Reproducing the paper: *Stochastic Gradient Hamiltonian Monte Carlo* by Tianqi Chen, Emily B. Fox and Carlos Guestrin

Sam Adam-Day, Alexander Goodall, Theo Lewy and Fanqi Xu

Abstract

We reproduce the experiments contained in 'Stochastic Gradient Hamiltonian Monte Carlo' [CFG14] by Chen, Fox and Guestrin.

FiXme: Give more details in abstract

List of Corrections

1 Introduction

- Overview of paper and its context.
- Which experiments replicated, and rationale for this choice.
- Target questions of paper.
- Experimental methodology.
- Implementation details.
  - Integration with Pyro.
  - Which parts are new, and which are from publicly available code?
  - Details about how key aspects were implemented.
- Link to repository.
- New aspects?

2 Background

Hamiltonian Monte Carlo (HMC) ([Dua+87; Nea11]) is a Markov Chain Monte Carlo (MCMC) sampling algorithm. Given a target probability distribution — in our case the posterior distribution of a set of variables θ given independent observations $x \in \mathscr{D}$ — it produces samples by carrying out a random walk over the parameter space using Hamiltonian dynamics.

To begin with prior distribution $p(\theta)$ and likelihood $p(x \mid \theta)$. Using these we define the *potential energy* function U:

$$U(\theta) := -\sum_{x \in \mathscr{D}} \log p(x \mid \theta) - \log p(\theta)$$

Note that, using Bayes' rule, we have that the posterior $p(\theta \mid \mathscr{D}) \propto \exp(-U)$. Hamiltonian dynamics introduces an auxiliary set of momentum variables r. These dynamics have a physical interpretation in which an object moves about a landscape determined by U. We let this object have *mass matrix* M. Then $U(\theta)$ represents the potential energy of the object, and its kinetic energy is given by $\frac{1}{2} r^\top M^{-1} r$. The total energy of the system is a quantity known as the *Hamiltonian function*:

$$H(\theta, r) = U(\theta) + \frac{1}{2} r^\top M^{-1} r$$

The development of the system is governed by the following equations.

$$\begin{aligned} \mathrm{d}\theta &= M^{-1} r \, \mathrm{d}t \\ \mathrm{d}r &= -\nabla U(\theta) \, \mathrm{d}t \end{aligned}$$

To simulate these continuous dynamics in practice, we must use a discretised version of these equations. To correct for the inaccuracies introduced by doing so, it is necessary to make a *Metropolis-Hastings correction step*. A simple algorithm is given in Algorithm 1.

Algorithm 1 A simple HMC algorithm

for $t = 1, 2, \ldots$ **do**
  $r \sim \mathscr{N}(0, 1)$ ▷ Resample momentum
  $(\theta_0, r_0) = (\theta, r)$
  for $i = 1$ to m **do**
    $\theta \leftarrow \theta + \epsilon M^{-1} r$
    $r \leftarrow r - \epsilon \nabla U(\theta)$
  end for
  $u \sim \text{Uniform}[0, 1]$
  $\rho = \exp(H(\theta, r) - H(\theta_0, r_0))$ ▷ Acceptance probability
  if $u > \min(1, \rho)$ **then** ▷ Only accept new state with probability ρ
    $\theta = \theta_0$
  end if
end for

In practice, the dataset $\mathscr{D}$ may be large, and so running Algorithm 1 may be computationally expensive. One idea to combat this is to simulate the Hamiltonian system using only a subset of the data at a time, in analogy with stochastic gradient descent. Unfortunately, such a dynamical system can diverge quite rapidly from the true posterior distribution [Nea11], which necessitates frequent Metropolis-Hastings steps. Such steps are costly since they must be carried out using the whole dataset. The method 'Stochastic Gradient Hamiltonian Monte Carlo' (SGHMC) proposed in [CFG14] addresses this shortcoming. The idea is to incorporate friction into the dynamical system, which works to counteract the noise introduced by selecting a subset of the data.

To specify the SGHMC method, consider sampling a minibatch $\widetilde{\mathscr{D}} \subset \mathscr{D}$ uniformly at random. We estimate the gradient $\nabla U(\theta)$ using this minibatch as follows:

$$\nabla \widetilde{U}(\theta) = -\frac{|\mathscr{D}|}{|\widetilde{\mathscr{D}}|} \sum_{x \in \widetilde{\mathscr{D}}} \nabla \log p(x \mid \theta) - \nabla \log p(\theta)$$

Appealing to the Central Limit Theorem, we can take the noisy estimate $\nabla \widetilde{U}(\theta)$ to be normally distributed about $\nabla U(\theta)$, with some covariance matrix $V(\theta)$.

The naïve adaptation of HMC to this stochastic scenario simply replaces $\nabla U(\theta)$ with $\nabla \widetilde{U}(\theta)$ in Algorithm 1. The corresponding discrete system is then the ϵ-discretisation of the following dynamics:

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

where $B(\theta) = \frac{1}{2}\epsilon V(\theta)$.

