Piecewise Deterministic Markov Processes for Scalable Monte Carlo on Restricted Domains

Joris Bierkens[a,*], Alexandre Bouchard-Côté[b], Arnaud Doucet[c], Andrew B. Duncan[d], Paul Fearnhead[e], Thibault Lienart[c], Gareth Roberts[f], Sebastian J. Vollmer[f]

[a]*Delft Institute of Applied Mathematics, TU Delft, Netherlands*
[b]*Department of Statistics, University of British Columbia, Canada*
[c]*Department of Statistics, University of Oxford, UK*
[d]*School of Mathematical and Physical Sciences, University of Sussex, UK*
[e]*Department of Mathematics and Statistics, Lancaster University, UK*
[f]*Department of Statistics, University of Warwick, UK*

Abstract

Piecewise Deterministic Monte Carlo algorithms enable simulation from a posterior distribution, whilst only needing to access a sub-sample of data at each iteration. We show how they can be implemented in settings where the parameters live on a restricted domain.

Keywords: MCMC, Bayesian statistics, piecewise deterministic Markov processes, logistic regression

---

1. Introduction

Markov chain Monte Carlo (MCMC) methods have been central to the wide-spread use of Bayesian methods. However their applicability to some modern applications has been limited due to their high computational cost, particularly in big-data, high-dimensional settings. This has led to interest in new MCMC methods, particularly non-reversible methods which can mix better than standard reversible MCMC [8, 22], and variants of MCMC that require accessing only small subsets of the data at each iteration [24].

One of the main technical challenges associated with likelihood-based inference for big data is the fact that likelihood calculation is computationally expensive (typically $O(N)$ for data sets of size N). MCMC methods built from piecewise deterministic Markov processes (PDMPs) offer considerable promise for reducing this $O(N)$ burden, due to their ability to use sub-sampling

*Corresponding author
Email address: `joris.bierkens@tudelft.nl` (Joris Bierkens)

techniques, whilst still being guaranteed to target the true posterior distribution [4, 6, 10, 11, 17]. Furthermore, factor graph decompositions of the target distribution can be leveraged to perform sparse updates of the variables [6, 16, 20].

PDMPs explore the state space according to constant velocity dynamics, but where the velocity changes at random event times. The rate of these event times, and the change in velocity at each event, are chosen so that the position of the resulting process has the posterior distribution as its invariant distribution. We will refer to this family of sampling methods as Piecewise Deterministic Monte Carlo methods (PDMC).

Existing PDMC algorithms can only be used to sample from posteriors where the parameters can take any value in $\mathbb{R}^d$. In this paper (Section 2) we show how to extend PDMC methodology to deal with constraints on the parameters. Such models are ubiquitous in machine learning and statistics. For example, many popular models used for binary, ordinal and polychotomous response data are multivariate real-valued latent variable models where the response is given by a deterministic function of the latent variables [1, 9, 21]. Under the posterior distribution, the domain of the latent variables is then constrained based on the values of the responses. Additional examples arise in regression where prior knowledge restricts the signs of marginal effects of explanatory variables such as in econometrics [12], image processing and spectral analysis [3], [13] and non-negative matrix factorization [14]. A few methods for dealing with restricted domains are available but these either target an approximation of the correct distribution [19] or are limited in scope [18].

2. Piecewise Deterministic Monte Carlo on Restricted Domains

Here we present the general PDMC algorithm in a restricted domain. Specific implementations of PDMC algorithms can be derived as continuous-time limits of familiar discrete-time MCMC algorithms [5, 20], and these derivations convey much of the intuition behind why the algorithms have the correct stationary distribution. Our presentation of these methods is different, and more general. We first define a simple class of PDMPs and show how these can be simulated. We then give simple recipes for how to choose the dynamics of the PDMP so that it will have the correct stationary distribution.

