Stochastic Bouncy Particle Sampler

Ari Pakman [* 1]   **Dar Gilboa** [* 1]   **David Carlson** [2]   **Liam Paninski** [1]

Abstract

We introduce a stochastic version of the non-reversible, rejection-free Bouncy Particle Sampler (BPS), a Markov process whose sample trajectories are piecewise linear, to efficiently sample Bayesian posteriors in big datasets. We prove that in the BPS no bias is introduced by noisy evaluations of the log-likelihood gradient. On the other hand, we argue that efficiency considerations favor a small, controllable bias, in exchange for faster mixing. We introduce a simple method that controls this trade-off. We illustrate these ideas in several examples which outperform previous approaches.

1. Introduction

The advent of the Big Data era presents special challenges to practitioners of Bayesian modeling because typical sampling-based inference methods have a computational cost per sample linear in the size of the dataset. This computational burden has been addressed in recent years through two major approaches (see (Bardenet et al., 2015) for a recent overview): (i) split the data into batches and combine posterior samples obtained in parallel from each batch, or (ii) use variants of the Markov Chain Monte Carlo (MCMC) algorithm that only query a subset of the data at every iteration. Our interest in the paper is in the latter approach, where many methods are based on modifying both steps of the Metropolis-Hastings (MH) algorithm: in the proposal step, only a mini-batch of the data is used, and the accept-reject step is either ignored or approximated (Korattikara et al., 2013; Bardenet et al., 2014). This strategy has been explored using proposals from Langevin (Welling & Teh, 2011), Riemannian Langevin (Patterson & Teh, 2013), Hamiltonian (Chen et al., 2014) and Riemannian Hamiltonian (Ma et al., 2015) dynamics. Other relevant works

include (Ahn et al., 2012; Ding et al., 2014).

Despite the success of the above approach, the partial accept-reject step is a source of bias, the precise size of which is difficult to control, and which tends to be amplified by the noisy evaluation of the gradient. This has motivated the search for unbiased stochastic samplers, such as the Firefly MCMC algorithm (Maclaurin & Adams, 2014), the debiased pseudolikelihood approach of (Quiroz et al., 2016), and the quasi-stationary distribution approach of (Pollock et al., 2016).

The present work is motivated by the idea that the bias could be reduced by starting from a rejection-free MCMC algorithm, avoiding thus the Metropolis-Hastings algorithm altogether. Two similar algorithms of this type have been recently proposed: the Bouncy Particle Sampler (BPS) (Peters & de With, 2012; Bouchard-Côté et al., 2015), and Zig-Zag Monte Carlo (Bierkens & Roberts, 2015; Bierkens et al., 2016). These algorithms sample from the target distribution through non-reversible, piecewise linear Markov processes. Non-reversibility (i.e., the failure to satisfy detailed balance) has been shown in many cases to yield faster mixing rates (Neal, 2004; Vucelja, 2014; Bouchard-Côté et al., 2015).

Our contributions in this paper are twofold. Firstly, we show that the BPS algorithm is particularly well suited to sample from posterior distributions of big datasets, because the target distribution is invariant under zero-mean noisy perturbations of the log-likelihood gradient, such as those introduced by using mini-batches of the full dataset in each iteration.

Stochastic variants of BPS or Zig-Zag that preserve exactly the target distribution have been proposed, such as Local BPS (Bouchard-Côté et al., 2015) or Zig-Zag with subsampling (ZZ-SS) (Bierkens et al., 2016), but they lead to extremely slow mixing because are based on overly conservative bounds (which moreover must be derived on a case-by-case basis, and in many cases may not hold at all). This leads us to our second contribution, the Stochastic Bouncy Particle Sampler (SBPS), a stochastic version of the BPS algorithm which trades a small amount of bias for significantly reduced variance, yielding superior performance (and requiring no parameter tuning or derivation of problem-specific bounds) compared to exist-

---

*Equal contribution   [1]Statistics Department and Grossman Center for the Statistics of Mind, Columbia University, New York, NY 10027, USA [2]Duke University, Durham, NC 27708, USA. Correspondence to: Ari Pakman <aripakman@gmail.com>.

Algorithm 1 Bouncy Particle Sampler

Initialize particle position $\mathbf{w}_0 \in \mathbb{R}^D$ and velocity $\mathbf{v} \in S^{D-1}$
while desired **do**
   Sample Poisson process first arrivals t_r, t_b with rates $\lambda_r, \lambda(t) = [\mathbf{v} \cdot \nabla U(\mathbf{w}_0 + \mathbf{v}t)]_+$
   Let $t = \min(t_b, t_r)$
   Move $\mathbf{w}_t = \mathbf{w}_0 + \mathbf{v}t$,
   if $t_b < t_r$ **then**
      Reflect $\mathbf{v} \leftarrow \mathbf{v} - 2\frac{(\mathbf{v} \cdot \nabla U(\mathbf{w}_t)) \nabla U(\mathbf{w}_t)}{||\nabla U(\mathbf{w}_t)||^2}$
   else
      Refresh: sample $\mathbf{v} \sim \text{Unif}[S^{D-1}]$
   end if
   Let $\mathbf{w}_0 \leftarrow \mathbf{w}_t$
end while
RETURN piecewise linear trajectory of $\mathbf{w}$

---

ing subsampling-based Monte Carlo methods. SBPS inherits the piecewise linear sample paths of BPS, and therefore enjoys faster convergence of empirical means, particularly of rapidly varying test functions, compared to more standard approaches.

We organize this paper as follows. In Section 2 we review the Bouncy Particle Sampler, in Section 3 we study the invariance of the target distribution under noise perturbations to the BPS updates, in Section 4 we introduce SBPS, and in Section 5 a preconditioned variant. In Section 6 we discuss related works and in Section 7 we illustrate the advantages of SBPS in several examples.

