Efficient Bernoulli factory MCMC for intractable posteriors

Dootika Vats
Department of Mathematics and Statistics
Indian Institute of Technology Kanpur
dootika@iitk.ac.in

Flávio B. Gonçalves
Department of Statistics
Universidade Federal de Minas Gerais
fbgoncalves@est.ufmg.br

Krzysztof Łatuszyński
Department of Statistics
University of Warwick
K.G.Latuszynski@warwick.ac.uk

Gareth O. Roberts
Department of Statistics
University of Warwick
Gareth.O.Roberts@warwick.ac.uk

September 18, 2020

Abstract

Accept-reject based Markov chain Monte Carlo (MCMC) algorithms have traditionally utilised acceptance functions that can be explicitly written as a function of the ratio of the target density at the two contested points. This feature is rendered almost useless in Bayesian posteriors with unknown functional forms. We introduce a new family of MCMC acceptance probabilities that has the distinguishing feature of not being a function of the ratio of the target density at the two points. We present two efficient and stable Bernoulli factories that generate events within this class of acceptance probabilities. The resulting *portkey Barker's* algorithms are exact and computationally more efficient that the current state-of-the-art.

1 Introduction

Markov chain Monte Carlo (MCMC) is a popular tool for drawing samples from complicated distributions. For Bayesian posteriors, where MCMC is most often used, the target density is usually available only up to a proportionality constant. That is, the target density, $\pi(x)$ is such that, $\pi(x) \propto \pi'(x)$, where the functional form of π' is usually known. The popular MCMC algorithm of Metropolis et al. (1953) and Hastings (1970) has been immensely useful since unknown normalising constants of Bayesian posteriors play no role in the sampling process. Given a proposal density $q(x, y)$, a move from x to y in a Metropolis-Hastings (MH) algorithm is accepted with probability

$$\alpha_{\text{MH}}(x, y) = \min\left\{1, \frac{\pi(y)q(y, x)}{\pi(x)q(x, y)}\right\} = \min\left\{1, \frac{\pi'(y)q(y, x)}{\pi'(x)q(x, y)}\right\}.$$

On the other hand, there are many other acceptance probability functions that can be used to ensure the stationarity of π. For instance, Barker (1965) proposed a similar algorithm with

$$\alpha_{\text{B}}(x, y) = \frac{\pi(y)q(y, x)}{\pi(x)q(x, y) + \pi(y)q(y, x)} = \frac{\pi'(y)q(y, x)}{\pi'(x)q(x, y) + \pi'(y)q(y, x)}. \tag{1}$$

Barker's algorithm is used less frequently than MH due to the result of Peskun (1973), which establishes that MH is always better than Barker's algorithm, in terms of asymptotic variance of ergodic averages. However, as shown in Łatuszyński and Roberts (2013), the variance of ergodic averages from Barker's method are no worse than twice that of MH. For this reason, when the MH algorithm is difficult to implement, Barker's algorithm can be particularly important.

Gonçalves et al. (2017a,b) used Barker's acceptance probability for Bayesian posteriors with unknown functional forms. Such intractable posteriors can be due to intractable priors on constrained parameter spaces or intractable likelihoods for complex systems which cannot be conveniently modeled tractably. In either case, the Bayesian posterior is no longer available up to a normalising constant and $\pi(y)/\pi(x)$ cannot be evaluated making it difficult to implement both Barker's and MH algorithms. There are various solutions to this proposed in the literature; from inexact algorithms like the double Metropolis-Hastings (Liang et al., 2007) to exact ones like pseudo-marginal MCMC (Andrieu and Roberts, 2009). See Park and Haran (2018) for a comprehensive review.

Nearly all methods modify the Markovian dynamics of an underlying accept-reject based MCMC algorithm. We, instead, focus on the use of Bernoulli factories to avoid explicitly calculating the acceptance probabilities, as in Gonçalves et al. (2017a,b). Their *two-coin* Bernoulli factory generates events of probability $\alpha_{\mathrm{B}}(x, y)$ without explicitly evaluating $\alpha_{\mathrm{B}}(x, y)$. Thus, an algorithm which evaluates $\alpha_{\mathrm{B}}(x, y)$ explicitly, and one which embeds the Bernoulli factory within each iteration to determine whether to accept a proposed move, are statistically indistinguishable. However, as we will demonstrate, the computational effort required for the two-coin Bernoulli factory can be quite large.

This provides strong motivation to consider other acceptance probabilities $\alpha(x, y)$ that are stationary for π and for which efficient Bernoulli factories can be constructed for implementing the accept-reject step. To date, it has been customary to consider functions $\alpha(x, y)$ which can be written explicitly in terms of $\pi(y)/\pi(x)$ so that any unknown normalisation constant cancels (see Billera and Diaconis (2001); Peskun (1973); Tierney (1994)). However, when using Bernoulli factories we have far more flexibility. Keeping this in mind we propose an acceptance probability where for $d(x, y) \geq 0$ such that $d(x, y) = d(y, x)$, a move from x to y is accepted with probability

$$\alpha(x, y) = \frac{\pi(y)q(y, x)}{\pi(x)q(x, y) + \pi(y)q(y, x) + d(x, y)} \, .$$

We present two choices of $d(x, y)$ for which efficient Bernoulli factories can be constructed to generate events of probability $\alpha(x, y)$. Naturally, $\alpha(x, y) \leq \alpha_{\mathrm{B}}(x, y)$, so the variance ordering of Peskun (1973) applies. However, usually $d(x, y)$ will be small, so as to yield little decrease in efficiency. As we will demonstrate in our examples, any loss in statistical efficiency is made up for in computational efficiency.

