012; Chen et al., 2014; Betancourt, 2015; Ma et al., 2015; Nemeth & Fearnhead, 2019), split-and-merge algorithms (Scott et al., 2016; Srivastava et al., 2018; Xue & Liang, 2019), mini-batch Metropolis-Hastings algorithms (Chen et al., 2016; Korattikara et al., 2014; Bardenet et al., 2014; Maclaurin & Adams, 2014; Bardenet et al., 2017), nonreversible Markov process-based algorithms (Bierkens et al., 2019; Bouchard Coté et al., 2018), and some discrete sampling algorithms based on the multi-armed bandit (Chen & Ghahramani, 2016).

Although scalable algorithms have been developed for both continuous and discrete sampling problems, they are hard to be applied to dimension jumping problems. Dimension jumping is characterized by variable selection where the number of parameters changes from iteration to iteration in MCMC simulations. Under their current settings, the stochastic gradient MCMC and nonreversible Markov process-based algorithms are only applicable to problems for which the parameter space has a fixed dimension and the log-posterior density is differentiable with respect to the parameters. For the split-and-merge algorithms, it is unclear how to aggregate samples of different dimensions drawn from the posterior distributions based on different subset data. The multi-armed bandit algorithms are only applicable to problems with a small discrete domain and can be extremely inefficient for high-dimensional variable selection problems. The mini-batch Metropolis-Hastings algorithms are in principle applicable to dimension jumping problems. However, they are generally difficult to use. For example, the algorithms by Chen et al. (2016), Korattikara et al. (2014), and Bardenet et al. (2014) perform approximate acceptance tests using subset data. The amount of data consumed for each test varies significantly from one iteration to another, which

compromise their scalability. The algorithms by Maclaurin & Adams (2014) and Bardenet et al. (2017) perform exact tests but require a lower bound on the parameter distribution across its domain. Unfortunately, the lower bound is usually difficult to obtain.

This paper proposes an extended stochastic gradient Langevin dynamics algorithm which, by introducing appropriate latent variables, extends the stochastic gradient Langevin dynamics algorithm to more general large-scale Bayesian computing problems such as variable selection and missing data. The extended stochastic gradient Langevin dynamics algorithm is highly scalable and much more efficient than traditional MCMC algorithms. Compared to the mini-batch Metropolis-Hastings algorithms, the proposed algorithm is much easier to use, which involves only a fixed amount of data at each iteration and does not require any lower bound on the parameter distribution.

2. A BRIEF REVIEW OF STOCHASTIC GRADIENT LANGEVIN DYNAMICS

Let $X_N = (X_1, X_2, \ldots, X_N)$ denote a set of N independent and identically distributed samples drawn from the distribution $f(x|\theta)$, where N is the sample size and θ is the parameter. Let $p(X_N|\theta) = \prod_{i=1}^N f(X_i|\theta)$ denote the likelihood function, let $\pi(\theta)$ denote the prior distribution of θ, and let $\log \pi(\theta|X_N) = \log p(X_N|\theta) + \log \pi(\theta)$ denote the log-posterior density function. If θ has a fixed dimension and $\log \pi(\theta|X_N)$ is differentiable with respect to θ, then the stochastic gradient Langevin dynamics algorithm (Welling & Teh, 2011) can be applied to simulate from the posterior, which iterates by

$$\theta_{t+1} = \theta_t + \frac{\epsilon_{t+1}}{2}\widehat{\nabla}_\theta \log \pi(\theta_t|X_N) + \sqrt{(\epsilon_{t+1}\tau)}\eta_{t+1}, \quad \eta_{t+1} \sim N(0, I_d), \tag{1}$$

where d is the dimension of θ, I_d is an $d \times d$-identity matrix, ϵ_{t+1} is the step size (also known as the learning rate), τ is the temperature, and $\widehat{\nabla}_\theta \log \pi(\theta_t|X_N)$ denotes an estimate of $\nabla_\theta \log \pi(\theta_t|X_N)$ based on a mini-batch of samples. The learning rate can be decreasing or kept as a constant. For the former, the convergence of the algorithm was studied in Teh et al. (2016). For the latter, the convergence of the algorithm was studied in Sato & Nakagawa (2014) and Dalalyan & Karagulyan (2017). Refer to Nemeth & Fearnhead (2019) for more discussions on the theory, implementation and variants of this algorithm.

3. AN EXTENDED STOCHASTIC GRADIENT LANGEVIN DYNAMICS ALGORITHM

To extend the applications of the stochastic gradient Langevin dynamics algorithm to varying-dimensional problems such as variable selection and missing data, we first establish an identity for evaluating $\nabla_\theta \log \pi(\theta|X_N)$ in presence of latent variables. As illustrated below, the latent variables can be the model indicator in the variable selection problems or missing values in the missing data problems.

LEMMA 1. *For any latent variable ϑ,*

$$\nabla_\theta \log \pi(\theta \mid X_N) = \int \nabla_\theta \log \pi(\theta \mid \vartheta, X_N)\pi(\vartheta \mid \theta, X_N)d\vartheta, \tag{2}$$

where $\pi(\vartheta \mid \theta, X_N)$ and $\pi(\theta \mid \vartheta, X_N)$ denote the conditional distribution of ϑ and θ, respectively.

Lemma 1 provides us a Monte Carlo estimator for $\nabla_\theta \log \pi(\theta \mid X_N)$ by averaging over the samples drawn from the conditional distribution $\pi(\vartheta|\theta, X_N)$. The identity (2) is similar to Fisher's identity. The latter has been used in evaluating the gradient of the log-likelihood function in presence of latent variables, see e.g. Cappé et al. (2005). When N is large, the computation can be accelerated by subsampling. Let X_n denote a subsample, where n denotes the subsample size. Without loss of generality, we assume that N is a multiple of n, i.e., N/n is an integer. Let $X_{n,N} = \{X_n, \ldots, X_n\}$ denote a duplicated dataset with the subsample, whose total sample size is also N. Following from (2), we have

$$\nabla_\theta \log \pi(\theta \mid X_{n,N}) = \int \nabla_\theta \log \pi(\theta \mid \vartheta, X_{n,N})\pi(\vartheta \mid \theta, X_{n,N})d\vartheta. \tag{3}$$

Since $\nabla_\theta \log \pi(\theta \mid X_{n,N}) = \nabla_\theta \log p(X_{n,N}|\theta) + \nabla_\theta \log \pi(\theta)$ is true and $\log p(X_{n,N}|\theta)$ is unbiased for $\log p(X_N|\theta)$, $\nabla_\theta \log \pi(\theta \mid X_{n,N})$ forms an unbiased estimator of $\nabla_\theta \log \pi(\theta \mid X_N)$. Sampling from $\pi(\gamma_S|\theta, X_{n,N})$ can be much faster than sampling from $\pi(\gamma_S|\theta, X_N)$ as for the former the likelihood only needs to be evaluated on a mini-batch of samples.

3.1. *Bayesian Variable Selection*

As an illustrative example, we consider the problem of variable selection for linear regression

$$Y = z^T\beta + \varepsilon, \tag{4}$$

where ε is a zero-mean Gaussian random error with variance σ^2, $\beta \in \mathbb{R}^p$ is the vector of regression coefficients, and $z = (z_1, z_2, \ldots, z_p)$ is the vector of explanatory variables. Let $\gamma_S = (\gamma_S^1, \ldots, \gamma_S^p)$ be a binary vector indicating the variables included in model S, and let β_S be the vector of regression coefficients associated with the model S. From the perspective of Bayesian statistics, we are interested in estimating the posterior probability $\pi(\gamma_S|X_N)$ for each model $S \in \mathcal{S}$ and the posterior mean $\pi(\rho) = \int \rho(\beta)\pi(\beta|X_N)$ for some integrable function $\rho(\cdot)$, where $\mathcal{S}$ comprises 2^p models. Both quantities can be estimated using the reversible jump Metropolis-Hastings algorithm (Green, 1995) by sampling from the posterior distribution $\pi(\gamma_S, \beta_S|X_N)$. However, when N is large, the algorithm can be extremely slow due to repeated scans of the full dataset in simulations.

As aforementioned, the existing stochastic gradient MCMC algorithms cannot be directly applied to simulate of $\pi(\gamma_S, \beta_S|X_N)$ due to the dimension jumping issue involved in model transition. To address this issue, we introduce an auxiliary variable $\theta = (\theta^1, \theta^2, \ldots, \theta^p)$, which links γ_S and β_S through

$$\beta_S = \theta * \gamma_S = (\theta^1\gamma_S^1, \theta^2\gamma_S^2, \ldots, \theta^p\gamma_S^p), \tag{5}$$

where $*$ denotes elementwise multiplication. Let $\theta_{[S]} = \{\theta^i : \gamma_S^i = 1, i = 1, 2, \ldots, p\}$ and $\theta_{[-S]} = \{\theta^i : \gamma_S^i = 0, i = 1, 2, \ldots, p\}$ be subvectors of θ corresponding to the nonzero and zero elements of γ_S, respectively. Note that β_S is sparse with all elements in $\theta_{[-S]}$ being zero, while θ can be dense. Based on the relation (5), we suggest to simulate from $\pi(\theta|X_N)$ using the stochastic gradient Langevin dynamic algorithm, for which the gradient $\nabla_\theta \log \pi(\theta|X_N)$ can be evaluated using Lemma 1 by treating γ_S as the latent variable. Let $\pi(\theta)$ denote the prior of θ. To simplify the computation of $\nabla_\theta \log \pi(\theta \mid \gamma_S, X_N)$, we further assume the *a priori* independence that $\pi(\theta|\gamma_S) = \pi(\theta_{[S]}|\gamma_S)\pi(\theta_{[-S]}|\gamma_S)$. Then it is easy to derive

$$\nabla_\theta \log \pi(\theta \mid \gamma_S, X_N) = \begin{cases} \nabla_{\theta_{[S]}} \log p(X_N|\theta_{[S]}, \gamma_S) + \nabla_{\theta_{[S]}} \pi(\theta_{[S]}|\gamma_S), & \text{for component } \theta_{[S]}, \\ \nabla_{\theta_{[-S]}} \log \pi(\theta_{[-S]}|\gamma_S), & \text{for component } \theta_{[-S]}, \end{cases}$$

which can be used in evaluating $\nabla \log \pi(\theta|X_N)$ by Lemma 1. If a mini-batch of data is used, the gradient can be evaluated based on (3). This leads to an extended stochastic gradient langevin dynamics algorithm.

Algorithm 1. [Extended Stochastic Gradient Langevin Dynamics for Bayesian Variable Selection]

(i) (Subsampling) Draw a subsample of size n (with or without replacement) from the full dataset X_N at random, and denote the subsample by $X_n^{(t)}$, where t indexes the iteration.

(ii) (Simulating models) Simulate models $\gamma_{S_1,n}^{(t)}, \ldots, \gamma_{S_m,n}^{(t)}$ from the conditional posterior $\pi(\gamma_S|\theta^{(t)}, X_{n,N}^{(t)})$ by running a short Markov chain, where $X_{n,N}^{(t)} = \{X_n^{(t)}, \ldots, X_n^{(t)}\}$ and $\theta^{(t)}$ is the sample of θ at iteration t.