2. The Bouncy Particle Sampler

Consider a distribution $p(\mathbf{w}) \propto e^{-U(\mathbf{w})}$, $\mathbf{w} \in \mathbb{R}^D$, where the normalization factor may be intractable. The Bouncy Particle Sampler (BPS), proposed in (Peters & de With, 2012; Monmarché, 2014) and formalized and developed in (Bouchard-Côté et al., 2015), introduces a random velocity vector $\mathbf{v}$ distributed uniformly in the unit sphere S^{D-1}, and defines a continuous Markov process in $(\mathbf{w}, \mathbf{v})$. To describe this process we begin in discrete time and then take the continuous-time limit. Denoting time by t, consider a discrete-time Markov process that acts on the variables $(\mathbf{w}, \mathbf{v})$ as

$$(\mathbf{w}, \mathbf{v})_{t+\Delta t} = \begin{cases} (\mathbf{w} + \mathbf{v}\Delta t, \mathbf{v}) & \text{w/prob. } 1 - \Delta t[G]_+ \\ (\mathbf{w} + \mathbf{v}\Delta t, \mathbf{v}_r) & \text{w/prob. } \Delta t[G]_+ \end{cases} \quad (1)$$

where

$$[x]_+ = \max(x, 0), \quad (2)$$

$$G = \mathbf{v} \cdot \nabla U(\mathbf{w}), \quad (3)$$

$$\mathbf{v}_r = \mathbf{v} - 2\frac{(\mathbf{v} \cdot \nabla U(\mathbf{w})) \nabla U(\mathbf{w})}{||\nabla U(\mathbf{w})||^2}. \quad (4)$$

Note that G in (3) is the directional derivative of $U(\mathbf{w})$ in the direction $\mathbf{v}$, and $\mathbf{v}_r$ is a reflection of $\mathbf{v}$ with respect to the plane perpendicular to the gradient ∇U, satisfying $\mathbf{v}_r \cdot \nabla U = -\mathbf{v} \cdot \nabla U$ and $(\mathbf{v}_r)_r = \mathbf{v}$. In other words, the particle $\mathbf{w}$ moves along a straight line in the direction of $\mathbf{v}$ and this direction is reflected as (4) with probability $\Delta t[G]_+$. This probability is non-zero only if the particle is moving in a direction of lower target probability $p(\mathbf{w})$, or equivalently higher potential $U(\mathbf{w})$.

Applying the transition (1) repeatedly and taking $\Delta t \to 0$, the random reflection point becomes an event in an inhomogeneous Poisson process with intensity $[G]_+$. The resulting sampling procedure generates a piecewise linear Markov process (Davis, 1984; Dufour et al., 2015), and is summarized in Algorithm 1. Note that the algorithm also includes occasional resamplings of $\mathbf{v}$, to ensure ergodicity (Bouchard-Côté et al., 2015). Remarkably, in the limit $\Delta t \to 0$, the algorithm leaves the joint factorized distribution $p(\mathbf{w})p(\mathbf{v})$ invariant, as we review in Supp. Material A.1.

The Zig-Zag process (Bierkens & Roberts, 2015; Bierkens et al., 2016) is similar to BPS, but velocity components can take only ± 1 values, and the piecewise linear trajectories change direction only in a *single* coordinate at each random breakpoint. For a review of these methods, see (Fearnhead et al., 2016; Bierkens et al., 2017).

3. Noise Resilience and Big Data

3.1. Noise Resilience

Let us assume that only a noisy version of the gradient is available to compute the probability of bouncing and the reflected velocity in (4). In the Big Data scenario described below, this is the result of using a random subset of the data at each gradient evaluation, and can be represented as

$$\nabla \tilde{U}(\mathbf{w}) = \nabla U(\mathbf{w}) + \mathbf{n_w}, \qquad \mathbf{n_w} \sim p(\mathbf{n_w}|\mathbf{w}), \quad (5)$$

where $\mathbf{n_w} \in \mathbb{R}^D$ and $p(\mathbf{n_w}|\mathbf{w})$ has zero mean.

Theorem 1: *The invariance of $p(\mathbf{w}, \mathbf{v})$ under the BPS algorithm is unaffected by the zero-mean noise (5) if $\mathbf{n_{w_1}}$ and $\mathbf{n_{w_2}}$ are independent for $\mathbf{w}_1 \neq \mathbf{w}_2$.*

See Supp. Material A.2 for a proof sketch. Defining $\tilde{G} = \mathbf{v} \cdot \nabla \tilde{U}(\mathbf{w})$, the intensity of the inhomogeneous Poisson process $[\tilde{G}]_+$, which determines the time of the velocity bounce, now becomes stochastic, and the resulting point process is called a doubly stochastic, or Cox, process (Cox, 1955; Grandell, 1976). The effect of the gradient noise is to increase the average point process intensity, since $E\left[[\tilde{G}]_+\right] \geq \left[E[\tilde{G}]\right]_+$, from Jensen's inequality. This
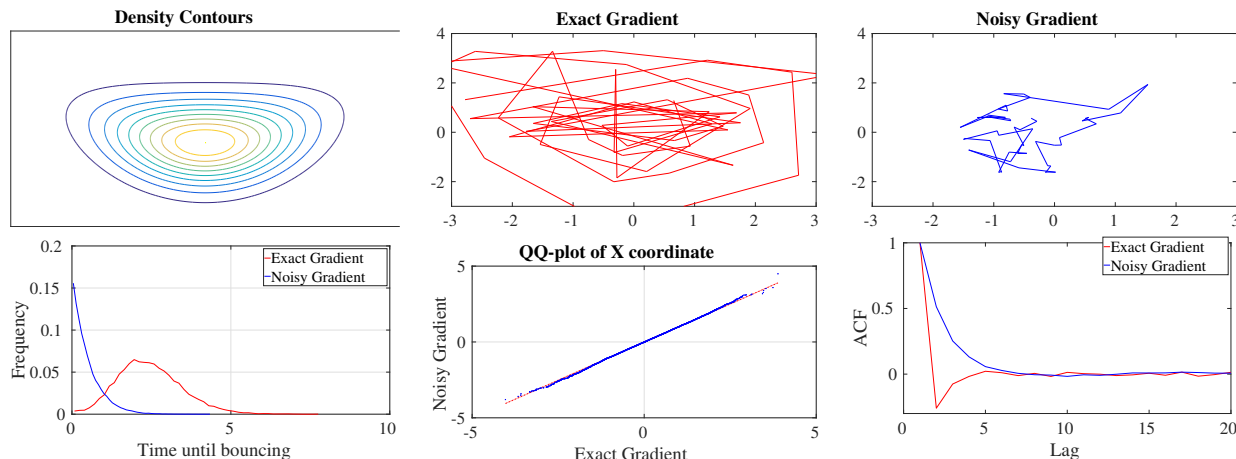
Figure 1: Noisy vs. noiseless gradients in BPS. *Above*: Contour plot of the 2D density considered and sample BPS trajectories during 50 bounces, with exact and noisy gradients. The noise was sampled in each component from a $\mathcal{N}(0, 5^2)$ distribution. *Below, Left*: smoothed histogram of travel times until bouncing, with shorter times for noisy gradients. *Middle*: QQ-plot of one of the coordinates, showing that the invariant distribution is not changed by the noise. *Right*: ACFs of one of the coordinates, with slower mixing per iteration in the noisy case. However, note that these ACF plots do not account for computational cost per iteration.

leads to more frequent bounces and typically a slower mixing of the Markov process, as illustrated in Figure 1.