The SGHMC method adds a 'friction term' $BM^{-1}r$ to the momentum update. Since we are unlikely to know the noise model B in practice, we instead take an estimate $\widehat{B}$ of B, together with a user-specified friction term $C \succeq \widehat{B}$ and simulate the following dynamics.

$$\begin{aligned} \mathrm{d}\theta &= M^{-1} r \, \mathrm{d}t \\ \mathrm{d}r &= -\nabla U(\theta)\,\mathrm{d}t - CM^{-1} r \, \mathrm{d}t + \mathcal{N}(0, 2(C - \widehat{B}(\theta))\,\mathrm{d}t) + \mathcal{N}(0, 2B(\theta)\,\mathrm{d}t) \end{aligned}$$

The algorithm is given in Algorithm 2. In the case where $\widehat{B} = B$, these dynamics accurately traverse from the posterior distribution. In practice, we must rely on an inaccurate estimate $\widehat{B}$. The simplest choice is $\widehat{B} = 0$. A better but more costly estimate is $\widehat{B}(\theta) = \frac{1}{2}\epsilon\widehat{V}(\theta)$, where $\widehat{V}$ is the observed (empirical Fisher) information [AKW12]. The friction term C can then be taken as a hyperparameter, and set so as to counteract the inaccuracies of the estimate $\widehat{B}$.

Algorithm 2 The SGHMC algorithm

for $t = 1, 2, \ldots$ **do**
  $r \sim \mathcal{N}(0, 1)$ ▷ Resample momentum
  for $i = 1$ to m **do**
    $\theta \leftarrow \theta + \epsilon M^{-1} r$
    $r \leftarrow r - \epsilon \nabla \widetilde{U}(\theta) - \epsilon C M^{-1} r + \mathcal{N}(0, 2(C - \widehat{B}(\theta))\epsilon)$
  end for
end for

3 Implementation Details

We implemented the following algorithms from scratch: HMC, SGHMC, SGLD, SGD, SGD (with nesterov momentum) and SGNUTS - a novel extension to SGHMC that uses ideas from the popular No U-Turn Sampler. All of our implementations subclass Pyro's `MCMCKernel` and are designed to be used directly with Pyro - a universal probabilistic programming language (PPL) written in Python. In Pyro the user specifies a model which is a probabilistic program (PP) that describes a posterior distribution $p(\theta \mid D)$ that we want to sample from; θ corresponds to the sampled parameters or latent parameters of the model and D corresponds to the observed parameters of the model.

Pyro already comes with the following MCMC samplers: HMC, MH and NUTS. So while the algorihtms we implemented are not novel they are in fact innovative since Pyro doesn't come with any of them already built. The main reason for choosing to implement our algorithms on top of Pyro is because Pyro comes with a method `initialize_model` that given a Pyro PP or model P transforms it into a potential function U. Once we have U we can pass the latent and observed parameters (θ, D) to U, which computes the negative log joint $-\log p(\theta, D)$. Bayes' rule tells us that $p(\theta \mid D) \propto p(\theta, D)$ which is typically all

we need for MCMC samplers and even simpler ones such as Importance and Rejection samplers. Pyro also comes with the method `potential_grad`, which given (θ, D) computes the gradient of U with respect to the parameters θ. The result is that we can specify any arbitrary PP and apply our algorithms to them - letting Pyro handle the transformation from PP to potential function and the gradient computations.

In traditional MCMC samplers the observed dataset D is constant and so once a Pyro PP has been transformed into a potential function $U(\theta) = -\log p(\theta, D)$ we need not change it. Unfortunately this is less straight forward for stochastic gradient samplers such as SGHMC and SGLD since we subsample the full dataset D by sampling minibatches $\tilde{D}$, where $\tilde{D} \subset D$. In both SGHMC and SGLD we require that the potential function has the form,

$$\tilde{U}(\theta) = -\frac{|D|}{|\tilde{D}|} \log p(\tilde{D} \mid \theta) - \log p(\theta)$$

Unfortunately when we subsample the dataset and call `initialize_model` Pyro doesn't explicitly supply us with the likelihood term $p(\tilde{D} \mid \theta)$ - it only gives us the negative log joint $-\log p(\theta, \tilde{D})$, so we had to modify Pyro's source code to get the desired behaviour above. As a result every time we generate a new sample using SGHMC or SGLD we have to call `initialize_model` so that it gives us the correct $\tilde{U}(\theta)$ for some given minibatch $\tilde{D}$, although this is a small price to pay for much quicker gradient computations.

4 Experiments

- Describe experiments and compare with results in the paper.

4.1 Simulated examples

4.2 Bayesian Neural Networks for Classification

5 Conclusion

- Analysis and discussion of findings.
- Suggest what could have been done with more time.

References

[AKW12] Sungjin Ahn, Anoop Korattikara and Max Welling. *Bayesian Posterior Sampling via Stochastic Gradient Fisher Scoring*. 2012. DOI: 10.48550/ARXIV.1206.6380. URL: https://arxiv.org/abs/1206.6380.

[CFG14] Tianqi Chen, Emily Fox and Carlos Guestrin. 'Stochastic Gradient Hamiltonian Monte Carlo'. In: *Proceedings of the 31st International Conference on Machine Learning*. Ed. by Eric P. Xing and Tony Jebara. Vol. 32. Proceedings of Machine Learning Research 2. Bejing, China: PMLR, June 2014, pp. 1683–1691. URL: https://proceedings.mlr.press/v32/cheni14.html.

[Dua+87] Simon Duane, A.D. Kennedy, Brian J. Pendleton and Duncan Roweth. 'Hybrid Monte Carlo'. In: *Physics Letters B* 195.2 (1987), pp. 216–222. ISSN: 0370-2693. DOI: https://doi.org/10.1016/0370-2693(87)91197-X. URL: https://www.sciencedirect.com/science/article/pii/037026938791197X.

[Nea11] Radford M. Neal. 'Handbook of Markov Chain Monte Carlo'. In: ed. by BySteve Brooks, Andrew Gelman, Galin Jones and Xiao-Li Meng. Chapman and Hall/CRC, 2011. Chap. MCMC Using Hamiltonian Dynamics, pp. 113–162. ISBN: 9780429138508.