(iii) (Updating θ) Update $\theta^{(t)}$ by setting $\theta^{(t+1)} = \theta^{(t)} + (2m)^{-1}\epsilon_{t+1} \sum_{k=1}^m \nabla_\theta \log \pi(\theta^{(t)}|\gamma_{S_k,n}^{(t)}, X_{n,N}^{(t)}) + \sqrt{(\epsilon_{t+1}\tau)}\eta_{t+1}$, where ϵ_{t+1} is the learning rate, $\eta_{t+1} \sim N(0, I_p)$, τ is the temperature, and p is the dimension of θ.

Theorem 1 justifies the validity of this algorithm with the proof given in the Appendix.

THEOREM 1. *Assume that the conditions (A.1)-(A.3) (given in Appendix) hold, m, p, n are increasing with N such that $N \geq n \succ p$, $m \succ p^{1/2}$, and a constant learning rate $\epsilon \prec 1/N$ is used. Then, as $N \to \infty$,*

(i) $W_2(\pi_t, \pi_*) \to 0$ *as* $t \to \infty$*, where* π_t *denotes the distribution of* $\theta^{(t)}$*,* $\pi_* = \pi(\theta|X_N)$*, and* $W_2(\cdot,\cdot)$ *denotes the second order Wasserstein distance between two distributions.*

(ii) *If* $\rho(\theta)$ *is* α*-Lipschitz for some constant* $\alpha > 0$*, then* $\sum_{t=1}^{T} \rho(\theta^{(t)})/T \xrightarrow{p} \pi_*(\rho)$ *as* $T \to \infty$*, where* $\xrightarrow{p}$ *denotes convergence in probability and* $\pi_*(\rho) = \int_\Theta \rho(\theta)\pi(\theta|X_N)d\theta$*.*

(iii) *If (A.4) further holds,* $\sum_{t=1}^{T}\sum_{i=1}^{m} I(\gamma_{S_i,n}^{(t)} = \gamma_S)/(mT) - \pi(\gamma_S|X_N) \xrightarrow{p} 0$ *as* $T \to \infty$*.*

Part (i) establishes the weak convergence of θ_t; that is, if the total sample size N and the iteration number t are sufficiently large, and the subsample size n and the number of models m simulated at each iteration are reasonably large, then $\pi(\theta_t|X_N)$ will converge to the true posterior $\pi(\theta|X_N)$ in 2-Wasserstein distance. Refer to Gibbs & Su (2002) for discussions on the relation between Wasserstein distance and other probability metrics. Parts (ii) & (iii) address our general interests on how to estimate the posterior mean and posterior probability, respectively, based the samples simulated by Algorithm 1. For parts (i), (ii) and (iii), the explicit convergence rates are given in equations (3), (5) and (10), respectively.

For the choice of $m \succ p^{1/2}$, p can be approximately treated as the maximum size of the models under consideration, which is of the same order as the true model. Therefore, m can be pretty small under the model sparsity assumption. Theorem 1 is established with a constant learning rate. In practice, one may use a decaying learning rate, see e.g. Teh et al. (2016), where it is suggested to set $\epsilon_t = O(1/t^\kappa)$ for some $0 < \kappa \leq 1$. For the decaying learning rate, Teh et al. (2016) recommended some weighted averaging estimators for $\pi_*(\rho)$. Theorem 2 shows that the unweighted averaging estimators used above still work if the learning rate slowly decays at a rate of $\epsilon_t = O(1/t^\kappa)$ for $0 < \kappa < 1$. However, if $\kappa = 1$, the weighted averaging estimators are still needed. The proof of Theorem 2 is given in the supplementary material.

THEOREM 2. *Assume the conditions of Theorem* 1 *hold. If a decaying learning rate* $\epsilon_t = O(1/t^\kappa)$ *is used for some* $0 < \kappa < 1$*, then parts (i), (ii) and (iii) of Theorem* 1 *are still valid.*

3.2. *Missing Data*

Missing data are ubiquitous over all fields from science to technology. However, under the big data scenario, how to conduct Bayesian analysis in presence of missing data is still unclear. The existing data-augmentation algorithm (Tanner & Wong, 1987) is full data based and thus can be extremely slow. In this context, we let X_N denote the incomplete data and let θ denote the model parameters. If we treat the missing values as latent variables, then Lemma 1 can be used for evaluating the gradient $\nabla_\theta \log \pi(\theta|X_N)$. However, Algorithm 1 cannot be directly applied to missing data problems, since the imputation of the missing data might depend on the subsample only. To address this issue, we propose Algorithm S1 (given in the Supplementary material), where the missing values ϑ are imputed from $\pi(\vartheta|\theta, X_n)$ at each iteration. Theorem 1 and Theorem 2 are still applicable to this algorithm.

4. AN ILLUSTRATIVE EXAMPLE

This section illustrates the performance of Algorithm 1 using a simulated example. More numerical examples are presented in the supplementary material. Ten synthetic datasets were generated from the model (4) with $N = 50,000$, $p = 2001$, $\sigma^2 = 1$, $\beta_1 = \cdots = \beta_5 = 1$, $\beta_6 = \beta_7 = \beta_8 = -1$, and $\beta_0 = \beta_9 = \cdots = \beta_p = 0$, where σ^2 is assumed to be known, and the explanatory variables are normally distributed with a mutual correlation coefficient of 0.5. A hierarchical prior was assumed for the model and parameters with the detail given in the supplementary material. For each dataset, Algorithm 1 was run for 5000 iterations with $n = 200$, $m = 10$, and the learning rate $\epsilon_t \equiv 10^{-6}$, where the first 2000 iterations were discarded for the burn-in process and the samples generated from the remaining iterations were used for inference. At each iteration, the reversible jump Metropolis-Hastings algorithm (Green, 1995) was used for simulating the models $\gamma_{S_i,n}^{(t)}$, $i = 1, 2, \ldots, m$ with the detail given in the supplementary material.

Table 1 summarizes the performance of the algorithm, where the false selection rate (FSR), negative selection rate (NSR), mean squared errors for false predictors (MSE_0) and mean squared errors for true predictors (MSE_1) are defined in the supplementary material. The variables were selected according to the median posterior probability rule (Barbieri & Berger, 2004), which selects only the variables with the

Table 1. *Bayesian variable selection with the extended stochastic gradient Langevin dynamics (eSGLD), reversible jump Metropolis-Hastings (RJMH), split-and-merge (SaM) and Bayesian Lasso (B-Lasso) algorithms, where FSR, NSR, MSE_1 and MSE_0 are reported in averages over 10 datasets with standard deviations given in the parentheses, and the CPU time (in minutes) was recorded for one dataset on a Linux machine with Intel® Core™i7-3770 CPU@3.40GHz.*

Algorithm	FSR	NSR	MSE_1	MSE_0	CPU(m)
eSGLD	0(0)	0(0)	$2.91 \times 10^{-3}(1.90 \times 10^{-3})$	$1.26 \times 10^{-7}(1.18 \times 10^{-8})$	3.3
RJMH	0.50(0.10)	0.16(0.042)	$1.60 \times 10^{-1}(3.89 \times 10^{-2})$	$2.64 \times 10^{-5}(8.75 \times 10^{-6})$	180.1
SaM	0.05(0.05)	0.013(0.013)	$1.29 \times 10^{-2}(1.27 \times 10^{-2})$	$1.01 \times 10^{-6}(1.00 \times 10^{-6})$	150.4
B-Lasso	0(0)	0(0)	$2.32 \times 10^{-4}(3.58 \times 10^{-5})$	$1.40 \times 10^{-7}(5.08 \times 10^{-9})$	32.8

marginal inclusion probability greater than 0.5. The Bayesian estimates of parameters were obtained by averaging over a set of thinned (by a factor of 10) posterior samples. For comparison, some existing algorithms were applied to this example with the results given in Table 1 and the implementation details given in the supplementary material. The comparison show that the proposed algorithm has much alleviated the pain of Bayesian methods in big data analysis.

5. Discussion

This paper has extended the stochastic gradient Langevin dynamics algorithm to general large-scale Bayesian computing problems, such as those involving dimension jumping and missing data. To the best of our knowledge, this paper provides the first Bayesian method and theory for high-dimensional discrete parameter estimation with mini-batch samples, while the existing methods work for continuous parameters or very low dimensional discrete problems only. Other than generalized linear models, the proposed algorithm can have many applications in data science. For example, it can be used for sparse deep learning and accelerating computation for statistical models/problems where latent variables are involved, such as hidden Markov models, random coefficient models, and model-based clustering problems.

Algorithm 1 can be further extended by updating θ using a variant of stochastic gradient Langevin dynamics, such as stochastic gradient Hamiltonian Monte Carlo (Chen et al., 2014), stochastic gradient thermostats (Ding et al., 2014), stochastic gradient Fisher scoring (Ahn et al., 2012), or preconditioned stochastic gradient Langevin dynamics (Li et al., 2016). We expect that the advantages of these variants (over stochastic gradient Langevin dynamics) can be carried over to the extension.

Acknowledgements

This work was partially supported by the grants DMS-1811812, DMS-1818674, and R01-GM126089. The authors thank the editor, associate editor and referees for their insightful comments/suggestions.

Appendix

A.1. *Proof of Lemma* 1

Proof. Let $\pi(\theta)$ denote the prior density of θ, and let $\pi(\vartheta)$ denote the density of ϑ. Then

$$\nabla_\theta \log \pi(\theta|X_N) = \nabla_\theta \log p(X_N \mid \theta) + \nabla_\theta \log \pi(\theta) = \frac{1}{p(X_N \mid \theta)} \nabla_\theta \int p(X_N, \vartheta \mid \theta) d\vartheta + \nabla_\theta \log \pi(\theta)$$

$$= \int \frac{p(X_N, \vartheta \mid \theta)}{p(X_N \mid \theta)} \nabla_\theta \log p(X_N, \vartheta \mid \theta) d\vartheta + \nabla_\theta \log \pi(\theta)$$

$$= \int \pi(\vartheta \mid \theta, X_N) \nabla_\theta \left[\log p(X_N \mid \theta, \vartheta) + \log \pi(\theta \mid \vartheta) + \log \pi(\vartheta) - \log \pi(\theta)\right] d\vartheta + \nabla_\theta \log \pi(\theta)$$

$$= \int \nabla_\theta \log \pi(\theta \mid \vartheta, X_N) \pi(\vartheta \mid \theta, X_N) d\vartheta,$$

where the second and third equalities follow from the relation $\nabla_\theta \log(g(\theta)) = \nabla_\theta g(\theta)/g(\theta)$ (for an appropriate function $g(\theta)$), and the fourth and fifth equalities are by direct calculations of the conditional distributions. □

A.2. *Proof of Theorem* 1

Let $\pi_* = \pi(\theta|X_N)$ denote the posterior density function of θ, and let $\pi_t = \pi(\theta^{(t)}|X_N)$ denote the density of $\theta^{(t)}$ generated by Algorithm 1 at iteration t. We are interested in studying the discrepancy between π_* and π_t in the 2nd order Wasserstein distance. The following conditions are assumed.