Many Cox processes are based on Poisson intensities obeying stochastic differential equations, or assume that the joint distribution at several $\mathbf{w}$'s has a non-trivial $\mathbf{w}$-dependent structure. Our case is different because we assume that $\mathbf{n}_{\mathbf{w}_1}$ and $\mathbf{n}_{\mathbf{w}_2}$ are independent even when $\mathbf{w}_1$ and $\mathbf{w}_2$ are infinitesimally close.

3.2. Sampling from Big Data posteriors

In a prototypical Bayesian setting, we have a prior $f(\mathbf{w})$, i.i.d. data points x_i, and the negative log-posterior gradient is

$$\nabla U(\mathbf{w}) = -\nabla \left[\log f(\mathbf{w}) + \sum_{i=1}^{N} \log p(x_i | \mathbf{w}) \right]. \quad (6)$$

When N is big we consider replacing the above gradient by the noisy approximation

$$\nabla \tilde{U}(\mathbf{w}) = -\nabla \left[\log f(\mathbf{w}) + \frac{N}{n} \sum_{i=1}^{n} \log p(x_{r_i} | \mathbf{w}) \right], \quad (7)$$

where $n \ll N$ and the n indices $\{r_i\}$ are sampled randomly without replacement. To sample from the posterior using the noisy gradient (7), we want to simulate the first arrival time in a doubly stochastic Poisson process with random intensity $[\tilde{G}(t)]_+$, where

$$\tilde{G}(t) = \mathbf{v} \cdot \nabla \tilde{U}(\mathbf{w} + \mathbf{v}t). \quad (8)$$

Note that $\tilde{U}$ is a stochastic process, and noise independence for different $\mathbf{w}$'s implies that different t's require independent mini-batches. Out of several methods to sample from

(noisy) Poisson processes, the thinning method (Lewis & Shedler, 1979) is compatible with the noise independence assumption. This is a form of rejection sampling which proposes a first arrival time t, sampled from an inhomogeneous Poisson process with intensity $\lambda(t)$ such that $\lambda(t) \geq [\tilde{G}(t)]_+$ The particle moves a distance $t\mathbf{v}$, and accepts the proposal to bounce the velocity with probability $[\tilde{G}(t)]_+/\lambda(t)$. Note that this accept-reject step is different from the MH algorithm (Robert & Casella, 2013), since the particle always moves the distance $t\mathbf{v}$, and a rejection only affects the velocity bouncing. This can greatly improve the efficiency of the sampler. As in the noiseless case, one should in general also resample $\mathbf{v}$ occasionally, to ensure ergodicity (Bouchard-Côté et al., 2015), although in the examples we considered this was not empirically necessary, since the mini-batch noise serves to randomize the velocity sufficiently, preventing "non-ergodic" trajectories that do not explore the full space.

In some special cases one can derive a bound $\lambda(t)$ that always holds (Bouchard-Côté et al., 2015; Bierkens et al., 2017). But this is atypical, due to the dependence of $\tilde{G}(t)$ in (8) on the changing velocity $\mathbf{v}$ and the mini-batch noise. Even when such bounds do exist, they tend to be conservatively high, leading to an inefficient sampler with many rejected proposals (wasting many mini-batches of data) before accepting.

Instead, we propose below an adaptive approximate bound which achieves a bias-variance trade-off between the frequency of the bounce proposals and a controllable probability of bound violation.

4. Proposal from Local Regression

Our approach to an adaptive and tractable proposal intensity $\lambda(t)$ relies on a predictive model of $\tilde{G}$ based on previous observations; the key idea is to exploit the correlations between nearby $\tilde{G}$ values. The upper value of the resulting predictive confidence band can then be used as $\lambda(t)$, and this band is adaptively updated as more proposals are generated.

While there are many possibilities for such a predictive model, we found that a simple local linear model was very effective and computationally trivial. Consider then the linear regression of m observed values $\tilde{G}_i \equiv \tilde{G}(t_i)$ since the previous bounce,

$$\tilde{G}_i = \beta_1 t_i + \beta_0 + \varepsilon_{t_i} \qquad \varepsilon_{t_i} \sim N(0, c_{t_i}^2), \tag{9}$$

where $i = 1, \ldots, m$ and the noise variance can be estimated from the mini-batch in (7) as

$$c_t^2 = \frac{N^2}{n}(1 - \frac{n}{N})\mathrm{Var}_i\left[\mathbf{v} \cdot \nabla \log p(x_{r_i}|\mathbf{w})\right]. \tag{10}$$

Here Var_i denotes the sample variance of the mini-batch, and we included the *finite population correction factor* $(1 - \frac{n}{N})$ because the indices $\{r_i\}$ are sampled without replacement. The Gaussian noise assumption in $\tilde{G}(t)$ in (9) is valid when the mini-batch is sufficiently large that we can appeal to a central limit theorem. (For heavy-tailed noise we could consider more robust estimators, but we do not pursue this direction here.)

Adding a Gaussian prior $N(\mu, \sigma^2)$ to β_1, and defining $\mathbf{x}_i \equiv (1, t_i)$, the log posterior of $\boldsymbol{\beta} = (\beta_0, \beta_1)^T$ is

$$2 \log p(\boldsymbol{\beta}|\{t_i, \tilde{G}_i, c_{t_i}^2\}) = -\sum_{i=1}^{m} \frac{(\tilde{G}_i - \mathbf{x}_i \cdot \boldsymbol{\beta})^2}{c_{t_i}^2}$$
$$-\frac{(\beta_1 - \mu)^2}{\sigma^2} + const.$$

Let $\hat{\boldsymbol{\beta}}$ and $\boldsymbol{\Sigma}$ be the mean and covariance of this distribution. Using these estimates, we obtain the predictive distribution $\hat{G}(t)$ for $\tilde{G}(t)$ for $t > t_m$,

$$\hat{G}(t) = \hat{\beta}_1 t + \hat{\beta}_0 + \eta_t \qquad \eta_t \sim N(0, \rho^2(t)) \tag{11}$$

$$\text{where} \quad \rho^2(t) = \mathbf{x}\boldsymbol{\Sigma}\mathbf{x}^T + c_{t_m}^2 \tag{12}$$

with $\mathbf{x} = (1, t)$. Note that as usual the noise variance is different in (9) and (11), since in (9) we are fitting observed pairs $\tilde{G}_i, t_i$, while in (11) we are predicting the value of $\tilde{G}(t)$ and we include the uncertainty from the $\hat{\boldsymbol{\beta}}$ estimates. Also, for simplicity we extrapolate the observation noise to be the same as in the last mini-batch, $c_{t_m}^2$.