Our objective is to compute expectations with respect to a probability distribution π on

$\mathcal{O} \subseteq \mathbb{R}^d$ which is assumed to have a smooth density, also denoted $\pi(x)$, with respect to the Lebesgue measure on $\mathcal{O}$. With this objective in mind, we will construct a continuous-time Markov process $Z_t = (X_t, V_t)_{t\geq 0}$ taking values in the domain $E = \mathcal{O} \times \mathcal{V}$, where $\mathcal{O}$ and $\mathcal{V}$ are subsets of $\mathbb{R}^d$, such that $\mathcal{O}$ is open, pathwise connected and with Lipschitz boundary $\partial\mathcal{O}$. In particular, if $\mathcal{O} = \mathbb{R}^d$ then $\partial\mathcal{O} = \emptyset$. The dynamics of Z_t are easy to describe if one views X_t as position and V_t as velocity. The position process X_t moves deterministically, with constant velocity V_t between a discrete set of *switching times* which are simulated according to N inhomogeneous Poisson processes, with respective intensity functions $\lambda_i(X_t, V_t)$, $i = 1, \ldots, N$, depending on the current state of the system. At each switching time the position stays the same, but the velocity is updated according to a specified transition kernel. More specifically, suppose the next switching event occurs from the i^{th} Poisson process, then the velocity immediately after the switch is sampled randomly from the probability distribution $Q_i(x, v, \cdot)$ given the current position x and velocity v. The switching times are random, and designed in conjunction with the kernels $(Q_i)_{i=1}^N$ so that the invariant distribution of the process coincides with the target distribution π.

To ensure that X_t remains confined within $\mathcal{O}$ the velocity of the process is updated whenever X_t hits $\partial\mathcal{O}$ so that the process moves back into $\mathcal{O}$. We shall refer to such updates as *reflections* even though they need not be specular reflections.

The resulting stochastic process is a Piecewise Deterministic Markov Process (PDMP, [7]). For it to be useful as the basis of a Piecewise Deterministic Monte Carlo (PDMC) algorithm we need to (i) be able to easily simulate this process; and (ii) have simple recipes for choosing the intensities, $(\lambda_i)_{i=1}^N$, and transition kernels, $(Q_i)_{i=1}^N$, such that the resulting process has $\pi(x)$ as its marginal stationary distribution. We will tackle each of these problems in turn.

2.1. Simulation

The key challenge in simulating our PDMP is simulating the event times. The intensity of events is a function of the state of the process. But as the dynamics between event times are deterministic, we can easily represent the intensity for the next event as a deterministic function of time. Suppose that the PDMP is driven by a single inhomogeneous Poisson process with intensity function

$$\widetilde{\lambda}(u; X_t, V_t) = \lambda(X_t + uV_t, V_t), \quad u \geq 0.$$

We can simulate the first event time directly if we have an explicit expression for the inverse function of the monotonically increasing function

$$u \mapsto \int_0^u \widetilde{\lambda}(s; X_t, V_t)\, ds. \tag{1}$$

In this case the time until the next event is obtained by (i) simulating a realization, y say, of an exponential random variable with rate 1; and (ii) setting the time until the next event as the value τ that solves $\int_0^\tau \widetilde{\lambda}(s; X_t, V_t)\, ds = y$.

Inverting (1) is often not practical. In such cases simulation can be carried out via *thinning* [15]. This requires finding a tractable upper bound on the rate, $\overline{\lambda}(u) \geq \widetilde{\lambda}(u; X_t, V_t)$ for all $u > 0$. Such an upper bound will typically take the form of a piecewise linear function or a step function. Note that the upper bound $\overline{\lambda}$ is only required to be valid along the trajectory $u \mapsto (X_t + uV_t, V_t)$ in $\mathcal{O} \times \mathcal{V}$. Therefore the upper bound may depend on the starting point (X_t, V_t) of the line segment we are currently simulating. We then propose potential events by simulating events from an inhomogenous Poisson process with rate $\overline{\lambda}(u)$, and accept an event at time u with probability $\widetilde{\lambda}(u; X_t, V_t)/\overline{\lambda}(u)$. The time of the first accepted event will be the time until the next event in our PDMP.

To handle boundary reflections, at every given time t, we also keep track of the next reflection event in the absence of a switching event, i.e. we compute

$$\tau_b = \inf \{u > 0 \, : \, X_t + uV_t \notin \mathcal{O}) \, .$$

If the boundary $\partial\mathcal{O}$ can be represented as a finite set of M hyper-planes in $\mathbb{R}^d$, then the cost of computing τ_b is $O(Md)$. When generating the switching event times and positions for Z_t we determine whether a boundary reflection will occur before the next potential switching event. If so, then we induce a switching event at time $t + \tau_b$ where $X_{t+\tau_b} \in \partial\mathcal{O}$ and sample a new velocity from the transition kernel Q_b, i.e. $V_{t+\tau_b} \sim Q_b(X_{t+\tau_b}, V_t, \cdot)$.

