Stochastic Gradient Hamiltonian Monte Carlo

Kristian Håland
EURECOM Engineering School
Campus SophiaTech, 450 Route des Chappes, 06410 Biot
Kristian.Haland@eurecom.fr // krishaal@stud.ntnu.no

Abstract

This report aims to explain and reproduce some of the results from the paper "Stochastic Gradient Hamiltonian Monte Carlo" [1]. The paper explores the effect of using a noisy estimate of the gradient in Hamiltonian dynamics. To mitigate the impact of this noise, the authors propose adding a friction term to counteract its effects, resulting in the Stochastic Gradient Hamiltonian Monte Carlo (SGHMC) algorithm. The results show that SGHMC is effective in simulated scenarios, producing an algorithm capable of handling noisy gradients while maintaining low computational cost. However, tests on real-world scenarios yield less accurate results, indicating that the algorithm requires careful hyperparameter tuning and training strategy.

1 Introduction

The Hamiltonian Monte Carlo algorithm is a powerful algorithm for sampling from a target distribution. It borrows concepts from physics, where we simulate a particle moving through a given state space, called a Hamiltonian dynamical system. One limitation of the HMC method is the need for computing gradients in the simulations. This can be costly when we have a lot of data available, and thus we can use minibatch. However, this introduces noise to the gradient estimate, which can yield unwanted results.
In the paper "Stochastic Gradient Hamiltonian Monte Carlo" [1], the effect of such a noisy estimate of the gradient is studied. Furthermore, a friction term is added to counteract the effect of the noisy estimate, resulting in the stochastic gradient HMC (SGHMC) algorithm. This report focuses on explaining the main theory given in the paper, as well as recreating some of the results obtained.

2 Theory

The following section provides an overview of the theoretical foundations underlying the simulations presented in Section 3.

2.1 Hamiltonian Monte Carlo

Our aim is to obtain the posterior distribution $p(\theta|\mathcal{D})$, where θ is the model parameters, and $x \in \mathcal{D}$ our dataset containing the observations. However, in many cases, such a posterior is intractable to compute. We can therefore use Markov Chain Monte Carlo (MCMC) algorithms, which is algorithms where we draw samples from a probability distribution. Thus it is possible to estimate the posterior instead. Throughout the presented theory, always keep in mind Bayes Theorem

$$p(\theta|\mathcal{D}) \propto p(\mathcal{D}|\theta)p(\theta), \tag{1}$$

Submitted to 38th Conference on Neural Information Processing Systems (NeurIPS 2024). Do not distribute.

meaning we always use the likelihood and prior of θ to approximate the posterior. One MCMC algorithm is the Hamiltonian Monte Carlo (HMC) algorithm. We define

$$p(\theta|\mathcal{D}) \propto exp(-U(\theta)), \tag{2}$$

where $U(\theta)$ is the potential energy function

$$U(\theta) = -\sum_{x \in D} \log p(x \mid \theta) - \log p(\theta). \tag{3}$$

The goal is for the algorithm to propose samples θ which has high probability given our observations. In order to effectively explore the state space, we introduce the auxiliary momentum variable r. The joint pdf of θ and r is

$$\pi(\theta, r) \propto \exp\left(-U(\theta) - \frac{1}{2} r^\top M^{-1} r\right), \tag{4}$$

where M is called the weight matrix. We can think of θ as a particle moving through our state space, and r as an associated momentum variable pushing us towards areas of high probability. We denote the right term in Equation (4) as kinetic energy, meaning π is composed of a potential and kinetic energy term. When simulating the trajectory of the particle, we simply extract the obtained θ, and omit r since its not used in the posterior function.

The trajectory of the particle is simulated with Hamiltonian dynamics

$$\begin{aligned} d\theta &= M^{-1} r dt \\ dr &= -\nabla U(\theta) dt. \end{aligned} \tag{5}$$

Note that this dynamic is defined as a continuos system, which is not applicable in a computer simulation, and thus discretization is needed. Such a discretization is the "leapfrog" method including an MH step. A more detailed explenation can be found in [1]. The main takeaway is the use of the stepsize ϵ for updating θ and r positions, as well as an MH step in order to not drift away from the target distribution over time.

2.2 Stochastic Gradient HMC

One limitation from the Hamiltonian dynamics in Equation (5) is the calculation of the gradient of $U(\theta)$. To get an accurate estimate, we should use all data from our dataset $\mathcal{D}$. However, this may be computationally heavy if $\mathcal{D}$ is large, and thus a minibatch $\widetilde{\mathcal{D}}$ can be used. Such an estimation of $\mathcal{D}$ yields a noisy estimate, where we assume the added noise to be Gaussian. We also assume all observations x to be independent. The estimated gradient becomes

$$\nabla \widetilde{U}(\theta) \approx \nabla U(\theta) + \mathcal{N}(0, V(\theta)), \tag{6}$$

where $V(\theta)$ is the covariance of the model parameters. In the following sections, we study the use of the estimate in Equation (6), as well as the use of extra terms for counteracting the effect of the noise.