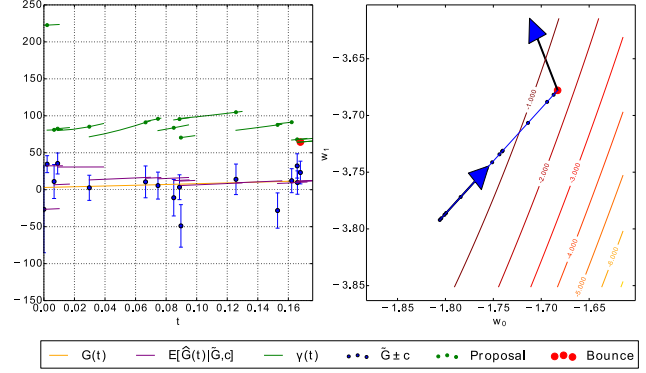We can now construct a tractable approximate thinning proposal intensity by choosing a confidence band multiple k,

Figure 2: Thinning proposal intensity for bounce times from a linear regression predictive confidence interval applied to a two-dimensional logistic regression posterior with $N = 1000, n = 100$. *Left:* Starting from $t = 0$, the piecewise linear intensity $\gamma(t)$ is used to propose bounce times (green points). As these proposals are rejected additional observations $\tilde{G}_i$ are made until a proposal is accepted (red point). The decrease in the slope of $\gamma(t)$ indicates the decreasing uncertainty in the estimated regression parameters as observations increase; note that the linear approximation for the true $G(t)$ is quite accurate here. Note also the reduced observation frequency at lower values of $G(t)$ indicating more efficient sampling than is achievable with the constant and much higher bounds used in (Bouchard-Côté et al., 2015; Bierkens et al., 2016), which were in the range $[10^4, 2 * 10^4]$ for this data. *Right*: The corresponding SBPS particle trajectory, with arrows indicating the initial velocity and the velocity after the bounce. The contours show the Laplace approximation of the log posterior.

and defining $\gamma(t)$ as a linear interpolation between selected points along the non-linear curve

$$\hat{\beta}_1 t + \hat{\beta}_0 + k\rho(t). \tag{13}$$

The proposal intensity is now $\lambda(t) = [\gamma(t)]_+$, and sampling from an inhomogeneous Poisson process with piecewise linear rate $\lambda(t)$ can be done analytically using the inverse CDF method. When a bounce time is proposed at time t, the particle moves a distance $t\mathbf{v}$, a noisy observation $\tilde{G}(t)$ is made as in (8) and the bounce time is accepted with probability $\min(1, [\tilde{G}(t)]_+/\lambda(t))$. If the bounce is accepted, the velocity is reflected as in (4) (using $\tilde{U}$ instead of U), and the set of observed values is reinitialized with $(-\tilde{G}(t), c_t)$, which are the values one would obtain from sampling the same mini-batch *after* the bounce, since $\mathbf{v}_r \cdot \tilde{U} = -\mathbf{v} \cdot \tilde{U} = -\tilde{G}(t)$. On the other hand, if the proposal is rejected, the observed $(\tilde{G}(t), c_t)$ are added to the set of observed values. The hyperparameters μ, σ^2 of the regression model can be learned by performing, after each bounce, a gradient ascent step on the marginal likelihood, $p(\{\tilde{G}_i\}|\mu, \sigma^2)$; this gradient can be computed analytically and does not significantly impact the computational cost.

The linear model for $\tilde{G}$ is good when the target distribution can be locally approximated by a Gaussian, since $\tilde{G}(t)$ in (8) is a projection of the derivative of the negative log

posterior. When the posterior is highly non-Gaussian, a decaying weight can be used for more-distant observations, leading to a local regression; the scale of this decay can be fit again via stochastic gradient ascent on the predictive likelihood. We have also explored a Gaussian Process regression model, but it did not improve over the linear model in the cases we considered. In Supp. Material E we discuss a potential problem with our approach in the case of multimodal distributions, and propose a solution for such cases.

Finally, note that the directional derivative of $\tilde{U}(\mathbf{w})$ needed in (8) can in many cases be computed at a cheaper cost (by a factor of $d = \dim(\mathbf{w})$) than the full gradient. The latter is only needed when a bounce is accepted. This is in contrast to other gradient based samplers which require the full gradient at every step.

We dub this approach to BPS with noisy gradients Stochastic BPS (SBPS). See Supp. Material C for pseudocode. Figure 2 illustrates the evolution of these dynamic proposal intensities in a simple example. In Section 5, we consider a variant to SBPS, called pSBPS, that learns a diagonal preconditioning factor for the gradient, and leads to a more efficient exploration of the space when the posterior is highly anisotropic and roughly axis-aligned.

4.1. Bias in the Samples

The constant k in (13) controls the tradeoff between bias from possible $[\tilde{G}(t)]_+/\lambda(t) > 1$ cases and lower computational cost: higher k leads to a more conservative (higher) proposal intensity and therefore a less-biased but more data-inefficient sampler. We present a bound on the Wasserstein distance between the exact and bias distributions in Supp. Material B, and explore this bias-variance tradeoff further in Supp. Material F. A quick bias diagnostic is the rate at which the bound is violated, i.e., cases with $[\tilde{G}(t)]_+/\lambda(t) > 1$; if this rate is significantly higher than expected under the local linear regression model, then a different approach should be considered.

5. Preconditioned SBPS

Consider now the linear transformation $\mathbf{w} = \mathbf{A}\mathbf{z}$ with an arbitrary square matrix $\mathbf{A}$. A distribution $p(\mathbf{w})$ of interest can be expressed in terms of $\mathbf{z}$ as

$$
\begin{aligned}
p_z(\mathbf{z})d\mathbf{z} &= p(\mathbf{w}(\mathbf{z}))d\mathbf{w} = p(\mathbf{A}\mathbf{z})|\mathbf{A}|d\mathbf{z}\,, \quad (14) \\
&= \exp(-U_z(\mathbf{z}))d\mathbf{z}\,. \quad (15)
\end{aligned}
$$

The SBPS algorithm can be applied to the density $p_z(\mathbf{z})$ using the gradients of $U(\mathbf{w})$. For this note that $\nabla_z U_z(\mathbf{z}) = \mathbf{A}\nabla_w U(\mathbf{w})$. The Poisson intensity to compute bounces is $[G]_+$, with $G = \mathbf{v} \cdot \mathbf{A}\nabla U(\mathbf{w})$, and the velocity reflection
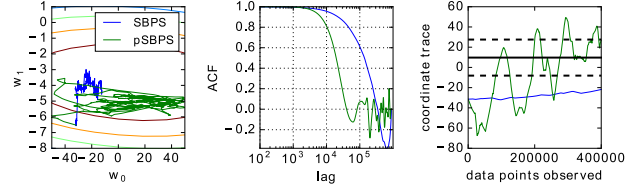