Although theoretically we may choose a new velocity pointing outwards and have an immediate second jump, we will for algorithmic purposes assume that the probability measure $Q_b(x, u, \cdot)$ for $(x, u) \in \partial O \times \mathcal{V}$ is concentrated on those directions v for which $(v \cdot n(x)) \leq 0$, where $n(x)$ is the outward normal at $x \in \partial\mathcal{O}$.

For a PDMP driven by N inhomogeneous Poisson processes with intensities $(\lambda_i)_{i=1}^N$ the previous steps lead to the following algorithm for simulating the next event of our PDMP. This algorithm can be iterated to simulate the PDMP for a chosen number of events or a pre-specified time-interval.

(0) **Initialize**: Set t to the current time and (X_t, V_t) to the current position and velocity.

(1) **Determine bound**: For each $i \in 1, \ldots, N$, find a convenient function $\overline{\lambda}_i$ satisfying $\overline{\lambda}_i(u) \geq \widetilde{\lambda}_i(u; X_t, V_t)$ for all $u \geq 0$, depending on the initial point (X_t, V_t) from which we are departing.

(2) **Propose event**: For $i = 1, \ldots, N$ simulate the first event times τ_i' of a Poisson process with rate function $\overline{\lambda}_i$. Compute the next boundary reflection time τ_b.

(3) Let $i_{\min} = \arg\min_{j=1,\ldots,N} \tau_j'$ and $\tau' = \tau_{i_{min}}'$.

(4) **Accept/Reject event:**

(4.1) If $\tau_b < \tau'$ then set $\tau = \tau_b$; set $X_{t+\tau} = X_t + \tau V_t$; sample a new velocity $V_{t+\tau} \sim Q_b(X_{t+\tau}, V_t, \cdot)$.

(4.2) Otherwise with probability

$$\frac{\widetilde{\lambda}_{i_{\min}}(\tau'; X_t, V_t)}{\overline{\lambda}_{i_{\min}}(\tau')}$$

accept the event at time $\tau = \tau'$.

(4.2.1) **Upon acceptance**: set $X_{t+\tau} = X_t + \tau V_t$; sample a new velocity $V_{t+\tau} \sim Q_{i_{\min}}(X_{t+\tau}, V_t, \cdot)$.

(4.2.2) **Upon rejection**: set $X_{t+\tau} = X_t + \tau V_t$ and set $V_{t+\tau} = V_t$.

(5) **Update**: Record the time $t + \tau'$ and state $(X_{t+\tau'}, V_{t+\tau'})$.

2.2. Output of PDMC algorithms

The output of these algorithms will be a sequence of event times $t_1, t_2, t_3, \ldots, t_K$ and associated states (X_1, V_1), $(X_2, V_2), \ldots, (X_K, V_K)$. To obtain the value of the process at times $t \in [t_k, t_{k+1})$, we can linearly interpolate the continuous path of the process between event times, i.e. $X_t = X_{t_k} + V_k(t - t_k)$. Time integrals $\int_0^t f(X_s)\, ds$ of a function f of the process X_t can often be computed analytically from the output of the above algorithm. If not they can be approximated by numerically integrating the one dimensional integral along the piecewise linear trajectory of the PDMP. Alternatively we can sample the PDMP at a set of evenly spaced time points along the trajectory and use this collection as an approximate sample from our target distribution.

Under the assumption that the resulting PDMP is ergodic (for sufficient conditions see e.g. [4, 6]) and that the marginal density on $\mathcal{O}$ of the stationary distribution of (X_t, V_t) is equal to π, we have the following version of the law of large numbers for the PDMP $(X_t, V_t)_{t\geq 0}$: For all $f \in L^2(\pi)$ we have that, with probability one,

$$\int_{\mathbb{R}^d} f(x)\pi(x)\,dx = \lim_{T\to\infty} \frac{1}{T}\int_0^T f(X_s)\,ds.$$

It is this formula which allows us to use PDMPs for Monte Carlo purposes.