(A.1) The posterior π_* is strongly log-concave and gradient-Lipschitz:

$$f(\theta) - f(\theta') - \nabla f(\theta')^T(\theta - \theta') \geq \frac{q_N}{2}\|\theta - \theta'\|_2^2, \quad \forall \theta, \theta' \in \Theta, \tag{1}$$

$$\|\nabla f(\theta) - \nabla f(\theta')\|_2 \leq Q_N\|\theta - \theta'\|_2, \quad \forall \theta, \theta' \in \Theta, \tag{2}$$

where $f(\theta) = -\log \pi(\theta|X_N)$, and $c_0' N \leq q_N \leq Q_N \leq c_0 N$ for some positive constants c_0 and c_0'.

(A.2) The posterior π_* has bounded second moment: $\int_\Theta \theta^T\theta\pi_*(\theta)d\theta = O(p)$.

(A.3) $\max_{S\in\mathcal{S}} E_{X_N}[\|\nabla_\theta \log \pi(\theta|\gamma_S, X_N)\|^2|\theta] = O(N^2(\|\theta\|^2 + p))$, where E_{X_N} denotes expectation with respect to the distribution of X_N, and $\mathcal{S}$ denotes the set of all possible models.

(A.4) Let $L_N(\gamma_S, \theta) = \log p(X_N|\gamma_S, \theta)/N$ and let $\{L_N^{(i)}(\theta) : i = 1, 2, \ldots, |\mathcal{S}|\}$ be the descending order statistics of $\{L_N(\gamma_S, \theta) : S \in \mathcal{S}\}$. Assume that there exists a constant $\delta > 0$ such that $\inf_{\theta\in\Theta}(L_N^{(1)}(\theta) - L_N^{(2)}(\theta)) \geq \delta$.

Proof. Part (i). In Algorithm 1, the gradient $\nabla \log \pi(\theta^{(t)}|X_N)$ is estimated by running a short Markov chain with a mini-batch of data. Since the initial distribution of the Markov chain might not coincide with its equilibrium distribution, the resulting gradient estimate can be biased. Let $\zeta^{(t)} = \frac{1}{m}\sum_{k=1}^m \nabla_\theta \log \pi(\theta^{(t)}|\gamma_{S_k,n}^{(t)}, X_{n,N}^{(t)}) - \nabla \log \pi(\theta^{(t)}|X_N)$. Following from (A.3), we have

$$\|E(\zeta^{(t)}|\theta^{(t)})\|^2 = O\left[\frac{N^2(\|\theta^{(t)}\|^2 + p)}{m^2}\right], \; E\|\zeta^{(t)} - E(\zeta^{(t)}|\theta^{(t)})\|^2 = O\left[\frac{N^2(\|\theta^{(t)}\|^2 + p)}{mn}\right].$$

Following from Lemma S2 in the supplementary material, if $m \succ p^{1/2}$, $\epsilon \prec 1/N \prec (mn)/(Np)$, and $V = O(p)$ holds, then

$$W_2(\pi_t, \pi_*) = (1-\omega)^t W_2(\pi_0, \pi_*) + O\big(\frac{p^{1/2}}{m}\big) + O((\epsilon p)^{1/2}) + O\big(\big(\frac{\epsilon N p}{mn}\big)^{1/2}\big) \to 0, \quad \text{as } t \to \infty, \tag{3}$$

for some $\omega > 0$, since $q_N \asymp N$ and $Q_N \asymp N$ hold by conditions (A.1) and (A.2).

Part (ii). Since $\rho(\theta)$ is α-Lipschitz, we have $|\rho(\theta)| \leq \alpha\|\theta\| + C'$ for some constant C'. Further, π_* is strongly log-concave, so $\pi_*(|\rho|) < \infty$, i.e., ρ is π_*-integrable. On the other hand,

$$\begin{aligned} &\|\int \rho(\theta)d\pi_*(\theta) - \int \rho(\tilde\theta)d\pi_t(\tilde\theta)\| = \|E\rho(\theta) - E\rho(\tilde\theta)\| \leq E\|\rho(\theta) - \rho(\tilde\theta)\| \\ \leq &\alpha E\|\theta - \tilde\theta\|_2 \leq \alpha\{E\|\theta - \tilde\theta\|_2^2\}^{1/2} = \alpha W_2(\pi_*, \pi_t) = o(1), \quad \text{(due to eq. (3)).} \end{aligned} \tag{4}$$

where θ and $\tilde\theta$ are two random variables whose marginal distributions follow π_* and π_t respectively, $E(\cdot)$ denotes expectation with respect to the joint distribution of θ and $\tilde\theta$, and $(E\|\theta - \theta_t\|_2^2)^{1/2} = W_2(\pi_*, \pi_t)$. This implies that ρ is also π_t-integrable and $\int \rho(\tilde\theta)d\pi_t(\tilde\theta) \to \int \rho(\theta)d\pi_*(\theta)$ as $t \to \infty$.

Further, by the property of Markov chain, WLLN applies and thus $\sum_{t=1}^T \rho(\theta^{(t)})/T - \sum_{t=1}^T \int \rho(\tilde\theta)d\pi_t(\tilde\theta)/T = O_p(T^{-1/2})$. Combining it with the above result leads to

$$\sum_{t=1}^T \rho(\theta^{(t)})/T - \pi_*(\rho) = O_p(T^{-1/2}) + \alpha\sum_{t=1}^T W_2(\pi_*, \pi_t)/T \to 0. \tag{5}$$

Part (iii). To establish the convergence of $\hat{\pi}(\gamma_S|X_N)$, we define $L_N(\gamma_S,\theta^{(t)}) = \log p(X_N|\gamma_S,\theta^{(t)})/N$, $L_n(\gamma_S,\theta^{(t)}) = \log p(X_n^{(t)}|\gamma_S,\theta^{(t)})/n$, and $\xi_{n,S}^{(t)} = L_n(\gamma_S,\theta^{(t)}) - L_N(\gamma_S,\theta^{(t)})$ for any $S \in \mathcal{S}$. For each S, $\xi_{n,S}^{(t)}$ is approximately Gaussian with $E(\xi_{n,S}^{(t)}) = 0$ and $\text{Var}(\xi_{n,S}^{(t)}) = O(1/n)$. Therefore, for any positive ν, with probability $1 - |\mathcal{S}|^{-\nu}$, $\max_S |\xi_{n,S}|$ is bounded by $\delta_n := \{(2\nu+2)\log|\mathcal{S}|/n\}^{1/2} = O[\{(\nu+1)p/n\}^{1/2}]$ according to the tail probability of the Gaussian. It implies, with high probability, that if S is the most likely model, i.e., $L_N^{(t)}(\gamma_S) = L_N^{(1)}(\theta^{(t)})$, then

$$
\begin{aligned}
&|\pi(\gamma_S|X_{n,N}^{(t)},\theta^{(t)}) - \pi(\gamma_S|X_N,\theta^{(t)})| \\
=&\Big|\frac{1}{1+\sum_{S'\neq S} e^{N(L_N(\gamma_{S'},\theta^{(t)})-L_N(\gamma_S,\theta^{(t)})+\xi_{n,S'}-\xi_{n,S})}} - \frac{1}{1+\sum_{S'\neq S} e^{N(L_N(X_N|\gamma_{S'},\theta^{(t)})-L_N(X_N|\gamma_S,\theta^{(t)}))}}\Big| \\
=&\frac{\sum_{S'\neq S} e^{N(L_N(X_N|\gamma_{S'},\theta^{(t)})-L_N(X_N|\gamma_S,\theta^{(t)}))+b_{S'})}}{[1+\sum_{S'\neq S} e^{N(L_N(X_N|\gamma_{S'},\theta^{(t)})-L_N(X_N|\gamma_S,\theta^{(t)})+b_{S'})}]^2} N|\xi_{n,S'} - \xi_{n,S}| \\
\leq&(2^p-1)e^{-N(\delta-2\delta_n)}N2\delta_n \leq e^{-N\delta/2} \to 0,
\end{aligned}
$$

if $\nu p \prec n$ (i.e., $\delta_n \prec \delta$) and $N \succ p$, where the second equality follows from the mean-value theorem by viewing $N(L_N(X_N|\gamma_{S'},\theta^{(t)}) - L_N(X_N|\gamma_S,\theta^{(t)}))$'s as the arguments of $\pi(\gamma_S|X_N,\theta^{(t)})$, and $b_{S'}$ denotes a value between 0 and $(\xi_{n,S'} - \xi_{n,S})$. Similarly, if S is not the most likely model, then we denote S^* as the most likely model and, by the mean-value theorem,

$$
\begin{aligned}
&|\pi(\gamma_S|X_{n,N}^{(t)},\theta^{(t)}) - \pi(\gamma_S|X_N,\theta^{(t)})| \\
=&\left|\frac{e^{N(L_N(\gamma_S,\theta^{(t)})-L_N(\gamma_{S^*},\theta^{(t)})+\xi_{n,S}-\xi_{n,S^*})}}{1+\sum_{S'\neq S} e^{N(L_N(\gamma_{S'},\theta^{(t)})-L_N(\gamma_{S^*},\theta^{(t)})+\xi_{n,S'}-\xi_{n,S^*})}} - \frac{e^{N(L_N(\gamma_S,\theta^{(t)})-L_N(\gamma_{S^*},\theta^{(t)}))}}{1+\sum_{S'\neq S} e^{N(L_N(\gamma_{S'},\theta^{(t)})-L_N(\gamma_{S^*},\theta^{(t)})}}\right| \\
\leq&[1+(2^p-1)e^{-N(\delta-2\delta_n)} + e^{2N\delta_n}]e^{-N(\delta-2\delta_n)}N2\delta_n \leq e^{-N\delta/2} \to 0.
\end{aligned}
$$

In conclusion, with probability $1 - 1/|\mathcal{S}|^\nu$, $|\pi(\gamma_S|X_{n,N}^{(t)},\theta^{(t)}) - \pi(\gamma_S|X_N,\theta^{(t)})| < \exp(-N\delta/2)$ for all S, any iteration t and any $\theta^{(t)} \in \Theta$. Then, one could choose some $\nu = (n/p)^{1/2} \to \infty$, such that $\pi(\gamma_S|X_{n,N}^{(t)},\theta^{(t)}) - \pi(\gamma_S|X_N,\theta^{(t)})$ is bounded by

$$\max_S E|\pi(\gamma_S|X_{n,N}^{(t)},\theta^{(t)}) - \pi(\gamma_S|X_N,\theta^{(t)})| \leq \exp(-N\delta/2) + 1/|\mathcal{S}|^\nu \to 0, \tag{6}$$

for any iteration t. Conditioned on $\{\theta^{(t)} : t = 1,2,\ldots\}$, $[\pi(\gamma_S|X_{n,N},\theta^{(t)}) - \pi(\gamma_S|X_N,\theta^{(t)})]$'s are independent and each is bounded by 1, so WLLN applies. Therefore, for any $S \in \mathcal{S}$, by WLLN,

$$\frac{1}{T}\sum_{t=1}^T \pi(\gamma_S|X_{n,N}^{(t)},\theta^{(t)}) - \frac{1}{T}\sum_{t=1}^T \pi(\gamma_S|X_N,\theta^{(t)}) = O_p(T^{-1/2}) + \exp(-N\delta/2) + 1/|\mathcal{S}|^\nu \to 0, \tag{7}$$

provided $p \prec n \leq N$. Since $\{\theta^{(t)} : t = 1,2,\ldots\}$ forms a time-homogeneous Markov chain, whose convergence is measured by (3), and the function $\pi(\gamma_S|X_N,\theta)$ is bounded and continuous in θ,

$$\frac{1}{T}\sum_{t=1}^T \pi(\gamma_S|X_N,\theta^{(t)}) - \pi(\gamma_S|X_N) = O_p(T^{-1/2}), \tag{8}$$

holds for any $S \in \mathcal{S}$. Combining (8) with (7) leads to

$$\frac{1}{T}\sum_{t=1}^T \pi(\gamma_S|X_{n,N}^{(t)},\theta^{(t)}) - \pi(\gamma_S|X_N) = O_p(T^{-1/2}) + \exp(-N\delta/2) + 1/|\mathcal{S}|^\nu \to 0. \tag{9}$$

