Bayesian Learning via Stochastic Gradient Langevin Dynamics

Narjisse Lasri
IMT Nord Europe
narjisse.lasri@etu.imt-lille-douai.fr

Amal Chaoui
Centrale Lille
amal.chaoui@centrale.centralelille.fr

Abstract

Stochastic gradient optimization is widely used in most learning tasks since it provides faster convergence. Stochastic Gradient Langevin Dynamics (SGLD) was introduced to leverage the power of SGD in the Bayesian framework. The main idea is to add a noise term that scales as the step size in the gradient update expression. Although the algorithm achieves good performance for different learning tasks, we show that it results in a poor mixing behavior of the produced chain. To address this issue, we suggest a variant algorithm based on Hamiltonian Monte Carlo that decorrelates the moves and improves the chain mixing.

1 Introduction

In the last few years, methods with stochastic optimization-based approaches have been very successful and widely used in many machine learning tasks. Bayesian methods, on the other hand, are one of the categories of methods dropped by the recent advances in large-scale machine learning. This can be explained by the fact that each iteration of the algorithms requires computations on the whole data set. Nevertheless, they are attractive in their ability to capture uncertainty and avoid overfitting simultaneously. The method presented in "Bayesian Learning via Stochastic Gradient Langevin Dynamics" (SGLD) [Welling and Teh(2011)] combines Robbins-Monro stochastic optimization algorithms with Langevin dynamics. This association lessens the requirement of computing updates over all the data used in most MCMC methods while avoiding overfitting.

We will first introduce the theoretical setting that motivates SGLD method in section 2. We then present an application of SGLD logistic regression to a real dataset in section 3. In section 4, we show the strengths and the weaknesses of the method and suggest a variant sampling method based on Hamiltonian Monte Carlo (HMC) in section 5. Finally, we end up with a conclusion about possible future improvements (section 6). Source code and data are available at this GitHub repository.

2 Stochastic Gradient Langevin Dynamics

2.1 Theoretical setting

Notations Let θ denote a parameter vector, with $p(\theta)$ a prior distribution, and $p(x \mid \theta)$ the probability of data item x given the model parameterized by θ. The posterior distribution of a set of N data items $X = \{x_i\}_{i=1}^{N}$ is: $p(\theta \mid X) \propto p(\theta) \prod_{i=1}^{N} p(x_i \mid \theta)$.

Stochastic Optimization The Stochastic Optimization is a class of methods used to obtain the maximization of the log-posterior. The general idea is that an approximative gradient is computed on a subset of n data items ($X_t = x_{t1}, ..., x_{tn}$) instead of the whole dataset. At each iteration, it is updated as follow :

$$\Delta\theta_t = \frac{\epsilon_t}{2}\left(\nabla \log p\left(\theta_t\right) + \frac{N}{n}\sum_{i=1}^{n} \nabla \log p\left(x_{ti} \mid \theta_t\right)\right)$$

where (ϵ_t) is a sequence of step sizes that satisfy the property: $\sum_{t\geq 1} \epsilon_t = \infty \quad \sum_{t\geq 1} \epsilon_t^2 < \infty$. The issue with this method is that they do not capture parameter uncertainty and can potentially overfit data. Bayesian approaches can overcome this problem.

Markov chain Monte Carlo through Langevin dynamics Langevin dynamics is a class of Markov chain Monte Carlo (MCMC) techniques. It takes gradient steps and injects Gaussian noise into the parameter updates.

$$\Delta\theta_t = \frac{\epsilon}{2}\left(\nabla \log p\left(\theta_t\right) + \sum_{i=1}^{N} \nabla \log p\left(x_i \mid \theta_t\right)\right) + \eta_t; \quad \eta_t \sim N(0, \epsilon)$$

However, the main issue is that each iteration requires computations over the whole dataset. The consequence is a very high computational costs for large datasets.

Stochastic Gradient Langevin Dynamics The article proposes to combine the stochastic gradient algorithms and Langevin dynamics to benefit the advantages of both methods. The proposed update is:

$$\Delta\theta_t = \frac{\epsilon_t}{2}\left(\nabla \log p\left(\theta_t\right) + \frac{N}{n}\sum_{i=1}^{n} \nabla \log p\left(x_{ti} \mid \theta_t\right)\right) + \eta_t; \quad \eta_t \sim N\left(0, \epsilon_t\right)$$

In the initial phase the algorithm imitates an efficient stochastic gradient ascent algorithm. In the second phase the injected noise dominates and the algorithm approximates a Langevin dynamics algorithm. This allows efficient use of large datasets while allowing for parameter uncertainty to be captured in a Bayesian manner.