2.3. Choosing the intensity and transition kernels

Assume, as most existing PDMC methods do [4, 6, 20], that the target density, $\pi(x) : \mathcal{O} \to (0, \infty)$ is differentiable. Under this condition we can provide criteria on the switching intensities (λ_i) and transition kernels Q_i and Q_b which must hold for a given probability distribution to be a stationary distribution of Z_t. We shall consider stationary distributions for which x and v are independent, i.e. distributions of the form $\pi(x)dx \otimes \rho(dv)$ on E. Furthermore we assume that $\pi(x) \propto \exp(-U(x))$ where U is continuously differentiable.

We impose the condition that

$$\int_{v\in\mathcal{V}} \sum_{i=1}^N \lambda_i(x,v)Q_i(x,v,du)\,\rho(dv) = \int_{v\in\mathcal{V}} \sum_{i=1}^N \lambda_i(x,v)Q_i(x,u,dv)\,\rho(du), \quad x \in \mathcal{O}. \tag{2}$$

A sufficient condition for (2) is that each Q_i is reversible with respect to ρ, i.e. for every $i = 1, \ldots, N$ and $x \in \mathcal{O}$, we have that $Q_i(x, v, du)\rho(dv) = Q_i(x, u, dv)\rho(du)$.

Moreover, we shall require the following condition which relates the probability flow with the switching intensities λ_i:

$$\sum_{i=1}^N \int_{\mathcal{V}} \lambda_i(x,v)Q_i(x,u,dv) - \sum_{i=1}^N \lambda_i(x,u) = -u\cdot\nabla U(x), \quad (x,u) \in \mathcal{O}\times\mathcal{V}. \tag{3}$$

Finally, the boundary transition kernel should satisfy

$$Q_b(x,u,dv)\rho(du) = Q_b(x,v,du)\rho(dv), \quad x \in \partial\mathcal{O}, \tag{4}$$

and

$$\int_{\mathcal{V}} (n(x) \cdot u)\, Q_b(x, v, du) = -v \cdot n(x), \quad (x, v) \in \partial\mathcal{O} \times \mathcal{V}, \tag{5}$$

where for $x \in \partial\mathcal{O}$, we denote by $n(x)$ the outward unit normal of $\partial\mathcal{O}$.

In practice we only have to satisfy (4) and (5) on the exit region $\Gamma \subset \mathcal{O} \times \mathcal{V}$. For example if $\mathcal{O} = (a, b) \subset \mathbb{R}^1$ and $\mathcal{V} = \{-1, +1\}$, then $\Gamma = \{b, +1\} \cup \{a, -1\}$. The specification of Q_b on $(\partial\mathcal{O} \times \mathcal{V}) \setminus \Gamma$ is irrelevant as these points are never reached by Z_t. On this irrelevant set, we may choose Q_b as desired to satisfy (5).

The proof of the following result relies on verifying that $\mathbb{E}_{\pi\otimes\rho}[\mathcal{L}f(X, V)] = 0$ where $\mathcal{L}$ denotes the generator of our PDMP. The proof is deferred to the supplementary material, Section 1.

Proposition 1. *Consider the process Z_t on $\mathcal{O}\times\mathcal{V}$ where $\mathcal{O}$ is an open, pathwise connected subset of $\mathbb{R}^d$ with Lipschitz boundary $\partial\mathcal{O}$. Suppose that conditions* (2),(3), (4) *and* (5) *are satisfied. Then $\pi(x)\, dx \otimes \rho(dv)$ is an invariant distribution for the process Z_t.*

2.4. Example: The Bouncy Particle Sampler

Current PDMC algorithms differ in terms of how the Q_i and λ_i are chosen such that the above equation holds for some simple distribution for the velocity. Here we discuss how the Bouncy Particle Sampler (BPS), introduced in [20] and explored in [6], is an example of the framework introduced here. In the supplementary material, Section 1.1, the Zig-Zag sampler is described as a second example. In the following example δ_x denotes the Dirac-measure centered in x.