Conditioned on $X_{n,N}^{(t)}$ and $\theta^{(t)}$, by the standard theory of MCMC, $m^{-1}\sum_{i=1}^m I(\gamma_S^{(t,i)} = \gamma_S)$ forms a consistent estimator of $\pi(\gamma_S|X_{n,N}^{(t)},\theta^{(t)})$ with an asymptotic bias of $O(1/m)$. Since m is increasing with

p and N, the estimator is asymptotically unbiased. Combining this result with (9) leads to

$$\frac{1}{mT}\sum_{t=1}^{T}\sum_{i=1}^{m} I(\gamma_{S_{i,n}}^{(t)} = \gamma_S) - \pi(\gamma_S|X_N) = O_p(T^{-1/2}) + \exp(-N\delta/2) + 1/|\mathcal{S}|^{\nu} + O_p(m^{-1/2}), \tag{10}$$

which converges to 0 as $T \to \infty$ and $N \to \infty$. □

REFERENCES

AHN, S., BALAN, A. K. & WELLING, M. (2012). Bayesian posterior sampling via stochastic gradient Fisher scoring. In *ICML*.

BARBIERI, M. & BERGER, J. (2004). Optimal predictive model selection. *Annals of Statistics* **32**, 870–897.

BARDENET, R., DOUCET, A. & HOLMES, C. (2014). Towards scaling up Markov chain Monte Carlo: an adaptive subsampling approach. In *ICML*.

BARDENET, R., DOUCET, A. & HOLMES, C. C. (2017). On Markov chain Monte Carlo methods for tall data. *Journal of Machine Learning Research* **18**, 47:1–47:43.

BETANCOURT, M. (2015). The fundamental incompatibility of scalable Hamiltonian Monte Carlo and naive data subsampling. In *ICML*.

BIERKENS, J., FEARNHEAD, P. & ROBERTS, G. (2019). The zig-zag process and super-efficient Monte Carlo for Bayesian analysis of big data. *Annals of Statistics* **47**, 1288–1320.

BOUCHARD COTÉ, A., VOLLMER, S. & DOUCET, A. (2018). The bouncy particle sampler: A nonreversible rejection-free Markov chain Monte Carlo method. *Journal of the American Statistical Association* **113**, 855–867.

CAPPÉ, O., MOULINES, E. & RYDEN, T. (2005). *Inference in Hidden Markov Models*. New York: Springer.

CHEN, H., SEITA, D., PAN, X. & CANNY, J. (2016). An efficient minibatch acceptance test for Metropolis-Hastings. *arXiv:1610.06848* .

CHEN, T., FOX, E. B. & GUESTRIN, C. (2014). Stochastic gradient Hamiltonian Monte Carlo. In *ICML*.

CHEN, Y. & GHAHRAMANI, Z. (2016). Scalable discrete sampling as a multi-armed bandit problem. In *ICML*.

DALALYAN, A. S. & KARAGULYAN, A. G. (2017). User-friendly guarantees for the Langevin Monte Carlo with inaccurate gradient. *CoRR* **abs/1710.00095**.

DING, N., FANG, Y., BABBUSH, R., CHEN, C., SKEEL, R. D. & NEVEN, H. (2014). Bayesian sampling using stochastic gradient thermostats. In *NIPS*.

GIBBS, A. & SU, F. (2002). On choosing and bounding probability metrics. *International Statistical Review* **70**, 419–435.

GREEN, P. (1995). Reversible jump Markov chain Monte Carlo computation and Bayesian model determination. *Biometrika* **82**, 711–732.

KORATTIKARA, A., CHEN, Y. & WELLING, M. (2014). Austerity in MCMC land: Cutting the Metropolis-Hastings budget. In *ICML*.

LI, C., CHEN, C., CARLSON, D. E. & CARIN, L. (2016). Preconditioned stochastic gradient Langevin dynamics for deep neural networks. In *AAAI*.

MA, Y.-A., CHEN, T. & FOX, E. B. (2015). A complete recipe for stochastic gradient MCMC. In *NIPS*.

MACLAURIN, D. & ADAMS, R. P. (2014). Firefly Monte Carlo: Exact MCMC with subsets of data. In *IJCAI*.

NEMETH, C. & FEARNHEAD, P. (2019). Stochastic gradient Markov chain Monte Carlo. *arXiv:1907.06986* .

SATO, I. & NAKAGAWA, H. (2014). Approximation analysis of stochastic gradient Langevin dynamics by using Fokker-Planck equation and Ito process. In *ICML*.

SCOTT, S. L., BLOCKER, A. W., BONASSI, F. V., CHIPMAN, H. A., GEORGE, E. I. & MCCULLOCH, R. E. (2016). Bayes and big data: The consensus Monte Carlo algorithm. *International Journal of Management Science and Engineering Management* **11**, 78–88.

SRIVASTAVA, S., LI, C. & DUNSON, D. B. (2018). Scalable Bayes via Barycenter in Wasserstein space. *Journal of Machine Learning Research* **19**, 1–35.

TANNER, M. & WONG, W. (1987). The calculation of posterior distributions by data augmentation (with Discussions). *Journal of the American Statistical Association* **82**, 528–540.

TEH, W., THIERY, A. & VOLLMER, S. (2016). Consistency and fluctuations for stochastic gradient Langevin dynamics. *Journal of Machine Learning Research* **17**, 1–33.

WELLING, M. & TEH, Y. W. (2011). Bayesian learning via stochastic gradient Langevin dynamics. In *ICML*.

XUE, J. & LIANG, F. (2019). Double-parallel Monte Carlo for Bayesian analysis of big data. *Statistics and Computing* **29**, 23–32.

Biometrika (2019), *pp*. 1–12
Printed in Great Britain

Supplementary Material for "Extended Stochastic Gradient MCMC for Large-Scale Bayesian Variable Selection"

BY QIFAN SONG, YAN SUN, MAO YE, FAMING LIANG
Department of Statistics, Purdue University,
West lafayette, IN 47906, USA
qfsong, sun748, ye207, fmliang@purdue.edu

This material is organized as follows. Section 1 presents an extended stochastic gradient Langevin dynamics algorithm for missing data problems. Section 2 presents some lemmas and and the proof of Theorem 2. Section 3 presents the reversible jump Metropolis-Hastings (RJMH) algorithm used in Algorithm 1 (of the main paper) for Bayesian variable selection. Section 4 presents more numerical examples.

1. AN EXTENDED STOCHASTIC GRADIENT LANGEVIN DYNAMICS ALGORITHM FOR MISSING DATA PROBLEMS

As mentioned in the main paper, Algorithm 1 cannot be directly applied to missing data problems, because the imputation of the missing data ϑ might depend on the subsamples only. To address this issue, we propose Algorithm S1, where the missing values (denoted by ϑ) are imputed from $\pi(\vartheta|\theta^{(t)}, X_n)$ at each iteration t.

*Algorithm S*1. [Extended Stochastic Gradient Langevin Dynamics for missing data problems]

(i) (Subsampling) Draw a subsample of size n from the full dataset X_N at random, and denote the subsample by $X_n^{(t)}$, where t indexes the iteration.
(ii) (Multiple Imputation) For the subsamples, simulate the missing values $\vartheta_{1,n}^{(t)}, \ldots, \vartheta_{m,n}^{(t)}$ from the conditional posterior $\pi(\vartheta|\theta^{(t)}, X_n^{(t)})$, where $\theta^{(t)}$ is the sample of θ at iteration t.
(iii) (Importance Resampling) Resample an imputed value from the set $\{\vartheta_{1,n}^{(t)}, \ldots, \vartheta_{m,n}^{(t)}\}$ according to the importance weights $\{p(X_n|\theta^{(t)}, \vartheta_{1,n}^{(t)})^{N/n-1}, \ldots, p(X_n|\theta^{(t)}, \vartheta_{m,n}^{(t)})^{N/n-1}\}$. Denote the resampled value by $\vartheta^{(t)}$.
(iv) (Updating θ) Update $\theta^{(t)}$ by setting

$$\theta^{(t+1)} = \theta^{(t)} + \frac{\epsilon_{t+1}}{2}\left\{\frac{N}{mn}\sum_{k=1}^{m}\nabla_\theta \log \pi(\theta^{(t)}|\vartheta_{k,n}^{(t)}, X_n^{(t)}) - \frac{N-n}{n}\nabla_\theta \log \pi(\theta^{(t)})\right\} + \sqrt{(\epsilon_{t+1}\tau)}\eta_{t+1},$$

where $\eta_{t+1} \sim N(0, I_d)$, and ϵ_{t+1} is the learning rate.

In the case that ϑ or some components of ϑ depend on all data X_N, an importance resampling step can be included for simulating samples from $\pi(\vartheta|\theta^{(t)}, X_{n,N})$ and then the corresponding inference can be made. For example, if Algorithm S1 is used for Bayesian variable selection where the model is treated as latent variable, such a step is needed. For the problems that ϑ depends on the subsamples only, this step can be omitted. The validity of Algorithm S1 follows directly from the fact that $N/n\nabla_\theta \log \pi(\theta|X_n) - (N-n)/n\nabla_\theta \log \pi(\theta)$ is an unbiased estimator of $\nabla_\theta \log \pi(\theta|X_N)$.

2. LEMMAS AND PROOF OF THEOREM 2

2.1. *Some Lemmas*

LEMMA S1. *Let $X \sim \mu$ and $Y \sim \nu$, then*

$$E\|Y\|_2^2 \leq E\|X\|_2^2 + W_2^2(\mu,\nu) + 2W_2(\mu,\nu)\{E\|X\|_2^2\}^{1/2},$$

where $W_2(\cdot,\cdot)$ denotes the second order Wasserstein distance.

Proof. By definition of the Wasserstein metric, without loss of generality, we assume that X and Y satisfy $E\|X-Y\|_2^2 = W_2^2(\mu,\nu)$. Then

$$\begin{aligned}
&[E\|Y\|_2^2 - E\|X\|_2^2] - W_2^2(\mu,\nu)\\
=&EY^TY - EX^TX - EX^TX - EY^TY + 2EX^TY\\
=&2EX^TY - 2EX^TX = 2EX^T(Y-X) \leq 2E\|X\|_2\|Y-X\|_2\\
\leq&2\{E\|X\|_2^2 E\|Y-X\|_2^2\}^{1/2} = 2W_2(\mu,\nu)\{E\|X\|_2^2\}^{1/2}.
\end{aligned}$$

Lemma S2 is a generalization of Theorem 4 of Dalalyan & Karagulyan (2017). Compared to Theorem 4 of Dalalyan & Karagulyan (2017), Lemma S2 allows the noise of the stochastic gradients to scale with $\|\theta^{(t-1)}\|^2$. A similar result can be found in Bhatia et al. (2019).