Figure 3: Effect of diagonal preconditioning on SBPS performance. Sampling is from the logistic regression posterior as described in Section 7.1 , with $d = 20, N = 1000, k = 3, n = 100$. The preconditioner parameters are $\beta = .99, \epsilon = 10^{-4}$. Left: Contour plots of posterior log likelihood under the Laplace approximation. Center, right: ACF and trajectories in the direction of greatest covariance.

is computed as

$$
\mathbf{v}_r = \mathbf{v} - 2\frac{(\mathbf{v} \cdot \mathbf{A}\nabla U(\mathbf{w}))\mathbf{A}\nabla U(\mathbf{w})}{||\mathbf{A}\nabla U(\mathbf{w})||^2}\,. \quad (16)
$$

The piecewise linear trajectory $\mathbf{z}_t = \mathbf{z}_0 + \mathbf{v}t$ becomes $\mathbf{w}_t = \mathbf{w}_0 + \mathbf{A}\mathbf{v}t$. The matrix $\mathbf{A}$ is called a preconditioner in the optimization literature, but can also be used in a sampling context to reduce anisotropy of posterior distributions; it is often the case that a good preconditioner is not known in advance but is instead learned adaptively (Duchi et al., 2011).

We use a diagonal preconditioner for simplicity. Denoting the ith component at the jth evaluation of the gradient by g_i^j, we define

$$
\begin{aligned}
a_i^j &= \beta(g_i^j)^2 + (1 - \beta)a_i^{j-1}, \quad (17) \\
\tilde{a}^j &= \frac{1}{d}\sum_{i=1}^d \frac{1}{\sqrt{a_i^j + \epsilon}}\,, \quad (18)
\end{aligned}
$$

for some $0 \le \beta \le 1, \epsilon \ll 1$. The preconditioner at iteration j is defined as $\mathbf{A}^j = \text{Diag}\left(\frac{\tilde{a}^j}{\sqrt{a_i^j + \epsilon}}\right)$. This is the same preconditioner used in (Li et al., 2016), up to the $\tilde{a}^j$ factor; the latter is needed here in order to prevent scaling of $\tilde{G}$.

As noted in (Li et al., 2016), a time dependent preconditioner requires adding a term proportional to $\frac{\partial \mathbf{A}^j}{\partial \mathbf{w}}$ to the gradient, yet this term is negligibly small and can be ignored when $\beta \approx 1$, since in this parameter regime the preconditioner changes slowly as a function of j and thus of $\mathbf{w}$.

We call this preconditioned variant pSBPS. It performs favorably compared to SBPS when the posterior is anisotropic and axis-aligned, since we use a diagonal approximation of the Hessian in the preconditioner. See (Bierkens et al., 2017) for a related approach. As Figure 3 shows, pSBPS converges to the posterior mode faster than SBPS, and mixes faster in the direction of greatest covariance.[1]

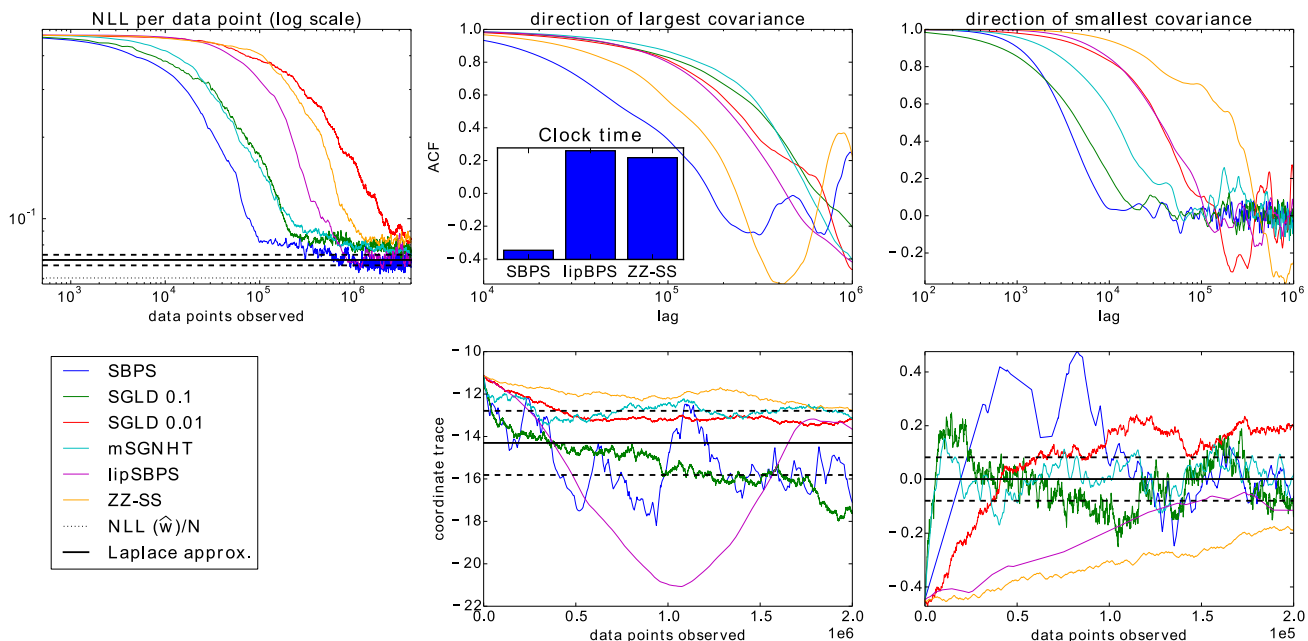[1] pSBPS code at https://github.com/dargilboa/SBPS-public.

Figure 4: Logistic regression posterior sampling, with $d = 20$, $N = 1000$, $k = 3$, $n = 100$ (best seen in color). *Top Left:* Negative Log Likelihood (NLL) per data point of samples from SBPS compared with SGLD (step sizes $0.01, 0.1$), mSGNHT (step size 0.1), lipSBPS and SS-ZZ (see text for definitions and references), all initialized at the same random positions. Also shown are the normalized NLL of the MAP estimator $NLL(\hat{\mathbf{w}})/N$ and the mean $\pm$ std. dev. of the Laplace approximation NLL (distributed as $\frac{1}{2N}\chi^2(d) + NLL(\hat{\mathbf{w}})/N$). The continuous samplers (SBPS, SS-ZZ, lipSBPS) were run to match the data cost of the discrete (SGLD, mSGNHT), and for their ACFs we discretized the continuous paths uniformly to obtain the same number of samples. Note that SBPS is the fastest to converge. *Center/Right:* Trajectories and ACFs in the directions of largest and smallest eigenvalues of the Laplace approximation inverse Hessian. The ACFs were calculated after burn-in, while the trajectory plots only show initial convergence. *Inset:* CPU runtime for 100 epochs, showing a $\times 35$ advantage of $n = 100$ SBPS over $n = 1$ SS-ZZ and lipSBPS

6. Related Works

Biased Samplers: Many stochastic gradient samplers (e.g. (Welling & Teh, 2011)) can be formulated exactly using a Wiener process (Ma et al., 2015), but they are biased because (i) the Gaussian assumption in the noise may not hold for small mini-batches, and (ii) the MH correction to the time discretization is avoided or approximated. Recently, irreversible samplers have been studied in this context (Ma et al., 2016). Choosing the step size in these samplers can be quite challenging, as discussed below: too-large step sizes increase the bias, while too-small step sizes slow the mixing, and in generic high-dimensional examples there is no way to automatically tune the step size (though see (Giles et al., 2016) for recent progress). In contrast, the bias in SBPS, controlled by the constant k, does not come from time discretization, but from easy-to-track violations of the thinning bound when $[\tilde{G}(t)]_+/\lambda(t) > 1$.