The Bouncy Particle Sampler is obtained setting $N = 1$ and $\rho = \mathcal{N}(0, I)$ on $\mathbb{R}^d$ or $\rho = \mathcal{U}(S^{d-1})$, i.e. the uniform distribution on the unit sphere. The single switching rate is chosen to be $\lambda_{\mathrm{BPS}}(x, v) = \max(v \cdot \nabla U(x), 0)$, with corresponding switching kernel Q which reflects v with respect to the orthogonal complement of ∇U with probability 1:

$$Q(x, v, dv') = \delta_{(I-2P_{\nabla U})v}(dv'),$$

where $P_y : z \mapsto \frac{z\cdot y}{\|y\|^2} y$ denotes orthogonal projection along the one dimensional subspace spanned by y.

As noted in [6] this algorithm suffers from reducibility issues. These can be overcome by refreshing the velocity by drawing a new velocity independently from $\rho(dv)$. In the simplest case the refreshment times come from an independent Poisson process with constant rate λ_{ref}. This also fits in the framework above by choosing $\widetilde{\lambda} = \lambda_{\text{BPS}} + \lambda_{\text{ref}}$ and

$$Q(x, u, dv) = \frac{\lambda_{\text{BPS}}}{\lambda_{\text{BPS}} + \lambda_{\text{ref}}} \delta_{(I-2P_{\nabla U})u}(dv) + \frac{\lambda_{\text{ref}}}{\lambda_{\text{BPS}} + \lambda_{\text{ref}}} \rho(dv).$$

As boundary transition kernel it is natural to choose

$$Q_b(s, v, du) = \delta_{(I-2P_{n(s)})v}(du),$$

for $s \in \partial\mathcal{O}$, so that the process X_t reflects specularly at the boundary (i.e. angle of incidence equals angle of reflection of process with respect to the boundary normal). It is straightforward to check that condition (2) holds at the boundary and that (5) is satisfied.

As a generalization of the BPS, one can consider a *preconditioned* version, which is obtained by introducing a constant positive definite symmetric matrix M to rescale the velocity process. The choice of M plays a very similar role to the mass matrix in HMC, and careful tuning can give rise to dramatic increases in performance [11, 17].

3. Subsampling

When using PDMC to sample from a posterior, we can use sub-samples of data at each iteration of the algorithm, as described in [4, 6], which reduces the computational complexity of the algorithm from $O(N)$ to $O(1)$, where N is the size of the data, without affecting the theoretical validity of the algorithm. In the following we will assume that we can write the posterior as $\pi(x) \propto \prod_{i=1}^{N} f(y_i; x)$, for some function f. For example this would be the likelihood for a single IID data point times the $1/N$th power of the prior.

The idea of using sub-sampling, within say the Bouncy Particle Sampler (BPS), is that at each iteration of our PDMC algorithm we can replace $\nabla U(x)$ by an unbiased estimator in step (3). We need to use the same estimate both when calculating the actual event rate in the accept/reject step and, if we accept, when simulating the new velocity. The only further alteration we need to the algorithm is to choose an upper bound $\overline{\lambda}$ that holds for all realizations of $\widehat{\nabla U}$. A more comprehensive explanation of this argument can be found in [4, 10] in the context of the Zig-Zag sampler, and in [6, 11] for the bouncy particle sampler.

We first present a way for estimating ∇U unbiasedly using control variates [2, 4]. For any $x, \hat{x} \in \mathcal{O}$ we note that $\nabla U(x) = \nabla U(\hat{x}) + [\nabla U(x) - \nabla U(\hat{x})]$. We can then introduce the estimator

$\widehat{\nabla U}(x)$ of $\nabla U(x)$ by

$$\widehat{\nabla U}(x) = \nabla U(\hat{x}) + N\left[\nabla \log f(y_I; x) - \nabla \log f(y_I; \hat{x})\right], \tag{6}$$

where I is drawn uniformly from $\{1, \ldots, N\}$.

It is straightforward to show that the resulting BPS algorithm uses an event rate that is $\mathbb{E}\left[\max\left(0, (\widehat{\nabla U}(x) \cdot v)\right)\right]$, and that this rate and the resulting transition probability Q at events satisfies Proposition 1. Hence this algorithm still targets $\pi(x)$, but only requires access to one data point at each accept-reject decision.