LEMMA S2. *Let $\theta^{(t-1)}$ and $\theta^{(t)}$ be two random vectors in Θ satisfying*

$$\theta^{(t)} = \theta^{(t-1)} - \epsilon[\nabla f(\theta^{(t-1)}) + \zeta^{(t-1)}] + \sqrt(2\epsilon)e^{(t)},$$

where $\zeta^{(t-1)}$ denotes the independent random error of the gradient estimate which can depend on $\theta^{(t-1)}$, and $e^{(t)} \sim N(0, I_p)$. Let π_t be the distribution of $\theta^{(t)}$, and let $\pi_* \propto \exp\{-f\}$ be the target distribution. If $\zeta^{(t)}$ satisfies

$$\|E(\zeta^{(t-1)}|\theta^{(t-1)})\|^2 \leq \eta^2(\sqrt{p} + \|\theta^{(t-1)}\|)^2, \quad E[\|\zeta^{(t-1)} - E(\zeta^{(t-1)}|\theta^{(t-1)})\|^2] \leq \sigma^2(p + \|\theta^{(t-1)}\|^2),$$

for some constants η and σ which might depend on m, n, p and N, and $\zeta^{(t-1)}$'s are independent of $e^{(t)}$'s, and if the conditions (A.1) and (A.2) (given in the main paper) are satisfied, $q_N > \eta + \sqrt(2)\sigma$ and the learning rate $\epsilon \leq \min\{2/(q_N + Q_N), 1/q_N)\}$, then

$$\begin{aligned}
W_2(\pi_t,\pi_*) \leq& [1 - (q_N - \eta - \sqrt(2)\sigma)\epsilon]^t W_2(\pi_0,\pi_*) + \frac{\eta(\sqrt{p} + \sqrt{V})}{q_N + \eta + \sqrt(2)\sigma} + 1.65\frac{Q_N\sqrt(\epsilon p)}{q_N + \eta + \sqrt(2)\sigma}\\
&+ \frac{\sqrt(\epsilon)\sigma^2(p+2V)}{1.65Q_N p^{1/2} + \{q_N + \eta + \sqrt(2)\sigma\}^{1/2}\{\sigma^2(p+2V)\}^{1/2}},
\end{aligned} \tag{S1}$$

where $V = \int \|\theta\|^2\pi_*(\theta)d\theta$.

Proof. By the same arguments as used in the proof of Theorem 4 of Dalalyan & Karagulyan (2017), we have

$$\begin{aligned}
W_2^2(\pi_{k+1},\pi_*) \leq& \Big[(1 - q_N\epsilon)W_2(\pi_k,\pi_*) + 1.65Q_N(\epsilon^3 p)^{1/2} + \epsilon\eta\sqrt{p} + \epsilon\eta\{E\|\theta^{(k)}\|^2\}^{1/2}\Big]^2\\
&+ \epsilon^2\sigma^2 p + \epsilon^2\sigma^2 E\|\theta^{(k)}\|^2.
\end{aligned}$$

By Lemma S1, $E\|\theta^{(k)}\|^2 \leq (W_2(\pi_k, \pi_*) + \sqrt{V})^2$, we can derive that

$$\begin{aligned}
W_2^2(\pi_{k+1}, \pi_*) &\leq \Big[(1 - q_N\epsilon)W_2(\pi_k, \pi_*) + 1.65Q_N(\epsilon^3 p)^{1/2} + \epsilon\eta\sqrt{p} + \epsilon\eta(W_2(\pi_k, \pi_*) + \sqrt{V})\Big]^2 \\
&\quad + \epsilon^2\sigma^2 p + \epsilon^2\sigma^2(W_2(\pi_k, \pi_*) + \sqrt{V})^2 \\
&\leq \Big[(1 - q_N\epsilon + \eta\epsilon)W_2(\pi_k, \pi_*) + 1.65Q_N(\epsilon^3 p)^{1/2} + \epsilon\eta(\sqrt{p} + \sqrt{V})\Big]^2 \\
&\quad + \epsilon^2\sigma^2(p + 2V) + 2\epsilon^2\sigma^2 W_2^2(\pi_k, \pi_*) \\
&\leq \Big[(1 - q_N\epsilon + \eta\epsilon + \sqrt{(2)}\sigma\epsilon)W_2(\pi_k, \pi_*) + 1.65Q_N(\epsilon^3 p)^{1/2} + \epsilon\eta(\sqrt{p} + \sqrt{V})\Big]^2 \\
&\quad + \epsilon^2\sigma^2(p + 2V).
\end{aligned}$$

The proof can then be concluded by Lemma 1 of Dalalyan & Karagulyan (2017). □

LEMMA S3. *Consider a stochastic gradient Langevin dynamics algorithm,*

$$\theta^{(t)} = \theta^{(t-1)} + \frac{\epsilon_t}{2}[F(\theta^{(t-1)}) + \zeta^{(t-1)}] + \sqrt{(\epsilon_t)}e^{(t)}, \quad e^{(t)} \sim \textit{Normal}(0, 1),$$

for some smooth function F, where the independent random error term $\zeta^{(t-1)}$ can depend on $\theta^{(t-1)}$ and $E\zeta^{(t-1)} = 0$. If the learning rate satisfies $\lim_t \epsilon_t = 0$, $\sum_{t=1}^{\infty} \epsilon_t = \infty$, $\sum_{t=1}^{\infty} |1/\epsilon_{t+1} - 1/\epsilon_t|/t \leq \infty$, and $\sum(\epsilon_t t^2)^{-1} \leq \infty$, and furthermore, there exists a Lyapunov function $V(\theta)$, which tends to infinity as $\|\theta\| \to \infty$, is twice differentiable with bounded second derivatives and satisfies the following conditions:

$$\langle F(\theta), \theta\rangle \leq -r\|\theta\| \quad \textit{for all } \|\theta\| > M \textit{ and some } r, M > 0, \tag{S2}$$

$$E\|\zeta^{(t-1)}\|^{2k} \leq V^k(\theta^{(t-1)}) \quad \textit{for some } k \geq 2, \tag{S3}$$

$$\|\nabla V(\theta)\|^2 + \|F(\theta)\|^2 \leq V(\theta), \tag{S4}$$

$$\langle \nabla V(\theta), F(\theta)\rangle \leq -\alpha V(\theta) + \beta \quad \textit{for some } \alpha, \beta > 0, \tag{S5}$$

then the following results hold:

1. *There exists a unique invariant distribution, π, for the diffusion process $d\theta_t = (1/2)F(\theta_t)dt + dW_t$.*
2. *For any function h such that $|h|/V^{k'}$ is bounded for some $k' \leq k/2$,*

$$\lim \frac{1}{T}\sum_{i=1}^{T} h(\theta^{(i)}) = E_\pi h(\theta), \textit{ a.s.}$$

Proof. Part 1: Condition (S2) ensures existence of the invariant distribution by proposition 4.1 of Douc et al. (2009).

Part 2: The convergence of the sample path average is due to Theorem 7 of Teh et al. (2016). Note that Theorem 7 of Teh et al. (2016) shows the convergence of the weighted average $\sum_{i=1}^{T} w_i\theta^{(t)}/\sum_{i=1}^{T} w_i$ with $w_i \to 0$ and $F = \nabla f$ for some f, but its proof is applicable to our case (with a general drift F and $w_i \equiv 1$ for all i) as well. □

2.2. *Proof of Theorem 2*

Proof. (i) Without loss of generality, we let $\epsilon_t = 1/t^\kappa$ where $\kappa \in (0, 1)$. Define $\chi = \kappa/(1 - \kappa)$, and let T_0 be the index such that $\max\{q_N, (Q_N + q_N)/2\}\epsilon_{T_0} < 1$. Let $T_0 \leq T_1 < T_2 < \dots$ be a sequence of integers satisfying $(s_0)^{-\chi} \leq \epsilon_{T_1} < \epsilon_{T_2-1} \leq (s_0 + 1)^{-\chi} \leq \epsilon_{T_2} < \epsilon_{T_3-1} \leq (s_0 + 2)^{-\chi} \leq \dots$, where s_0 is some integer. It is easy to see that $T_{i+1} - T_i \approx (\chi/\kappa)(s_0 + i)^{\chi/\kappa-1} \asymp T_{i+1}^{\kappa}$.

Therefore, by the similar arguments as used in Lemma S2 and Theorem 1, let N be a fixed but sufficiently large value,

$$W_2(\pi_{T_{i+1}},\pi_*) \le (1-(q_N-\eta-\sqrt{(2)}\sigma)\epsilon_{T_{i+1}})^{T_{i+1}-T_i} W_2(\pi_{T_i},\pi_*) + C_1(\frac{\sqrt{p}}{m}) + C_2\left(\sqrt{p}+\left\{\frac{Np}{mn}\right\}^{1/2}\right)\sqrt{\epsilon_{T_i}}, \tag{S6}$$

for some constant C_1 and C_2, where $q_N-\eta-\sqrt{(2)}\sigma>0$. For a fixed value of N, $(1-(q_N-\eta-\sqrt{(2)}\sigma)\epsilon_{T_{i+1}})^{T_{i+1}-T_i} \le (1-(q_N-\eta-\sqrt{(2)}\sigma)(T_{i+1})^{-\kappa})^{T_{i+1}-T_i}$ converges to some positive constant less than 1 as $i\to\infty$, since $T_{i+1}-T_i \asymp T_{i+1}^{\kappa}$. If we further assume that $(1-(q_N-\eta-\sqrt{(2)}\sigma)(T_{i+1})^{-\kappa})^{T_{i+1}-T_i} \le \omega<1$ for all i, then (S6) is reduced to

$$W_2(\pi_{T_{i+1}},\pi_*) \le \omega W_2(\pi_{T_i},\pi_*) + C_1(\frac{\sqrt{p}}{m}) + C_2\left(\sqrt{p}+\left\{\frac{Np}{mn}\right\}^{1/2}\right)\sqrt{\epsilon_{T_i}}.$$

Iterating the above inequality, we have

$$\begin{aligned}
W_2(\pi_{T_K},\pi_*) &\le \omega^K W_2(\pi_{T_0},\pi_*) + \sum_{i=1}^{K-1}\omega^i C_1\frac{\sqrt{p}}{m} + C_2\left(\sqrt{p}+\left\{\frac{Np}{mn}\right\}^{1/2}\right)\sum_{i=1}^{K}\omega^{K-i}\sqrt{\epsilon_{T_i}} \\
&\le \omega^K W_2(\pi_{T_0},\pi_*) + \omega/(1-\omega)C_1\frac{\sqrt{p}}{m} + C_2\left(\sqrt{p}+\left\{\frac{Np}{mn}\right\}^{1/2}\right)\sum_{i=1}^{K}\omega^{K-i}(s_0+i-1)^{-\chi/2} \\
&\to \omega/(1-\omega)C_1\sqrt{p}/m \quad (\text{as } K\to\infty) \\
&\to 0, \quad (\text{as } N\to\infty, \sqrt{p}/m\to 0),
\end{aligned}$$

which concludes part (i).