2.2 Posterior sampling

Transition into Langevin dynamics phase The transition from the burn-in phase of optimization into the posterior sampling phase happens when the variance of the injected noise $\eta_t \sim \mathcal{N}(0, \epsilon_t M)$ dominates the variance of the stochastic gradient step. M is a preconditioning matrix introduced to better adapt the step sizes to the local structure of the posterior. This defines a condition on the sample threshold α for the transition to happen, which can be expressed $\frac{\epsilon_t N^2}{4n}\lambda_{max}(M^{1/2}V_s M^{1/2}) = \alpha \ll 1$, with $\lambda_{max}(A)$ being the largest eigenvalue of A.

Posterior expectation estimation Once the posterior samples $\{\theta_t\}_{1\leq t\leq T}$ collected, one can estimate the posterior expectation $\mathbb{E}[f(\theta)] \approx \frac{1}{T}\sum_{t=1}^{T} f(\theta_t)$. However, since the step size ϵ_t decreases, the mixing rate of the chain decreases too. The authors, therefore, proposed to weight the samples using the step sizes: $\mathbb{E}[f(\theta)] \approx \frac{\sum_{t=1}^{T}\epsilon_t f(\theta_t)}{\sum_{t=1}^{T}\epsilon_t}$, to prevent over-emphasizing the tail end of the distribution.

3 Application and results: logistic regression

We applied SGLD to a logistic regression classification task on the bank marketing dataset[1]. It consists of data 16 features collected from 11162 customers by a Portuguese banking institution. The

[1] `https://www.kaggle.com/datasets/janiobachmann/bank-marketing-dataset`

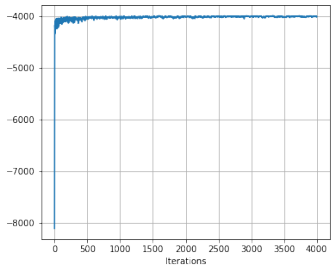


Figure 1: SGLD log-posterior along iterations.

	SGLD	NUTS
Train acc.	0.801 ± 0.001	0.801 ± 0.001
Test acc.	0.788 ± 0.003	0.790 ± 0.002

Table 1: Training and test accuracies obtained by SGLD and SGDClassifier.

classification goal is to predict whether the client will subscribe to a term deposit after a phone calls marketing campaign. For this, we consider a Bayesian logistic regression model with a Gaussian prior over the regression parameter θ: $y_i \underset{iid}{\sim} \mathcal{B}(\sigma(y_i x_i^T \theta)), \quad \forall i \in \{1, \dots, N\}$, where $\theta \sim \mathcal{N}(0, I)$. Under this setting, the Langevin dynamics gradient updates over a batch of size n can be formulated as follows

$$\theta_t + \frac{\epsilon_t}{2}\left[-\frac{1}{\sigma^2}\theta + \frac{N}{n}\sum_{i=1}^{n}(1 - \sigma(y_i x_i^T \theta))y_i x_i\right] + \eta_t. \tag{1}$$

For the experiment, we set $a = 0.01, b = 500, \gamma = 0.6$ and we run the SGLD sampler for 4000 iterations on a batch size of 140 samples at each iteration. Figure 1 illustrates the evolution of the log-posterior obtained by SGLD along iterations. We can see that the curve converges faster to the optimum value.

We compare the performance of SGLD on this classification task to that of NUTS sampler from PyMC. Table 1 shows the mean and standard deviation of the accuracies computed using the posterior samples produced by these sampling methods on the training and test sets (8929 and 2233 samples resp.). We obtain close performance to NUTS using SGLD while taking advantage of less computations, hence lower convergence time.

4 Strengths and weaknesses

Strenghts The combination of SGD and BML allows SGLD to benefit from two main advantages :

- Using the Bayesian framework to introduce uncertainty quantification while using the prior as a regularizer and leveraging the power of the mini-batch optimization for fast optimization.
- Performing optimization and posterior sampling in the same algorithm.