Note that this gain in computational efficiency does not come for free, as it follows from Jensen's inequality that the overall rate of events will be higher. This makes mixing of the PDMC process slower. It is also immediate that the bound, $\overline{\lambda}$, we will have to use will be higher. However [4] show that if our estimator of $\widehat{\nabla U}(x)$ has sufficiently small variance, then we can still gain substantially in terms of efficiency. In particular they give an example where the CPU cost effective sample size does not grow with N – by comparison all standard MCMC algorithms would have a cost that is at least linear in N.

To obtain such a low-variance estimator requires a good choice of $\hat{x}$, so that with high probability x will be closer to $\hat{x}$. This involves a preprocessing step to find a value $\hat{x}$ close to the posterior mode, a preprocessing step to then calculate $\nabla U(\hat{x})$ is also needed.

We now illustrate how to find an upper bound on the event rate. Following [4], if we assume L is a uniform (in space and i) upper bound on the largest eigenvalue of the Hessian of U^i, and if $\|v\| = 1$:

$$\begin{aligned}
&\max\left(0, \left(\nabla U(\hat{x}) + N(\nabla U^i(X_t) - \nabla U^i(\hat{x}))\right) \cdot v\right) \\
&\leq \max\left(0, \nabla U(\hat{x}) \cdot v\right) + N\left\|\nabla U^i(x) - \nabla U^i(\hat{x}) \nabla U^i(x) - \nabla^i U(X_t)\right\| \\
&\leq \max\left(0, \nabla U(\hat{x}) \cdot v\right) + NL\left\|x - \hat{x}\right\| + NLt
\end{aligned} \tag{7}$$

Thus the upper bound on the intensity is of the form $\bar{\lambda}(\tau) = a + b \cdot \tau$ with $a, b \geq 0$. In this case the first arrival time can be simulated as follows

$$\tau' = -a/b + \sqrt{\left(\frac{a}{b}\right)^2 + 2 \cdot \frac{R}{b}} \text{ with } R \sim \mathrm{Exp}(1). \tag{8}$$

An alternative and complementary approach to improve the efficiency of this subsampling procedure is to use an estimator of the gradient (3) where I is drawn according to a distribution dependent on the observations [6, 11].

4. Software and Numerical Experiments

A open-source Julia package `PDMP.jl` has been developed to provide efficient implementations of various recently developed piecewise deterministic Monte Carlo methods for sampling in (possibly restricted) continuous spaces. A variety of algorithms are implemented including the Zig-Zag sampler and the Bouncy Particle Sampler with full and local refreshment along with control variate based sub-sampling for these methods. The package has been specifically designed with extensibility in mind, permitting rapid implementation of new PDMP based methods. The library along with code and documentation is available at `github.com/alan-turing-institute/PDSampler.jl`.

We use Bayesian binary logistic regression as a testbed for our newly proposed methodology and perform a simulation study. The data $y_i \in \{-1, 1\}$ is modelled by

$$p(y_i|\xi_i, x) = f(y_i x^T \xi_i) \tag{9}$$

where $\xi \in \mathbb{R}^{p\times n}$ are fixed covariates and $f(z) = \frac{1}{1+\exp(-z)} \in [0, 1]$. We will assume that we wish to fit this model under some monotonicity constraints – so that the probability of $y = 1$ is known to either increase or decrease with certain covariates. This is modeled through the constraint $x_i > 0$ and $x_i < 0$ respectively. An example where such restrictions occur naturally is in logistic regression for questionnaires, see [23]. In following we consider the case $x_j \geq 0$ for $j = 1, \ldots, p$.

For simplicity we use a flat prior over the space of parameters values consistent with our constraints. By Bayes' rule the posterior π satisfies

$$\pi(x) \propto \prod_{i=1}^{N} f(y_i x^T \xi_i) \text{ for } x \in \mathcal{O},$$

where $\mathcal{O}$ is the space of parameter values consistent with our constraints. We implement the BPS with subsampling. As explained in the introduction, subsampling is a key benefit of using piecewise deterministic sampling methods. An informal overview of this idea and its application is located in Section 3. We use reflection at the boundary i.e. $Q_b(s, v, du) = \delta_{(I-2P_{n(s)})v}(du)$ for

$s \in \partial\mathcal{O}$. We can bound the switching intensity by a linear function of time, even when we use the subsampling estimator for the switching rate. See the supplementary material, Section 2, for details on the application of subsampling in this example. We use $n = 10000$ and $p = 20$ and generate artificial data based on $x^\star$ and ξ whose components are a realization i.i.d. uniformly distributed random variables on $[0, 1]$.