(ii) The extended stochastic gradient Langevin dynamics algorithm can be expressed as

$$\begin{aligned}
\theta^{(t+1)} &= \theta^{(t)} + \frac{\epsilon_{t+1}}{2m}\sum_{k=1}^{m}\nabla_\theta \log\pi(\theta^{(t)}|\gamma_{S_k}^{(t)}, X_{n,N}^{(t)}) + \sqrt{(\epsilon_{t+1})}\eta_{t+1}, \\
&= \theta^{(t)} + \frac{\epsilon_{t+1}}{2}\nabla_\theta \log\pi(\theta^{(t)}|X_N^{(t)}) + \frac{\epsilon_{t+1}}{2}\xi(\theta^{(t)}) + \frac{\epsilon_{t+1}}{2}\zeta^{(t)} + \sqrt{(\epsilon_{t+1})}\eta_{t+1},
\end{aligned}$$

where $\xi(\theta^{(t)}) = E[\sum_{k=1}^m \nabla_\theta \log\pi(\theta^{(t)}|\gamma_{S_k}^{(t)}, X_{n,N}^{(t)})/m] - \nabla_\theta \log\pi(\theta^{(t)}|X_N)$ and $\zeta^{(t)} = \sum_{k=1}^m \nabla_\theta \log\pi(\theta^{(t)}|\gamma_{S_k}^{(t)}, X_{n,N}^{(t)})/m - E[\sum_{k=1}^m \nabla_\theta \log\pi(\theta^{(t)}|\gamma_{S_k}^{(t)}, X_{n,N}^{(t)})/m]$ denote the bias term and noise term, respectively. It has been shown in the proof of Theorem 1 that

$$\|\xi(\theta)\|^2 = o[\frac{N^2(\|\theta\|^2+p)}{p}], \quad E(\|\zeta^{(t)}\|^2|\theta^{(t)}) = O(\frac{N^2(\|\theta^{(t)}\|^2+p)}{mn}).$$

With the same notation as in Lemma S3, $F(\theta) = \nabla\log\pi(\theta|X_N) + \xi(\theta)$. Because $-\log\pi(\theta|X_N)$ is Q_N-gradient Lipschitz continuous and q_N-strongly convex with $q_N \asymp Q_N \asymp N$, we have

$$\langle F(\theta),\theta\rangle \le -q_N\|\theta\|^2 + \|\theta\|\|\nabla_{\theta=0}\log\pi(\theta|X_N)\| + o(\frac{N}{\sqrt{p}}(\|\theta\|^2+\|\theta\|)),$$

which implies condition (S2). It is also easy to verify conditions (S3)-(S5) by choosing $k=1$ and $V(\theta) = C(\|\theta\|^2+1)$ for some sufficiently large constant C. Also, the choice $\epsilon_t \propto t^{-\kappa}$ with $\kappa\in(0,1)$ satisfies the conditions of Lemma S3, and any Lipschitz continuous ρ satisfies that $|\rho|/V$ is bounded. Putting all these together it implies, with given X_N, N, p and m, that the diffusion process $d\theta^{(t)} = (1/2)F(\theta^{(t)})dt + dW^{(t)}$ has a unique invariant distribution π_N, and

$$\lim\frac{1}{T}\sum_{t=1}^{T}\rho(\theta^{(t)}) = E_{\pi_N}\rho(\theta), \text{ a.s.} \tag{S7}$$

As implied by Part (i) of Theorem 1, with proper growth rates of m and n, π_N converges to π_* in W_2-distance. Hence,

$$E_{\pi_N}\rho(\theta) \to E_{\pi_*}\rho(\theta). \tag{S8}$$

That is, the unweighted estimator given in part (ii) of Theorem 1 is still valid if the learning rate $\epsilon_t = O(1/t^\kappa)$ for some $0 < \kappa < 1$.

(iii) For the proof of part (iii) of Theorem 1 in the main paper, equation (8) still holds by (S7) and (S8), and the other parts of the proof hold independent of the setting of the learning rate. Therefore, the unweighted estimator given in part (iii) of Theorem 1 is still valid if we set the learning rate $\epsilon_t = O(1/t^\kappa)$ for some $0 < \kappa < 1$. This concludes the proof of Theorem 2. □

3. The Reversible Jump Metropolis-Hastings Algorithm Used in Algorithm 1

For Algorithm 1, we employ the reversible jump MH algorithm (Green, 1995) to draw the models $\gamma^{(t)}_{S_i,n}$, $i = 1, 2, \ldots, m$, at each iteration. The reversible jump MH algorithm consists of three types of moves, namely, birth, death and exchange. Let $\{w_i\}_{i=0}^p$ denote the weights assigned to intercept term and variable z_i for $i = 1, 2, \ldots, p$. For linear models, we assign $w_0 = \exp(-1)$, $w_i = \exp(|\rho(y, z_i)| - 1)$ for $i = 1, \ldots, p$, where $\rho(y, z_i)$ denotes the correlation coefficient between y and z_i; for generalized linear models, let d_i denote the deviance of the model S_i, which consists of variable z_i and the intercept term only, and we assign $w_0 = \exp(-\epsilon)$, $w_i = \exp\{-(d_i - \min_j d_j)/(5\sigma_d) - \epsilon\}$ for all $i = 1, \ldots, p$, where $\epsilon = 0.1$ and σ_d denotes the standard deviation of $d_1, d_2, \ldots, d_p$. Let $S^{(t)}$ denote the set of variables included in the current model, and let $S_c^{(t)}$ denote its complementary set. The birth move is to randomly select a variable from $S_c^{(t)}$ (with a probability proportional to its weight) and add it to $S^{(t)}$. The death move is to randomly select a variable from $S^{(t)}$ (with a probability proportional to one minus its weight w_i) and remove it. The exchange move is to randomly select a variable from $S^{(t)}$ (with a probability proportional to one minus its weight) and replace it by another variable randomly selected from $S_c^{(t)}$ (with a probability proportional to its weight). The exchange move keeps the size of $S^{(t)}$ unchanged. In addition, when a variable is selected to be added to or removed from $S^{(t)}$, the sign of the corresponding component of θ will be randomized, i.e., altered with probability 0.5. This accommodates with the prior setting of $\theta_{[-S]}$ in equation (7) of the main paper. In simulations, we set the proposal probability $P(\text{birth}|S^{(t)}) = P(\text{death}|S^{(t)}) = P(\text{exchange}|S^{(t)}) = 1/3$ if $0 < |S^{(t)}| < q$, $P(\text{birth}|S^{(t)}) = 1$ if $|S^{(t)}| = 0$, and $P(\text{death}|S^{(t)}) = 1$ if $|S^{(t)}| = q$. Note that in the reversible jump moves, no regression coefficients were updated. This is different from traditional reversible jump MH algorithms, where the regression coefficients of $\theta_{S^{(t)}}$ might need to be updated accordingly.

4. Numerical Examples

4.1. *A Linear Regression Example*

This section provides more details for the illustrative example reported in the main paper. The true model follows

$$y = z_1 + z_2 + z_3 + z_4 + z_5 - z_6 - z_7 - z_8 + 0 \cdot z_9 + \cdots + 0 \cdot z_p + \varepsilon, \tag{S9}$$

where $\varepsilon \sim N(0, I_N)$, and I_N denotes an $N \times N$ identity matrix. The explanatory variables $z_1, \ldots, z_p$ were generated from a multivariate Gaussian distribution with a mutual correlation coefficient of 0.5. We generated 10 independent datasets from the model (S9) with $N = 50,000$ and $p = 2000$. A hierarchical prior was assumed for the model and parameters. For each model S, we set

$$\pi(\gamma_S) \propto \lambda^{|S|}(1 - \lambda)^{p+1-|S|} I(|S| \le q), \tag{S10}$$

where $|S|$ denotes the number of explanatory variables included in the model, q is set to 50, and λ is set to $1/(p + 1)^{1.1}$ by including the intercept term. Here the factor $p + 1$ means that the intercept term has been

included in the model as a special explanatory variable. Conditioned on γ_S, we set

$$\theta_{[S]} \sim N(0, \sigma_1^2 I_{|S|}), \quad \theta_{[-S]} \sim N(0, \sigma_0^2 I_{p+1-|S|}), \tag{S11}$$

where $\theta_{[S]}$ and $\theta_{[-S]}$ are the auxiliary variables corresponding to the variables included in and excluded by the model S, respectively; σ_1^2 is set to a constant of 25; and σ_0^2 is set to a constant of 0.025.

Algorithm 1 was run for 5000 iterations with the subsample size $n = 200$, the number of models $m = 10$ drawn at each iteration, and the learning rate $\epsilon_t \equiv 10^{-6}$, where the first 2000 iterations were discarded for the burn-in process. In each iteration, the subset models were drawn using the reversible jump Metropolis-Hastings algorithm described in Section 3 of this material. Regarding the choice of the learning rate, we note that Theorem 1 of the main paper suggests that it should be set in the order $o(1/N)$. In all our examples, we did not tune the learning rate much, just setting it around $1/N$ while avoiding possible blowups of the simulation. Refer to Nemeth & Fearnhead (2019) for discussions on this issue for the general stochastic gradient Langevin dynamics algorithm.

The variables were selected according to the median posterior probability rule (Barbieri & Berger, 2004), which selects only the variables with the marginal inclusion probability greater than 0.5. Table 1 of the main paper summarizes the performance of the algorithm, where the false selection rate (FSR), negative selection rate (NSR), MSE_0 and MSE_1 are reported in averages over 10 independent dataset. To be more precise, we define

$$\text{FSR} = \frac{1}{10}\sum_{i=1}^{10} |\hat{S}_i \setminus S_*|/|\hat{S}_i|, \quad \text{NSR} = \frac{1}{10}\sum_{i=1}^{10} |S_* \setminus \hat{S}_i|/|S_*|,$$

where S_* denotes the set of true variables, $\hat{S}_i$ denotes the set of selected variables for dataset i, and $|\cdot|$ denotes the cardinality of a set. The accuracy of the parameter estimates is measured by

$$\text{MSE}_0 = \frac{1}{10}\sum_{i=1}^{10} \text{MSE}(\hat{\beta}_i^{(0)}), \quad \text{MSE}_1 = \frac{1}{10}\sum_{i=1}^{10} \text{MSE}(\hat{\beta}_i^{(1)}),$$

where i indexes datasets, and $\hat{\beta}_i^{(0)}$ and $\hat{\beta}_i^{(1)}$ denote the estimates of the coefficients of the false and true variables, respectively. The $\hat{\beta}_i$ is obtained by averaging over a set of thinned (by a factor of 10) posterior samples produced by Algorithm 1. The extremely small values of MSE_0 and MSE_1 and the zero values of FSR and NSR indicate that all the simulations have converged to the true model.

For comparison, some existing algorithms, including the reversible jump Metropolis-Hastings algorithm (Green, 1995), split-and-merge (Song & Liang, 2015), and Bayesian Lasso (Park & Casella, 2008), were applied to this example. For this example, the reversible jump Metropolis-Hastings algorithm was also run for 5000 iterations. Similar to the implementation of extended stochastic gradient Langevin dynamic algorithm, the first 2000 iterations were discarded for the burn-in process, the variables were selected according to the median posterior probability rule (Barbieri & Berger, 2004), and the regression coefficients were estimated based on the samples collected from the last 3000 iterations (thinned by a factor of 10).