Weakenesses Although SGLD seems to reconcile the Bayesian framework and the frequentist framework, there are some limitations to the use of this MCMC algorithm:

- The need to tune the step size parameters a, b, γ to obtain fast convergence while having control over the chaotic behavior due to the stochastic updates.
- Poor mixing during the sampling phase (c.f. figure 3a) due to the decrease in the step size. This means that the chain can easily get trapped in one single mode of the posterior, making the algorithm unsuitable for sampling from complex posterior distributions with multiple modes.

5 Contribution: Stochastic Gradient Hamiltonian Dynamics

In this section, we attempt to address the poor mixing issue of the chain obtained by the SGLD sampling method. For this purpose, we take advantage of the Hamiltonian Monte Carlo sampling

to untangle the correlation between the posterior samples by introducing an additional variable p (momentum). The novelty compared to HMC would be to compute the updates on a mini-batch of training samples only at each iteration. We call this MCMC variant Stochastic Gradient Hamiltonian Dynamics (SGHD). We recall the expression of the Hamiltonian

$$H(q,p) = -\log \pi(q) + p^T M_H p$$

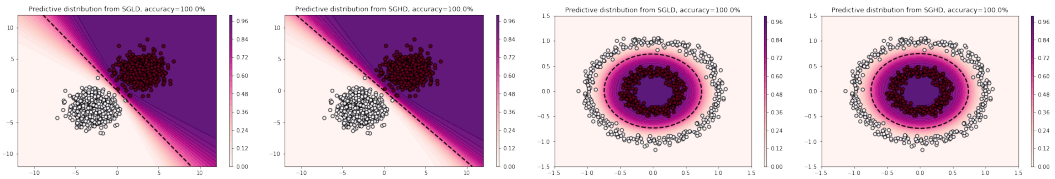
with π being the posterior distribution we want to sample from, and M_H a mass matrix that encodes relevant posterior information. It usually estimated from the covariance of the preliminary posterior samples [Neal(2012)].

We use the leapfrog integrand to approximate the Hamiltonian flow and update p and q using the following equations:

$$p_{1/2} = p + \frac{h}{2}\nabla \log \pi(q), \quad q' = q + hMp_{1/2}, \quad p' = p_{1/2} + \frac{h}{2}\nabla \log \pi(q').$$

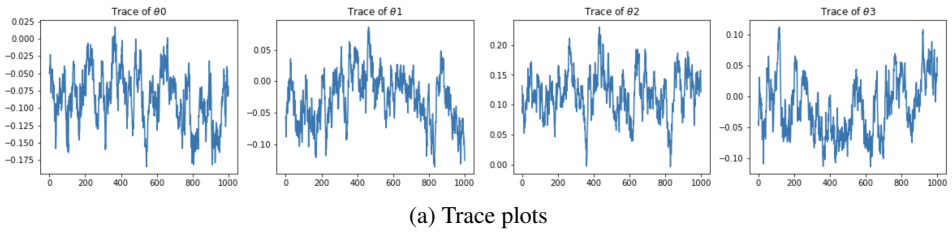
As a preliminary step of evaluating the SGHD sampler, we assess its performance of 2 classification tasks: linearly separable and non-linearly separable classes with 1500 and 700 samples respectively. We set the gradient step size parameters to $a = 0.1, b = 40, \gamma = 0.7$. We take a batch size of 750 samples (resp. 50 samples) for the first task (resp. second task). Figure 2 illustrates the decision boundaries obtained for each classification task by SGHD and SGLD for the same parameters setting, which look very similar.

Afterwards, we assess the performance of SGHD against SGLD on the bank marketing dataset. We take $a = 0.01, b = 100, \gamma = 0.6$, and a batch size of 140. We set the size and the number of the leapfrog steps to $h = 0.5$ and $n_h = 80$ resp. We visualize the trace plots as well as the correlation plots of the 1st 4 parameters in the posterior samples $(\theta)_t$ for each sampling method in figures 3a, 3b, 4a, and 4b.