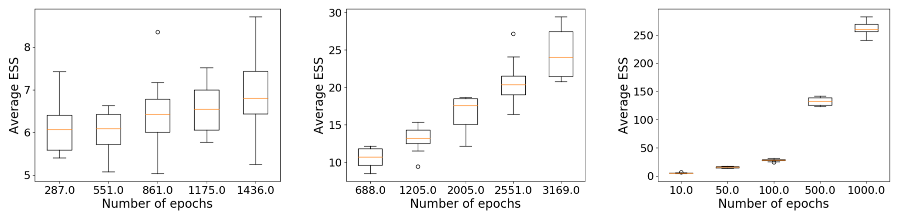
We compare the performance of BPS to standard MALA and HMC schemes, in terms of effective sample size (ESS) per epoch of data evaluation. More specifically, for each scheme we generate 5 chains of increasing length and obtain the distribution of effective sample size (averaged over the p covariates) for 10 independent realisations of each chain. For HMC we use 5 leap-frog steps. We find that we must tune both HMC and MALA to have a small step size due to proposals being rejected at the boundary.

In Figure 1 we plot for each scheme the distribution of ESS as a function of the number of times the dataset $\{y_i\}_{i=1}^n$ was evaluated (epoch). Due to the significant variance in performance between schemes we change the range of the axes in each plot. It is interesting to note that MALA appears to outperform HMC in this example. Indeed, due to increasingly many rejections at the boundary, the effective sample size of the HMC scheme is relatively unchanged over increasing epochs. On the other hand, we observe that the BPS shows a clear advantage over the other samplers, with the effective sample size orders of magnitude higher and increasing rapidly with epoch. Note that there exists a version of HMC which can sample from truncated Gaussian distributions [18] but to our knowledge no efficient HMC scheme is available for general restricted domains.

The Bouncy Particle Sampler for this model was implemented using `PDSampler.jl` while the corresponding HMC and MALA samplers implemented with `Klara.jl`. The code for this numerical experiment along with results are carefully presented in `github.com/tlienart/ConstrainedPDMP/`.

5. Discussion

This work provides a framework for describing a general class of PDMC methods which are ergodic with respect to a given target probability distribution. Open questions remain on how the choice of intensity function, velocity transition kernel as well as other parameters of the system influence the overall performance of the scheme. The problem of understanding the true



(a) Average ESS for HMC (b) Average ESS for MALA (c) Average ESS for PDMP

Figure 1: Average ESS versus number of epochs of data evaluation for MALA, HMC and PDMP (BPS) applied to logistic regression with $p = 20$ and $n = 10000$ and parameter x constrained to be positive. The graphic is based on 10 independent runs for each HMC, MALA and BPS for each choice of number of epochs.

computational cost of such PDMC schemes is more subtle than for classical discrete time MCMC schemes: often one needs to find a balance between fast mixing of the continuous time Markov process and having a switching rate that is relatively cheap to simulate. For example, when using subsampling the mixing of the Markov process is slower than without subsampling, but the computational cost per simulated switch is significantly smaller. Further investigation is required to understand this delicate balance.

Acknowledgements

All authors thank the Alan Turing Institute and Lloyds registry foundation for support. S.J.V. gratefully acknowledges funding through EPSRC EP/N000188/1. J.B., P.F. and G.R. gratefully acknowledge EPSRC ilike grant EP/K014463/1. A.B.D. acknowledges grant EP/L020564/1 and A.D. acknowledges grant EP/K000276/1.

References

[1] J. H. Albert and S. Chib. Bayesian analysis of binary and polychotomous response data. *Journal of the American Statistical Association*, 88(422):669–679, 1993.

[2] R. Bardenet, A. Doucet, and C. Holmes. On Markov chain Monte Carlo methods for tall data. *Journal of Machine Learning Research*, 18(18):1–43, 2017.