For the split-and-merge algorithm, we first split the data into 5 equal subsets along the dimension of predictors; that is, each subset consists of 400 predictors and 50,000 observations. For each subset, we imposed a hierarchical prior on the model and parameters, where each predictor has a prior probability of $(1/401)^{1.1}$ being selected, and if selected, its coefficient follows a Gaussian distribution $N(0, 5^2)$. For each subset, the reversible jump Metropolis-Hastings algorithm was run for 2500 iterations, where the first 1000 iterations were discarded for the burn-in process, and the predictors with the marginal posterior inclusion probability greater than 0.3 were selected. The predictors selected from each subset were then merged into one dataset. For the merged dataset, we imposed the same hierarchical prior on the model and parameters, and then ran the reversible jump Metropolis-Hastings algorithm for 2500 iterations with the first 1000 iterations discarded for the burn-in process. The final model was selected according to the median posterior probability rule, and the parameters were estimated based on the samples collected from the last 1500 iterations.

For Bayesian Lasso, we adopt the Laplace prior $\pi(\beta) \propto \exp(-\lambda \sum |\beta_j|)$ with $\lambda = \{2N \log(p)\}^{1/2}$. The Gibbs sampler was used to simulate posterior samples, as described in Park & Casella (2008), for 200 iterations. The predictors with the absolute value of the coefficient estimate greater than 0.1 were selected as the "true" predictors, where the coefficient was estimated by averaging its posterior sample over the iterations.

Among the four algorithms, the extended stochastic gradient Langevin dynamics algorithm cost the least CPU time and perfectly selected all true variables. The reversible jump Metropolis-Hastings algorithm took on average about 3 CPU hours to complete 5000 iterations. However, the resulting Markov chain was not yet mixed well and yielded high FSR, NSR and MSE_1 values. It is worth to note that the CPU time cost by the reversible-jump Metropolis-Hastings algorithm has a high variety among the ten runs, ranging from 42 to 305 minutes. This is due to the local trapping issue suffered by the reversible-jump Metropolis-Hastings algorithm; the CPU time can dramatically increase if the Markov chain is frequently trapped at large models. By working on lower dimensional subset data, the split-and-merge algorithm significantly improves the performance of the reversible jump Metropolis-Hastings algorithm, but is still inferior to the extended stochastic gradient Langevin dynamics algorithm in both variable selection accuracy and computational efficiency. We note that the split-and-merge algorithm can be further accelerated by parallel computing, although not done here. The Bayesian Lasso was also capable to perfectly recover the true model. For this algorithm, the major CPU cost is for computing at each iteration the inverse of an $p \times p$-matrix, which, unfortunately, is not scalable with respect to N. Although the algorithm was only run for 200 iterations, it still cost more CPU time than the extended stochastic gradient Langevin dynamics algorithm. It is worth noting that other than linear regression, the Bayesian Lasso algorithm will not be so efficient as the conjugate Gibbs updates will not be available anymore.

4.2. *A Logistic Regression Example*

Other than the linear regression example reported in the main paper, we also tested Algorithm 1 on large-scale logistic regression. We simulated 10 large datasets from the model

$$\text{logit}P(Y_i = 1) = \sum_{i=1}^{p} \beta_j z_{ij}, \quad i = 1, 2, \ldots, N, \tag{S12}$$

where $N = 50,000$, $p = 2000$, $\beta_1 = \cdots = \beta_5 = 1$, $\beta_6 = \beta_7 = \beta_8 = -1$, and $\beta_9 = \cdots = \beta_p = 0$. The explanatory variables z_i's, where $z_i = (z_{i1}, \ldots, z_{ip})^T$, were generated such that they have a mutual correlation coefficient of 0.5. The response variables were generated such that half of them have a value of 1 and the other half have a value of 0.

To conduct Bayesian analysis for the datasets, we adopted the same prior setting as in Liang et al. (2013). For each model S, we set

$$\pi(\gamma_S) \propto \lambda^{|S|}(1-\lambda)^{p+1-|S|} I(|S| \leq q), \tag{S13}$$

where $|S|$ denotes the number of explanatory variables included in the model, q is set to 500, λ is set to $1/\{1 + (p+1)^{\zeta}\sqrt{(2\pi)}\}$, and $\zeta = 0.5$. Similar to the linear regression case, the factor $p + 1$ means that the intercept term has been included in the model as a special explanatory variable. Conditioned on γ_S, we set

$$\theta_{[S]} \sim N(0, \sigma_S^2 I_{|S|}), \quad \theta_{[-S]} \sim N(0, \sigma_0^2 I_{p+1-|S|}), \tag{S14}$$

where $\theta_{[S]}$ and $\theta_{[-S]}$ denote the auxiliary variables corresponding to the variables included in and excluded by the model S, respectively; σ_0^2 is set to a constant of 0.025; and $\sigma_S^2 = \exp\{C_0/|S|\}/(2\pi)$. For this example, we set $C_0 = 10$. As shown in Liang et al. (2013), such a prior setting ensures the posterior consistency and variable selection consistency for high-dimensional generalized linear models, even when p is much larger than N.

For each dataset, Algorithm 1 was run for 5000 iterations with the subsample size $n = 300$, the learning rate $\epsilon_t \equiv 10^{-5}$, and the number of models $m = 10$ simulated at each iteration, where the first 2000 iterations were discarded for the burn-in process and the samples generated from the remaining iterations

Table S1. *Bayesian variable selection for logistic regression with the extended stochastic gradient Langevin Dynamics (eSGLD) algorithm, where FSR, NSR, MSE_1 and MSE_0 are reported in averages over 10 datasets with standard deviations given in the parentheses, and the CPU time (in minutes) was recorded for one dataset on a Linux machine with Intel(R) Core(TM) i7-7700 CPU@3.60GHz.*

Model	Algorithm	FSR	NSR	MSE_1	MSE_0	CPU(minutes)
Logistic	eSGLD	0(0)	0(0)	$2.37 \times 10^{-2}(2.88 \times 10^{-3})$	$2.70 \times 10^{-4}(1.40 \times 10^{-4})$	2.9

were used for inference. At each iteration, the models were simulated using the same reversible jump Metropolis-Hastings algorithm as described in Section 3 of this material.

The results were summarized in Table S1. For each dataset of this example, the extended stochastic gradient Langevin dynamics algorithm took only 2.9 CPU minutes on a personal computer of 3.6GHz! The numerical results indicate again that the extended stochastic gradient Langevin dynamic algorithm has much alleviated the pain of Bayesian methods in big data analysis.

4.3. *A Pascal Challenge Dataset*

We considered the dataset *epsilon* with logistic regression. The dataset is available at `https://www.csie.ntu.edu.tw/~cjlin/libsvmtools/datasets`, which has been used for Pascal large scale learning challenge in 2008. The raw dataset consists of 500,000 observations and 2000 features, which was split into two parts: 400,000 observations for training and 100,000 observations for testing. The training part is feature-wisely normalized to mean zero and variance one and then observation-wisely scaled to unit length. Using the scaling factors of the training part, the testing part is processed in a similar way.

For this example, we applied the same prior settings as used in Section 4.2 of this material except that C_0 was set to 2.5. A small value of C_0 enhances the selection of a larger set of variables, while keeping the regression coefficients of the selected variables relatively small. Our numerical experience shows that this often improves the prediction performance for real data problems, as for which the true model might deviate from the logistic regression and thus need to be approximated with a large number of variables (but the effect of each variable is small). With this prior setting, Algorithm 1 was run for the dataset for 10 times independently. Each run consisted of 1000 iterations, where the first 500 iterations were discarded for the burn-in process, and the samples drawn from the remaining iterations were used for inference. In simulations, we set the subsample size $n = 2000$, the learning rate $\epsilon_t \equiv 5 \times 10^{-4}$, and the number of simulated models $m = 10$ per iteration.

The numerical results were summarized in Table S2, where the variables were selected according to the median posterior probability rule (Barbieri & Berger, 2004). To measure the quality of the selected models, we evaluated the prediction accuracy of the selected models on the test dataset. For comparison, we applied the Lasso algorithm (Tibshirani, 1996), which was implemented using the package *glmnet*, to this example. For Lasso, we reported only the results with one value of the regularization parameter λ at which it selected about the same size of models as eSGLD. A t-test for the eSGLD prediction error for the hypothesis $H_0 : \nu = 0.1644$ versus $H_1 : \nu \neq 0.1644$ shows a p-value of 6.014×10^{-7}, where ν denotes the prediction error. Therefore, the models selected by eSGLD have significantly lower predication error than that selected by Lasso. Later, we have tried other values of C_0 such as 1, 5 and 10. Under each of them, the models selected by eSGLD have significantly lower prediction error than the same size models selected by Lasso.

For a thorough comparison, we have also applied the SCAD (Fan & Li, 2001) and MCP (Zhang, 2010) algorithms to this example, which both were implemented in the R-Package *SIS* (Saldana & Feng, 2018). Unfortunately, they did not produce any results after running 24 CPU hours. This comparison also shows that Lasso penalty is computationally more attractive than the SCAD and MCP penalties, although they share similar theoretical properties.

Table S2. *Numerical results for the dataset epsilon, where the results of the extended stochastic gradient Langevin dynamics (eSGLD) algorithm were averaged over 10 independent runs with the standard deviation reported in the parentheses, and the CPU time was recorded for a run on a Linux machine with Intel(R) Core(TM) i7-7700 CPU@3.60GHz.*

Algorithm	Setting	Model size	Prediction Error(ν)	CPU (minutes)
eSGLD	$C_0 = 2.5$	62.7(2.6)	0.1422 (1.80×10^{-3})	17.4
Lasso	$\lambda = 0.0239$	63	0.1644	21.2
MCP/SCAD	—	—	—	> 1440

This example shows again that the extended stochastic gradient Langevin dynamics algorithm has much alleviated the pain of Bayesian computing for big data problems. For this example, it has even about the same CPU cost as the Lasso algorithm.

4.4. *MovieLens Data*

Other than variable selection, we applied the extended stochastic gradient Langevin dynamics algorithm to a large-scale missing data problem, estimating the mixed effect model for the MovieLens 10M Dataset downloaded at `https://grouplens.org/datasets/movielens/`. The dataset contains $N = 10,000,054$ ratings of 10,681 movies by 71,567 users. The ratings vary from 0.5 to 5 in an increment of 0.5. The whole dataset contains information of the ratings, the time of the ratings, and the genre of the movies (with 19 possible categories).

This dataset has been modeled by mixed effect models, see e.g. Van Dyk (2000) and Srivastava et al. (2018). Let N be the total number of users, and let s_i denote the number of ratings of user i. Following Srivastava et al. (2018), we set

$$y_i = W_i\beta + Z_i b_i + e_i, \quad b_i \sim N_q(0, \Sigma),\ \Sigma = \sigma^2 D,\ e_i \sim N_{s_i}(0, \sigma^2 I_{s_i}),$$

where $y_i \in \mathbb{R}^{s_i}$, $W_i \in \mathbb{R}^{s_i \times p}$, $\beta \in \mathbb{R}^p$, $Z_i \in \mathbb{R}^{s_i \times q}$, $b_i \in \mathbb{R}^q$ for $i = 1, 2, \ldots, N$. Let $\theta = (\beta, D, \sigma^2)$ denote the parameter vector of the model, and let r_{ij} be the rating of user i for movie j. The response is defined as $y_i = (r_{ij_1}, r_{ij_2}, \ldots, r_{ij_{s_i}})^T$. Here we assume that $(y_i, W_i, Z_i),\ i = 1, 2, \ldots, N$ are independent and identically distributed random samples with varying dimensions, where W_i denotes the collection of the intercept, genre predictor, popularity predictor and previous predictor (defined below), and Z_i is the same with W_i.