Exact non-BPS-like Samplers: Firefly MCMC (Maclaurin & Adams, 2014) augments the target distribution with one binary variable per data point, and yields unbiased samples while only querying a subset of data points at each iteration. But it needs distribution-dependent lower bounds on the likelihood and requires an initial full sweep of the

data. Also mixing can be extremely slow (Quiroz et al., 2015; Bardenet et al., 2015), and all the dataset must be available for access all the time.

Two recent novel proposals are (Quiroz et al., 2016), based on debiased pseudolikelihood combined with variance reduction techniques, and (Pollock et al., 2016), based on quasi-stationary distributions. These methods are relatively more complex, and we have not yet systematically compared them against SBPS.

Exact BPS-like Samplers: Two subsampling variants of BPS which preserve the exact distribution are Local BPS (Bouchard-Côté et al., 2015), that needs a preprocessing step of computational cost $O(N \log N)$, and ZZ-SS (Bierkens et al., 2016). In these approaches, the requirement to preserve the distribution exactly leads to extremely conservative thinning bounds, which in turn yield a very slow exploration of the space, as we will see below. Also, the bounds need to be rederived for each new model (if possible at all), unlike SBPS which can be used for any differentiable posterior distribution.

7. Experiments

7.1. Logistic Regression

Although simpler MCMC methods perform well in Bayesian logistic regression (BLR) models (Chopin & Ridgway, 2015), we begin with this well-understood case for comparing SBPS against a few of the existing stochastic MCMC methods discussed in the previous section. To generate the data, we sampled the components of the true $\mathbf{w} \in \mathbb{R}^d$ from Unif$[-5, 5]$ and N data points $\{\mathbf{x}_i\}$ from a d-dimensional zero-mean Gaussian, with one component of the diagonal covariance set to 6 and all the rest to 1. Labels $\{y_i\}$ are drawn from $y_i \sim \text{Bern}(\sigma(\mathbf{w} \cdot \mathbf{x}_i))$, where $\sigma(x) = 1/(1 + e^x)$. In the regime $d \ll N$ the Laplace approximation holds fairly well, providing another good comparison method. Figure 4 shows results for $N = 1000, d = 20, k = 3, n = 100$.

We run comparisons against the biased stochastic samplers Stochastic Gradient Langevin Dynamics (SGLD) (Welling & Teh, 2011) and multivariate Stochastic Gradient Nose-Hoover Thermostat (mSGNHT) (Li et al., 2015) with fixed step sizes. As noted above, choosing optimal step sizes for these samplers is challenging. To allow SGLD and mS-GNHT to perform best, we performed a scan to find the largest (fastest-mixing) step size that did not lead to overly large bias compared to the Laplace approximation. (Note, importantly, that this scan is expensive and is not possible in high-dimensional examples where the Laplace approximation does not hold - precisely the cases where MCMC methods are most valuable.) See Supp. Material E for details of this scan, which led to an optimal step size of 0.1 for SGLD. Larger step sizes led to visible biases in the samples (not shown); we also show the results with step size 0.01 for comparison to note that the results do depend sensitively on this parameter.

We also compare against ZZ-SS. Instead of Local BPS, we ran comparisons against an unbiased method we call lipS-BPS (short for Lipshitz BPS), where the velocity bounces occur as first arrival events in a Poisson process with noisy intensity $[\mathbf{v} \cdot \nabla \tilde{U}(\mathbf{w})]_+$ built from a noisy gradient (7) of minimal size $n = 1$, and simulated with thinning using an *exact* upper bound derived in Supp. Material F. One can verify that the resulting stochastic process is identical to that of Local BPS. Our bound is higher than that used in (Bouchard-Côté et al., 2015) by up to a factor of 2, which results in up to twice as many bounce proposals. On the other hand, our bound can be computed in $O(N)$ time, does not require non-negative covariates, and can be used also for $n > 1$. Again, we note that this lipSBPS method, like Local BPS and ZZ-SS, are not generally applicable because the derived bounds only apply in special cases.

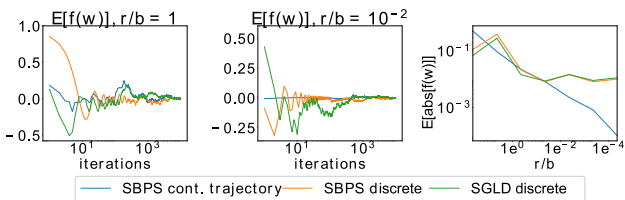The results of Figure 4 show that SBPS outperforms the op-

Figure 5: Estimated mean of $f(w) = \sin((w - \hat{w}/r)$, under continuous and discrete samples, with different ratios r/b, where $b \approx 2 \times 10^{-2}$ is the average linear trajectory length. The posterior distribution and settings are as in Figure 4. Assuming the Laplace approximation holds, the expectation of f is 0. *Left:* For $r/b = 1$ there is little difference between continuous or discrete samples. *Center:* For $r/b = 10^{-2}$ the continuous mean converges faster than the discrete. *Right:* Expectation of the absolute value of the test function averaged over 5 runs of 1000 epochs, as a function of r/b. The advantage of the continuous expectation when this ratio is $r/b \ll 1$ is evident.

timally tuned SGLD and mSGNHT, and converges orders of magnitude faster than lipSBPS and ZZ-SS. While the latter two methods are unbiased, our results suggest that the small bias introduced by SBPS is worth the massive reduction in variance.

In Supp. Material F we explore the effects of the hyper-parameters: k, n, and $\mathbf{v}$ refresh rate λ_r. The conclusion is that in this logistic example no manual hyperparameter tuning was required (in stark contrast to the careful step size tuning required for SGLD): the bias-controlling constant k can be set in the range $k \in [3, 5]$ (consistent with the tails of the Gaussian in the linear regression model) and the mini-batch size n should be small, but large enough for the CLT to justify the noise term in (9); $n = 100$ worked well, but the results were not sensitively dependent on n. For small values of n the mini-batch variability provided sufficient velocity randomness that no additional velocity refreshes were necessary, so we did not have to tune λ_r either.