[3] S. Bellavia, M. Macconi, and B. Morini. An interior point Newton-like method for non-negative least-squares problems with degenerate solution. *Numerical Linear Algebra with Applications*, 13(10):825–846, 2006.

[4] J. Bierkens, P. Fearnhead, and G. Roberts. The Zig-Zag process and super-efficient sampling for Bayesian analysis of big data. *arXiv preprint arXiv:1607.03188*, 2016.

[5] J. Bierkens and G. Roberts. A piecewise deterministic scaling limit of lifted Metropolis-Hastings in the Curie- Weiss model. *Annals of Applied Probability*, 27(2):846–882, 2017.

[6] A. Bouchard-Côté, S. J. Vollmer, and A. Doucet. The bouncy particle sampler: A non-reversible rejection-free Markov chain Monte Carlo method. *arXiv preprint arXiv:1510.02451*, 2015.

[7] M. H. A. Davis. Piecewise-deterministic Markov processes: A general class of non-diffusion stochastic models. *Journal of the Royal Statistical Society. Series B (Methodological)*, pages 353–388, 1984.

[8] P. Diaconis, S. Holmes, and R. Neal. Analysis of a nonreversible Markov chain sampler. *Annals of Applied Probability*, 10(3):726–752, 2000.

[9] L. Fahrmeir and G. Tutz. *Multivariate Statistical Modelling based on Generalized Linear Models*. Springer, New York, 2001.

[10] P. Fearnhead, J. Bierkens, M. Pollock, and G. O. Roberts. Piecewise deterministic Markov processes for continuous-time Monte Carlo. *arXiv preprint arXiv:1611.07873*, 2016.

[11] N. Galbraith. On Event-Chain Monte Carlo Methods. Master's thesis, Department of Statistics, Oxford University, 2016.

[12] J. Geweke. Exact inference in the inequality constrained normal linear regression model. *Journal of Applied Econometrics*, 1(2):127–141, 1986.

[13] Y. Guo and M. Berman. A comparison between subset selection and l1 regularisation with an application in spectroscopy. *Chemometrics and Intelligent Laboratory Systems*, 118:127–138, 2012.

[14] H. Kim and H. Park. Sparse non-negative matrix factorizations via alternating non-negativity-constrained least squares for microarray data analysis. *Bioinformatics*, 23(12):1495–1502, 2007.

[15] P. A. Lewis and G. S. Shedler. Simulation of nonhomogeneous Poisson processes by thinning. *Naval Research Logistics Quarterly*, 26(3):403–413, 1979.

[16] Y. Nishikawa, M. Michel, W. Krauth, and K. Hukushima. Event-chain algorithm for the Heisenberg model. *Physical Review E*, 92(6):63306, 2015.

[17] A. Pakman, D. Gilboa, D. Carlson, and L. Paninski. Stochastic bouncy particle sampler. *arXiv preprint arXiv:1609.00770*, 2016.

[18] A. Pakman and L. Paninski. Exact Hamiltonian Monte Carlo for truncated multivariate Gaussians. *Journal of Computational and Graphical Statistics*, 23(2):518–542, 2014.

[19] S. Patterson and Y. W. Teh. Stochastic gradient Riemannian Langevin dynamics on the probability simplex. In *Advances in Neural Information Processing Systems*, pages 3102–3110, 2013.

[20] E. A. J. F. Peters and G. De With. Rejection-free Monte Carlo sampling for general potentials. *Physical Review E*, 85(2):1–5, 2012.

[21] L. E. Train. *Discrete Choice Methods with Simulation.* Cambridge university press, 2009.

[22] K. S. Turitsyn, M. Chertkov, and M. Vucelja. Irreversible Monte Carlo algorithms for efficient sampling. *Physica D: Nonlinear Phenomena*, 240(4-5):410–414, 2011.

[23] G. Tutz and J. Gertheiss. Rating scales as predictors—the old question of scale level and some answers. *Psychometrika*, 79(3):357–376, 2014.

[24] M. Welling and Y. W. Teh. Bayesian learning via stochastic gradient Langevin dynamics. In *Proceedings of the 28th International Conference on Machine Learning (ICML-11)*, pages 681–688, 2011.