- Genera predictor, which is a categorical variable for four categories, namely 'Action', 'Children', 'Comedy', and 'Drama', grouped from 19 movie genres. The category 'Action' consists of the action, adventure, fantasy, horror, sci-fi, and thriller genres. The category 'Children' consists of the animation and children geres. The category 'Comedy' consists of only the comedy genre. The category 'Drama' consists of the crime, documentary, drama, film-noir, musical, mystery, romance, war, and western genres. We use effect coding to represent each category, i.e., using $(1, 0, 0), (0, 1, 0), (0, 0, 1), (-1, -1, -1)$ to represent Children, Comedy, Drama and Action, respectively. The genre predictor of movie j is defined as the average of all categories that movie j belongs to. For example, if movie j belongs to Fantasy, Thriller and Musical genres, then its genre predictor is $(1/3)\left((1, 0, 0) + (1, 0, 0) + (0, 1, 0)\right) = (2/3, 1/3, 0)$.
- Popularity predictor, which, for the rating r_{ij}, is defined as $\text{logit}\{(l_j + 0.5)/(L_j + 1.0)\}$, where L_j is the number of recent ratings of movie j, and l_j is the number of recent ratings of movie j with score greater than 3. Here "recent" means 30 or fewer most recent ratings.
- Previous predictor, which, for the rating r_{ij}, is defined as 1 if user i rated previous movie with score greater than 3 and 0 otherwise.

As in Srivastava et al. (2018), we treat the random effect b_i as missing data and set the priors

$$\beta|\sigma^2 \sim N_p(0,\ \sigma^2 I_p),\ \sigma^2 \sim \frac{1}{\chi^2_2},\ D \sim \text{inverse-Wishart}(p+2,\ I_q).$$

In each iteration of Algorithm S1, we first randomly select a mini-batch of users with the subsample size $n = 500$. For the selected users, we sample the random effect b_i based on $p(b_i|y_i, \beta, \sigma^2, D)$ (see Equation 3.6 in Van Dyk (2000) for its analytic form) and then update $\theta = (\beta, D, \sigma^2)$ according to the rule of Langevin Dynamics. We use the linear regression result for $Y = X\beta + e$ to initialize β and σ^2 and use a diagonal matrix with all diagonal elements equal to 0.2 to initialize D. Algorithm S1 was run for 4000 iterations. The learning rate was set to $0.002/N$ for this example.

To compare with other competitive Bayesian sampling algorithm, we also perform the data augmentation Gibbs sampler, which iteratively samples θ and b from the full data conditional distribution $\pi(b|\theta, X_N)$ and $\pi(\theta|b, X_N)$, respectively where X_N denotes the whole data. The algorithm was run for 500 iterations.

The convergence performance of the extended stochastic gradient Langevin dynamics algorithm and Gibbs sampler algorithm can be assessed by the kernel Stein discrepancy (KSD) (Gorham & Mackey, 2015, 2017). For a set of samples $\{\theta^{(t)}\}_{t=1}^T$ generated by a Markov chain, the kernel Stein discrepancy between the empirical distribution of $\theta^{(t)}$, denoted by π_t, and the target posterior distribution $\pi_* = \pi(\theta|X_N)$ is defined as

$$\text{KSD}(\pi_t, \pi_*) = \sum_{j=1}^{p} \left\{ \sum_{t,t'=1}^{T} \frac{k_j^0(\theta^{(t)}, \theta^{(t')})}{T^2} \right\}^{1/2},$$

where the Stein kernel for $j \in \{1, 2, \ldots, p\}$ is given by $k_j^0(\theta, \theta') = \nabla_{\theta_j} U(\theta) \nabla_{\theta'_j} U(\theta') k(\theta, \theta') + \nabla_{\theta_j} U(\theta) \nabla_{\theta'_j} k(\theta, \theta') + \nabla_{\theta'_j} U(\theta') \nabla_{\theta_j} k(\theta, \theta') + \nabla_{\theta_j} \nabla_{\theta'_j} k(\theta, \theta')$, p is the dimension of θ, $U(\theta) = \log \pi(\theta|X_N)$, and $k(\theta, \theta') = (1 + ||\theta - \theta'||_2^2)^{-0.5}$ is the inverse multi-quadratic kernel. In practice, KSD can also be calculated on some components of θ for simplicity. In this case, p will be smaller than the dimension of θ.

Figure S1 compares the KSD paths of β produced by the extended stochastic gradient Langevin dynamics algorithm and the Gibbs sampler, where the x-axis shows the elapsed CPU time. Figure S1 indicates that to achieve the same level of KSD, the Gibbs sampler costs much longer, approximately 7 times, CPU time than the extended stochastic gradient Langevin dynamics algorithm. Figure S2 plots the sample trajectories of β_i's for both algorithms, with respect to the iteration number. It is easy to see that with same number of iterations, the extended stochastic gradient Langevin dynamics algorithm does converge slower than the Gibbs sampler, however, such disadvantage is compensated by its shorter CPU time cost per iteration.

REFERENCES

BARBIERI, M. & BERGER, J. (2004). Optimal predictive model selection. *Annals of Statistics* **32**, 870–897.

BHATIA, K., MA, Y.-A., DRAGAN, A. D., BARTLETT, P. L. & JORDAN, M. I. (2019). Bayesian robustness: A nonasymptotic viewpoint. *arXiv preprint arXiv:1907.11826* .

DALALYAN, A. S. & KARAGULYAN, A. G. (2017). User-friendly guarantees for the Langevin Monte Carlo with inaccurate gradient. *CoRR* **abs/1710.00095**.

DOUC, R., FORT, G. & GUILLIN, A. (2009). Subgeometric rates of convergence of f-ergodic strong Markov processes. *Stochastic processes and their applications* **119**, 897–923.

FAN, J. & LI, R. (2001). Variable selection via nonconcave penalized likelihood and its oracle properties. *Journal of the American Statistical Association* **96**, 1348–1360.

GORHAM, J. & MACKEY, L. (2015). Measuring sample quality with Stein's method. In *Advances in Neural Information Processing Systems*.

GORHAM, J. & MACKEY, L. (2017). Measuring sample quality with kernels. In *Proceedings of the 34th International Conference on Machine Learning-Volume 70*. JMLR. org.

GREEN, P. (1995). Reversible jump Markov chain Monte Carlo computation and Bayesian model determination. *Biometrika* **82**, 711–732.

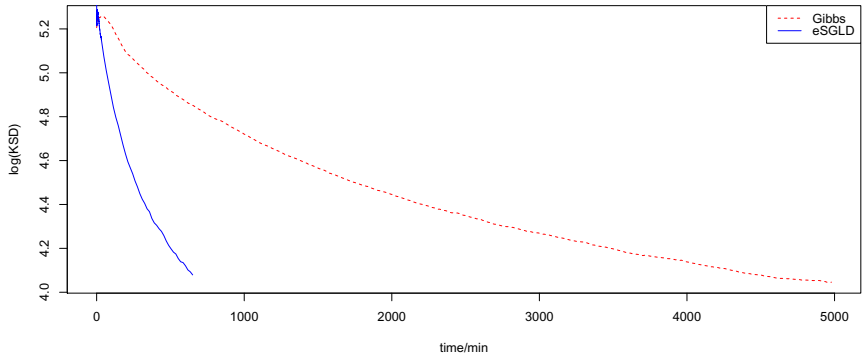

Fig. S1. KSD ($\log_{10}$ scale) paths of β against CPU time (minutes), where the solid line is for the extended stochastic gradient Langevin dynamics (eSGLD) algorithm and the dotted line is for the Gibbs Sampler.

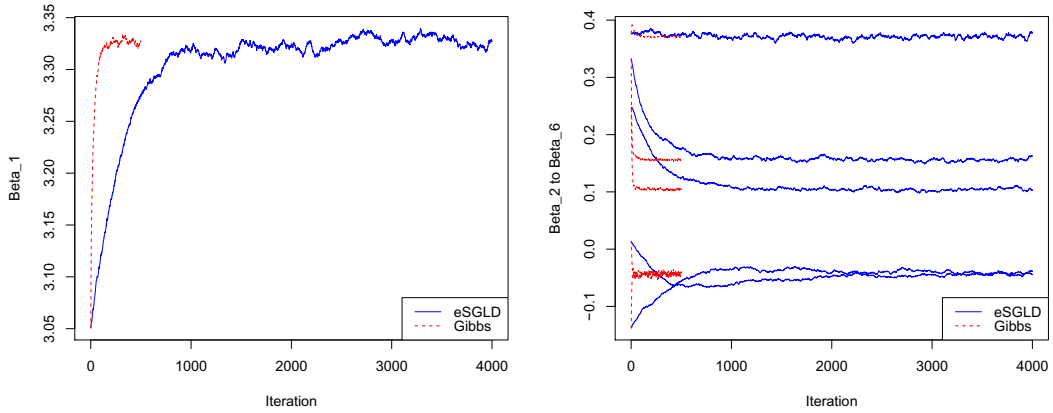

Fig. S2. Sample paths of different components of $\beta = (\beta_1, \ldots, \beta_6)$, where the left panel is for β_1, and the right panel is for components β_2 to β_6.

LIANG, F., SONG, Q. & YU, K. (2013). Bayesian subset modeling for high dimensional generalized linear models. *Journal of the American Statistical Association* **108**, 589–606.

NEMETH, C. & FEARNHEAD, P. (2019). Stochastic gradient Markov chain Monte Carlo. *arXiv:1907.06986* .

PARK, T. & CASELLA, G. (2008). The Bayesian Lasso. *Journal of the American Statistical Association* **103**, 681–686.

SALDANA, D. F. & FENG, Y. (2018). SIS: an R package for sure independence screening in ultrahigh dimensional statistical models. *Journal of Statistical Software* **83**, 1–25.

SONG, Q. & LIANG, F. (2015). A split-and-merge Bayesian variable selection approach for ultra-high dimensional regression. *Journal of the Royal Statistical Society, Series B* **77**, 947–972.

SRIVASTAVA, S., DEPALMA, G. & LIU, C. (2018). An asynchronous distributed expectation maximization algorithm for massive data: The dem algorithm. *arXiv preprint arXiv:1806.07533* .

TEH, Y. W., THIERY, A. H. & VOLLMER, S. J. (2016). Consistency and fluctuations for stochastic gradient Langevin dynamics. *The Journal of Machine Learning Research* **17**, 193–225.

TIBSHIRANI, R. (1996). Regression shrinkage and selection via the lasso. *Journal of the Royal Statistical Society, Series B* **58**, 267–288.

VAN DYK, D. A. (2000). Fitting mixed-effects models using efficient EM-type algorithms. *Journal of Computational and Graphical Statistics* **9**, 78–98.

ZHANG, C.-H. (2010). Nearly unbiased variable selection under minimax concave penalty. *Annals of Statistics* **38**, 894–942.