The comparison to pSBPS shows an improvement in the rate of convergence to the posterior mode. The MAP estimator $\hat{\mathbf{w}}$ was calculated using SAG (Roux et al., 2012), and the Hessian was computed exactly.

7.2. Continuous Trajectory Sampling

A unique feature of BPS-like samplers is that their output is a continuous trajectory. Given $\mathbf{w}_0$ and a set of R velocities and bounce times $\{\mathbf{v}_i, t_i\}$, the estimated expectation of a test function $f(\mathbf{w})$ is

$$\langle f(\mathbf{w}) \rangle_{BPS} \equiv \frac{1}{T} \sum_{i=0}^{R-1} \int_0^{t_i} f(\mathbf{w}_i + \mathbf{v}_i t) dt \qquad (19)$$

where $\mathbf{w}_{i+1} = \mathbf{w}_i + \mathbf{v}_i t_i$ and T is the total particle travel time. For simple test functions this integral is analytic, while more generally it can be computed numerically
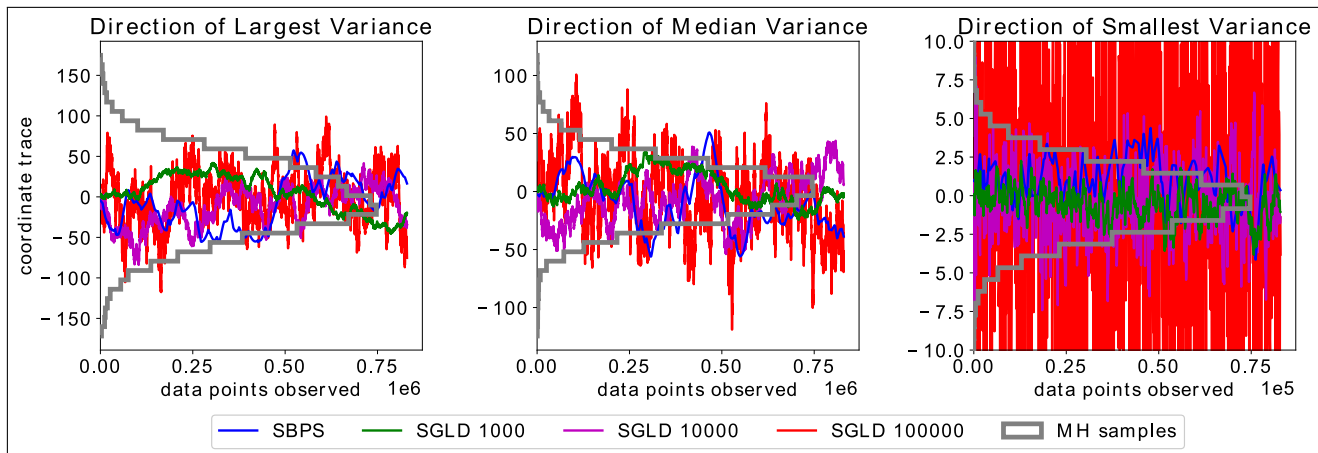
Figure 6: Neural network posterior sampling for a single hidden layer network trained on MNIST. $d = 192, N = 8340, n = 500$. For SBPS $k = 3$. The posterior is compared to an expensive Metropolis-Hastings run. SBPS shows comparable mixing to an appropriately chosen SGLD without the need for a scan over step sizes. As can be seen, a poor choice of SGLD step size can lead to slow mixing or bias in the narrow directions of the target

with standard efficient one-dimensional quadrature methods. When $f(\mathbf{w})$ varies across a characteristic length r shorter than the average trajectory length b of the linear segments, we intuitively expect the error in the estimate (19) to be smaller than in estimators based on discrete samples. Note that this advantage tends to diminish for higher SBPS noise, since the linear segments become shorter.

Figure 5 explores empirically this idea in a simple setting by comparing the value of the expectation of $f(w) = \sin((w - \hat{w})/r)$ under the posterior distribution of the logistic example considered above. Here $(w, \hat{w})$ are the first coordinates of the vectors $(\mathbf{w}, \hat{\mathbf{w}})$, $\hat{\mathbf{w}}$ is the MAP value, and r the characteristic length of f. As expected, the error in the expectation is lower in the continuous case for $r/b < 1$.

7.3. Neural Network Posterior Sampling

We considered a simple model of one hidden layer followed by a softmax. For Bayesian approaches to neural networks see (Neal, 2012; Gal, 2016). The likelihood was the standard cross entropy with an additional L_2 regularization term $L = -\sum_{i=1}^{N} \log(p_i) + \frac{c}{2} \sum_{j=1}^{d} \mathbf{w}_j^2$ where p_i is the probability of classifying the ith example correctly. L was approximated via subsampling, and $c = 0.001$. This architecture was trained on the MNIST dataset. A subset of the training set was preprocessed by downsampling the images to 7×7, removing pixels that are 0 for all training examples and decreasing the number of digits to 4. The resulting training set size was $N = 8340$. The resulting dimensionality of the posterior was $d = 192$. Mini-batch size was $n = 500$ for all methods. All weights were initialized at 0 and all methods were run for 10^4 epochs. SBPS is compared with SGLD at different step sizes, and performance

is comparable to SGLD with an appropriate step size without requiring an expensive scan over step sizes. Since the additional regularization term can lead to unbounded gradients of the log posterior $\nabla U(\mathbf{w})$ one can no longer use the bounds derived for the Local BPS and ZZ-SS algorithms and thus they cannot be applied to this problem without further work. This is not the case for SBPS. The posterior is not Gaussian due to the likelihood terms and thus the Laplace approximation is not effective unless the posterior is dominated by the prior.

In order to assess the quality of the sampling, we compare the trajectories to a standard costly Metropolis-Hastings MCMC using a Gaussian with variance 0.2 as the proposal distribution. This algorithm was run for $4 * 10^5$ epochs and the proposal acceptance rate was 0.43. Figure 6 shows samples in the directions of the largest, median and smallest variance of the empirical covariance matrix of the Metropolis-Hastings samples.

8. Conclusions

This paper introduced a non-reversible sampler that can be applied to big datasets by means of subsampling the data in each iteration. At the price of a small, controllable bias, it provides the benefits of (i) high mixing speed associated with non-reversibility, and (ii) continuous sample trajectories, with (iii) minimal hyperparameter tuning required, leading to state of the art performance and making it a convenient alternative to biased, difficult-to-tune MH-based stochastic samplers.

References

Ahn, Sungjin, Korattikara, Anoop, and Welling, Max. Bayesian posterior sampling via stochastic gradient fisher scoring. *ICML*, 2012.