2.2.1 Naïve Stochastic Gradient HMC

The naïve approach for the stochastic gradient HMC is simply replacing the gradient in the Hamiltonian dynamics with $\nabla \widetilde{U}(\theta)$ from Equation (6). The Hamiltonian dynamics becomes

$$\begin{aligned} d\theta &= M^{-1} r \, dt \\ dr &= -\nabla U(\theta) \, dt + \mathcal{N}(0, \, 2B(\theta) \, dt), \end{aligned} \tag{7}$$

where $B(\theta) = \frac{1}{2}\epsilon V(\theta)$. We can go back to thinking about θ as a particle. If we used the true gradient using all the data $\mathcal{D}$, the entropy of the particle would be preserved, as the particle is given no extra

energy, nor any energy is lost due to the absence of friction. Now, however, since we have noise injected, the energy of the particle can be changed. In fact, it will increase as $B(\theta)$ is positive definite [1]. As we simulate the trajectory, more and more energy is given to the particle, and we move away from the target distribution from which we are trying to sample.

2.2.2 Stochastic Gradient HMC with Friction

To make sure that we do not move away from the target distribution using the method in Section 2.2.1, we can either do a MH step often, or we can simulate the trajectory for a long time but get low acceptance rates for the proposed θ. To avoid these two possibilities, we can instead try to limit the effect of the noise itself. We do so by introducing a friction term, and the new Hamiltonian dynamics becomes

$$\begin{aligned} d\theta &= M^{-1}r\,dt \\ dr &= -\nabla U(\theta)\,dt - B(\theta)M^{-1}r dt + \mathcal{N}(0,\, 2B(\theta)\,dt). \end{aligned} \tag{8}$$

Here we still have injected noise which increases the entropy of the particle. However, this particle now experiences some friction. The particle can no longer "run away" as it pleases, and its energy is contained so that it stays around the target distribution.

There are two main benefits of using the Hamiltonian dynamics in Equation (8): First, we are using a minibatch of $\mathcal{D}$ to compute $\nabla U(\theta)$, which reduces the computational complexity. The second one is that by using a friction term, we do not need to do a MH step, which would increase the computational complexity, defeating the purpose.

2.2.3 Stochastic Gradient HMC in Practice

In the stochastic gradient HMC algorithms we have seen so far, we have assumed the noise model $B(\theta)$ to be known. In practical scenarios, this is not realistic, where we instead may have access to an estimate $\hat{B}$. Ongoing, we introduced a user specified friction term $C \geq \hat{B}$, which is a hyperparameter to be chosen. The final Hamiltonian dynamics to be used in the SGHMC algorithm is

$$\begin{aligned} d\theta &= M^{-1}r\,dt \\ dr &= -\nabla U(\theta)\,dt - CM^{-1}r\,dt \\ &\quad + \mathcal{N}\big(0,\, 2(C - \hat{B}\,dt\big) + \mathcal{N}\big(0,\, 2B(\theta)\,dt\big). \end{aligned} \tag{9}$$

All parameters are now user specified.

3 Experiments

The following sections recreates the simulations done in Sections 4.1 and 4.2 in the paper. SGLD is not included in the experiments.

3.1 Simulated Scenarios

The first experiment uses a synthetic potential energy term $U(\theta) = -2\theta^2 + \theta^4$, meaning we have access to the exact gradient $\nabla U(\theta)$. Thus, we define the estimated gradient as the true gradient corrupted by zero-mean Gaussian noise with standard deviation 2. Recalling Equation (2), we do not have access to the normalization coefficient in the posterior distribution. The recreated results are shown in Figure 1.

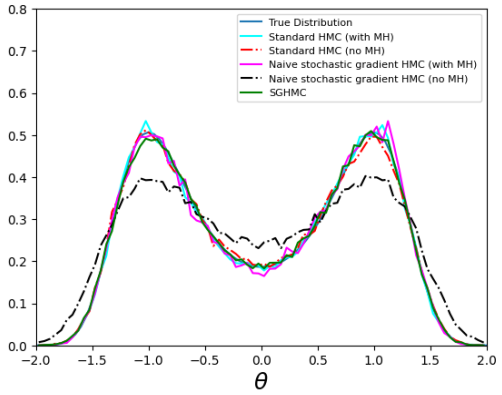

Figure 1: Sampled distributions for different sampling algorithms. The momentum r is re-sampled every 50 steps in all scenarios.

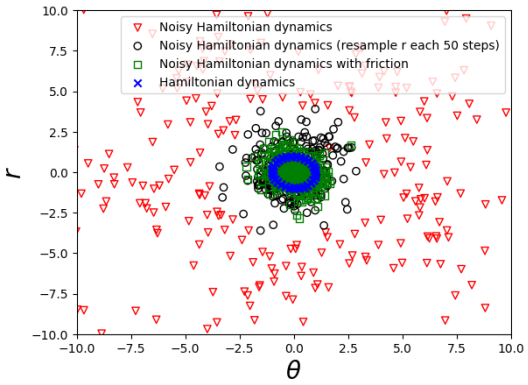

Figure 2: Hamiltonian dynamics for different sampling algorithms.