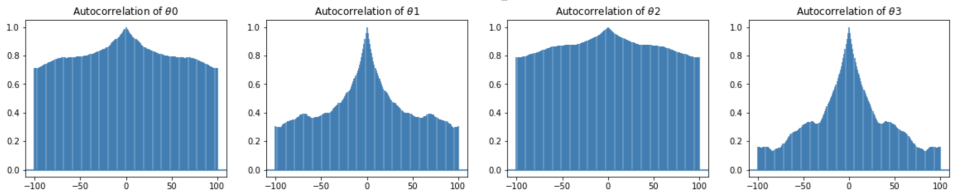


(a) SGLD (b) SGHD (c) SGLD (d) SGHD

Figure 2: Decision boundaries obtained by SGLD and SGHD for different classification tasks.



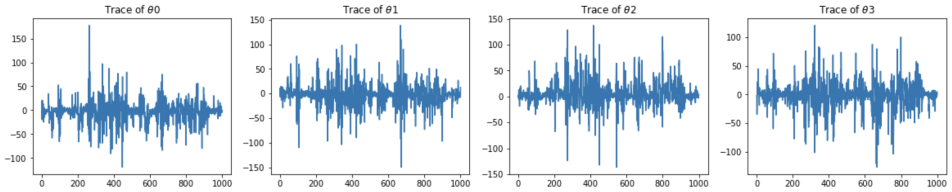
(a) Trace plots



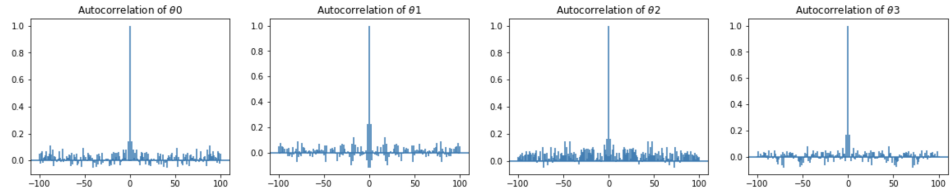
(b) Auto-correlation

Figure 3: Trace and auto-correlation plots of the first 4 parameters of θ obtained using SGLD.

Clearly, SGHD improves the mixing behavior and decorrelates the chain moves. The auto-correlation values are much less intense for SGHD than those obtained for SGMD. The same global tendency



(a) Trace plots



(b) Auto-correlation

Figure 4: Trace and auto-correlation plots of the first 4 parameters of θ obtained using SGHD.

	SGLD	SGHD	NUTS
Train acc.	0.801 ± 0.001	0.713 ± 0.042	0.801 ± 0.001
Test acc.	0.788 ± 0.003	0.708 ± 0.040	0.790 ± 0.002

Table 2: Training and test accuracies obtained using SGLD, SGHD, and NUTS.

was observed for the remaining 12 parameters of θ. This advantage in favor of the Hamiltonian process can therefore be exploited in the hope of mitigating the issue of sampling from one posterior mode. This hypothesis is yet to be confirmed on multimodal posterior distributions. Nevertheless, we clearly pay a price in terms of performance accuracy compared to SGLD and NUTS, as illustrated in table 2.

6 Conclusion

In this report, we illustrated the performance of SGLD in sampling from the posterior for the logistic regression classification task. We showed that the main limitation of this sampling method is the poor mixing of the chain it produces. This can result in getting trapped inside a small area around one single posterior mode. We suggest SGHD, a variant based on the Hamiltonian Monte Carlo with batch updates, and showed that it succeeds in decorrelating the chain moves yet performs less well than SGLD and NUTS. However, it is noteworthy to mention that the performance of SGHD relies on the estimation of the mass matrix M_H, which might not be easily accessible. Also, both SGLD and SGHD require preliminary steps of tuning the gradient step size parameters. Finally, the batch gradient update introduces in HMC provides little improvement in terms of computation time since, after all, the acceptance probability at each iteration needs to be computed over the whole training set.

References

[Neal(2012)] Radford Neal. MCMC using Hamiltonian Dynamics. *Handbook of Markov Chain Monte Carlo*, 06 2012. doi: 10.1201/b10905-6.

[Welling and Teh(2011)] Max Welling and Yee Whye Teh. Bayesian Learning via Stochastic Gradient Langevin Dynamics. In *Proceedings of the 28th International Conference on International Conference on Machine Learning*, ICML'11, page 681–688, Madison, WI, USA, 2011. Omnipress. ISBN 9781450306195.