Bardenet, Rémi, Doucet, Arnaud, and Holmes, Chris. Towards scaling up Markov chain Monte Carlo: an adaptive subsampling approach. In *ICML*, pp. 405–413, 2014.

Bardenet, Rémi, Doucet, Arnaud, and Holmes, Chris. On Markov chain Monte Carlo methods for tall data. *arXiv:1505.02827*, 2015.

Bierkens, Joris and Roberts, Gareth. A piecewise deterministic scaling limit of Lifted Metropolis-Hastings in the Curie-Weiss model. *arXiv:1509.00302*, 2015.

Bierkens, Joris, Fearnhead, Paul, and Roberts, Gareth. The Zig-Zag Process and Super-Efficient Sampling for Bayesian Analysis of Big Data. *arXiv preprint arXiv:1607.03188*, 2016.

Bierkens, Joris, Bouchard-Côté, Alexandre, Doucet, Arnaud, Duncan, Andrew B, Fearnhead, Paul, Roberts, Gareth, and Vollmer, Sebastian J. Piecewise Deterministic Markov Processes for Scalable Monte Carlo on Restricted Domains. *arXiv preprint arXiv:1701.04244*, 2017.

Bouchard-Côté, Alexandre, Vollmer, Sebastian J, and Doucet, Arnaud. The Bouncy Particle Sampler: A Non-Reversible Rejection-Free Markov Chain Monte Carlo Method. *arXiv:1510.02451*, 2015.

Chen, Tianqi, Fox, Emily B, and Guestrin, Carlos. Stochastic gradient HMC. In *ICML*, pp. 1683—1691, 2014.

Chopin, Nicolas and Ridgway, James. Leave Pima Indians alone: binary regression as a benchmark for Bayesian computation. *arXiv preprint arXiv:1506.08640*, 2015.

Cox, David R. Some statistical methods connected with series of events. *J. Royal Stat. Soc., Series B (Methodological)*, pp. 129–164, 1955.

Davis, Mark HA. Piecewise-deterministic Markov processes: A general class of non-diffusion stochastic models. *J. Royal Stat. Soc., Series B (Methodological)*, pp. 353–388, 1984.

Ding, Nan, Fang, Youhan, Babbush, Ryan, Chen, Changyou, Skeel, Robert D, and Neven, Hartmut. Bayesian sampling using stochastic gradient thermostats. In *NIPS*, pp. 3203–3211, 2014.

Duchi, John, Hazan, Elad, and Singer, Yoram. Adaptive subgradient methods for online learning and stochastic optimization. *Journal of Machine Learning Research*, 12(Jul):2121–2159, 2011.

Dufour, François, Zhang, Huilong, et al. *Numerical Methods for Simulation and Optimization of Piecewise Deterministic Markov Processes*. John Wiley & Sons, 2015.

Fearnhead, Paul, Bierkens, Joris, Pollock, Murray, and Roberts, Gareth O. Piecewise Deterministic Markov Processes for Continuous-Time Monte Carlo. *arXiv preprint arXiv:1611.07873*, 2016.

Gal, Yarin. Uncertainty in Deep Learning (Cambridge PhD Thesis). 2016.

Giles, Mike, Nagapetyan, Tigran, Szpruch, Lukasz, Vollmer, Sebastian, and Zygalakis, Konstantinos. Multilevel Monte Carlo for Scalable Bayesian Computations. *arXiv preprint arXiv:1609.06144*, 2016.

Grandell, Jan. *Doubly stochastic Poisson processes*. Springer, 1976.

Korattikara, Anoop, Chen, Yutian, and Welling, Max. Austerity in MCMC land: Cutting the Metropolis-Hastings budget. *arXiv:1304.5299*, 2013.

Lewis, Peter A and Shedler, Gerald S. Simulation of non-homogeneous Poisson processes by thinning. *Naval Research Logistics Quarterly*, 26(3):403–413, 1979.

Li, Chunyuan, Chen, Changyou, Fan, Kai, and Carin, Lawrence. High-Order Stochastic Gradient Thermostats for Bayesian Learning of Deep Models. *arXiv preprint arXiv:1512.07662*, 2015.

Li, Chunyuan, Chen, Changyou, Carlson, David, and Carin, Lawrence. Preconditioned stochastic gradient langevin dynamics for deep neural networks. *AAAI*, 2016.

Ma, Yi-An, Chen, Tianqi, and Fox, Emily. A complete recipe for stochastic gradient MCMC. In *NIPS*, pp. 2899–2907, 2015.

Ma, Yi-An, Chen, Tianqi, Wu, Lei, and Fox, Emily B. A Unifying Framework for Devising Efficient and Irreversible MCMC Samplers. *arXiv preprint arXiv:1608.05973*, 2016.

Maclaurin, Dougal and Adams, Ryan P. Firefly Monte Carlo: Exact MCMC with subsets of data. *arXiv:1403.5693*, 2014.

Monmarché, Pierre. Piecewise deterministic simulated annealing. *arXiv preprint arXiv:1410.1656*, 2014.

Neal, Radford M. Improving asymptotic variance of MCMC estimators: Non-reversible chains are better. *arXiv preprint math/0407281*, 2004.

Neal, Radford M. *Bayesian learning for neural networks*, volume 118. Springer Science & Business Media, 2012.

Patterson, Sam and Teh, Yee Whye. Stochastic gradient Riemannian Langevin dynamics on the probability simplex. In *NIPS*, pp. 3102–3110, 2013.

Peters, EAJF and de With, G. Rejection-free Monte Carlo sampling for general potentials. *Phys. Rev. E*, 85(2): 026703, 2012.

Pollock, Murray, Fearnhead, Paul, Johansen, Adam M, and Roberts, Gareth O. The Scalable Langevin Exact Algorithm: Bayesian Inference for Big Data. *arXiv preprint arXiv:1609.03436*, 2016.

Quiroz, Matias, Villani, Mattias, and Kohn, Robert. Speeding up MCMC by efficient data subsampling. *Riksbank Research Paper Series*, (121), 2015.

Quiroz, Matias, Villani, Mattias, and Kohn, Robert. Exact Subsampling MCMC. *arXiv:1603.08232*, 2016.

Robert, Christian and Casella, George. *Monte Carlo statistical methods*. Springer Science & Business Media, 2013.

Roux, Nicolas L, Schmidt, Mark, and Bach, Francis R. A stochastic gradient method with an exponential convergence rate for finite training sets. In *Advances in Neural Information Processing Systems*, pp. 2663–2671, 2012.

Vucelja, Marija. Lifting–A Nonreversible MCMC Algorithm. *arXiv:1412.8762*, 2014.

Welling, Max and Teh, Yee W. Bayesian learning via stochastic gradient Langevin dynamics. In *ICML*, pp. 681–688, 2011.