Simulations yields the same results as the paper, where the use of standard HMC, naive stochasic HMC with an MH step, as well as SGHMC follows the true distribution closely. Naïve stochastic gradient HMC without an MH step however deviates from the true distribution. These results underlines the main points of using SGHMC: We are not using a perfect gradient, and at the same time do not require an MH step. Thus we are efficiently exploring the true underlying distribution, while at the same time reducing computational complexity. Note however, that in order to get these results, a lot of noise had to be injected into the noisy gradient. This is the equivalent of using a small minibatch in a real world scenario.

Figure 2 shows the reproduced simulation of the hamiltonian dynamics. Using only the noisy gradient, we see that the trajectory diverges, meaning entropy of the particle increases. The consequence of this divergence is that we obtain samples far from the target distribution, yielding the unwanted deviation we saw in Figure 1. We can also see that the divergence is controlled by resampling the momentum r every 50 steps, yet we know that without an MH step, we are still not able to obtain the true target distribution. Lastly, we see that divergence is controlled when friction is added (SGHMC case). Thus we have ensured that the hamiltonian dynamics are well behaved, and at the same time yield the correct target distribution.

3.2 Bayesian Neural Networks for Classification

In this section, the test error over epochs are studied for optimization-based and sampling-based neural networks, using the MNIST dataset. Specifications of NN architectures and training strategy is further specified in [1]. Results are shown in Figure 3.

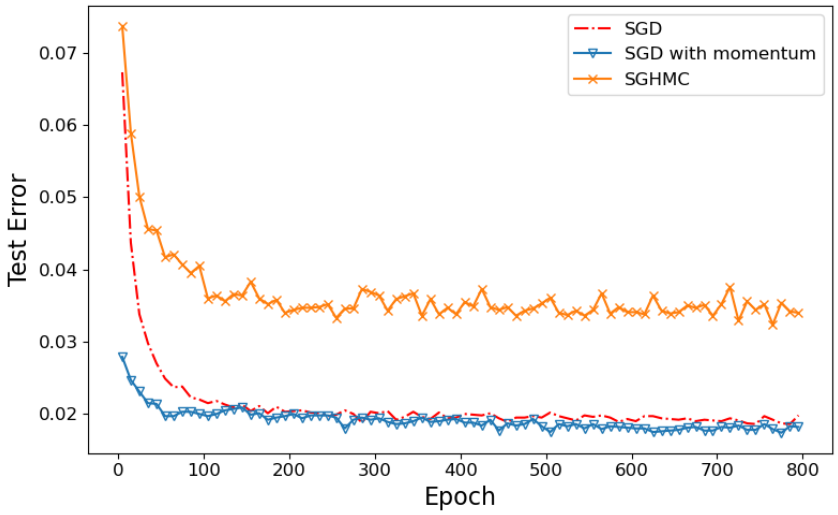

Figure 3: Test error over epochs for SGD, SGD with momentum and SGHMC on the MNIST dataset.

In all cases, the test error goes down to a relatively low level, indicating good generalization. Like the paper, SGD with momentum converges faster than normal SGD. Reproducing the obtained results for SGHMC however was deemed more challenging.

Initial testing resulted in slow convergence, where the model was highly influenced by the learning rate and friction C. The lowest obtained test error after 800 epochs was in the order 0.03. As a second experiment, training was first done for 100 epochs using SGD in order to approach a minima with low test error. After initial training, further training was done with a Bayesian approach in hopes of improved test error. After each epoch, the class of each samples was calculated by averaging the output of 10 posterior samples chosen at random. Unfortunately, test errors still remained constant, likely due to already being in and around a minima. Based on this analysis, we can say that achieving fast convergence during training is likely highly influenced by hyperparameters such as learning rate, friction C and the prior distribution $p(\theta)$. Also, initial starting point and running multiple chains of the parameters would be beneficial for better localization of the posterior.

4 Conclusion

In conclusion, simulated scenarios shows that using the SGHMC algorithm is technically doable, where we are able to achieve results close to the posterior. Thus we can utilize a noisy gradient, while still avoiding an MH step, which ultimately decreases computational cost. Tests on real-world scenarios using a Bayesian neural network on the MNIST dataset yielded suboptimal results, with reproduced test error and convergence performance worse than reported in the original paper. This highlights the importance of proper hyperparameter tuning and posterior exploration for SGHMC to be effective.

References

[1] Tianqi Chen, Emily B. Fox, and Carlos Guestrin.
Stochastic Gradient Hamiltonian Monte Carlo.
Proceedings of the 31st International Conference on Machine Learning (ICML), 2014.

A Appendix

`https://github.com/KristianHaaland/ASI_Project`