The rest of the paper is organised as follows. Section 2 presents the two-coin algorithm for the Barker's acceptance probability. We describe the scenarios in which this algorithm is computationally burdensome. In Sections 3 and 4 we present two new Bernoulli factories, and show that the resulting Markov chain is π-reversible. After a simple illustrative example, Section 5 describes two very different directions for applications of the portkey Barker's method, both stemming from important problems in Bayesian statistics. The first area involves MCMC on constrained spaces where prior normalisation constants are unknown. As an example of this, we give the first exact MCMC method for a well-known Bayesian correlation estimation model. The second class of applications concerns the Bayesian inference for diffusions, which we illustrate on the Wright-Fisher diffusion model. For all examples, our proposed algorithm leads to a far stable sampling process, with significant computational gains.

2 Barker's method and the two-coin algorithm

Given a current state of the Markov chain x, and a proposal density $q(x, y)$, recall the Barker's acceptance probability in (1). Algorithm 1 presents the Barker's update for obtaining a realization at time $m + 1$. Usually, Step 2 is implemented by drawing $U \sim U[0, 1]$ and checking if $U \leq \alpha_{\mathrm{B}}(x_m, y)$, however, this is not possible when $\alpha_{\mathrm{B}}(x_m, y)$ cannot be evaluated. Gonçalves et al. (2017a,b) noticed that a Bernoulli factory can be constructed to obtain events of probability $\alpha_{\mathrm{B}}(x_m, y)$ without explicitly evaluating it.

Algorithm 1 Barker's MCMC for x_{m+1}

1: Draw $y \sim q(x_m, dy)$
2: Draw $A \sim \mathrm{Bern}\left(\alpha_{\mathrm{B}}(x_m, y)\right)$
3: **if** $A = 1$ **then**
4: $\quad x_{m+1} = y$
5: **if** $A = 0$ **then**
6: $\quad x_{m+1} = x_m$

The Bernoulli factory problem is one in which given events that occur with probability p, the goal is to simulate an event with probability $h(p)$, for some function of interest, h (Asmussen et al., 1992; Huber, 2017; Keane and O'Brien, 1994; Łatuszyński et al., 2011; Morina et al., 2019; Nacu and Peres, 2005). Gonçalves et al. (2017a,b) proposed the following Bernoulli factory to sample events with probability $\alpha_{\mathrm{B}}(x, y)$. Suppose,

$$\pi(x)q(x, y) = c_x p_x \,,$$

where c_x is possibly known and $0 < p_x < 1$. Similarly, $\pi(y)q(y, x) = c_y p_y$. The roles of c_x and c_y are to ensure that p_x and p_y are valid probabilities. We stress that the bound c_x can be a local bound and a global bound over the full support is not required, thus allowing for tighter bounds. For asymmetric proposal distributions, c_x and p_x depend on both x on y, however we suppress the dependency on y for notational convenience.

One way to arrive at c_x and p_x is to find c_x such that

$$\pi(x)q(x, y) \leq c_x \text{ and then set } p_x = \frac{\pi(x)q(x, y)}{c_x}, \tag{2}$$

with analogous statements for c_y and p_y. The two-coin Bernoulli factory of Gonçalves et al. (2017a,b) presented in Algorithm 2, returns events of probability $\alpha_{\mathrm{B}}(x, y)$ with

$$h(p_x, p_y) := \frac{c_y p_y}{c_x p_x + c_y p_y} = \alpha_{\mathrm{B}}(x, y).$$

First, both c_x and c_y can be known up to a common normalising constant. Second, it

Algorithm 2 two-coin algorithm for $\alpha_{\mathrm{B}}(x, y)$

1: Draw $C_1 \sim \mathrm{Bern}\left(\dfrac{c_y}{c_y + c_x}\right)$
2: **if** $C_1 = 1$ **then**
3:     Draw $C_2 \sim \mathrm{Bern}(p_y)$
4:     **if** $C_2 = 1$ **then**
5:         output 1
6:     **if** $C_2 = 0$ **then**
7:         go to Step 1
8: **if** $C_1 = 0$ **then**
9:     Draw $C_2 \sim \mathrm{Bern}(p_x)$
10:     **if** $C_2 = 1$ **then**
11:         output 0
12:     **if** $C_2 = 0$ **then**
13:         go to Step 1

is assumed in Steps 3 and 9 that events of probabilities p_y and p_x, respectively, can be simulated. Using laws of conditional probability, it is easy to check that Algorithm 2 returns 1 with probability $\alpha_{\mathrm{B}}(x, y)$. In addition, as Gonçalves et al. (2017a) describe, the number of loops until the algorithm stops is distributed as a $\mathrm{Geom}((c_y p_y + c_x p_x)/(c_y + c_x))$, and the mean execution time is

$$\frac{c_x + c_y}{c_x p_x + c_y p_y} = \frac{c_x + c_y}{\pi(x)q(x, y) + \pi(y)q(y, x)} .$$

Clearly, the computational efficiency of the two-coin algorithm relies heavily on the upper bounds c_x and c_y. If the bound is loose, then the algorithm yields a large mean execution time.

3 Portkey Barker's method

The main source of inefficiency in implementing Barker's method via Bernoulli factories is the inefficiency of the two-coin algorithm. Motivated by this, we introduce a new family of acceptance probabilities and provide an efficient Bernoulli factory for members of this family.

For a proposal density $q(x, y)$, consider accepting a proposed value y with probability,

$$\alpha(x, y) = \frac{\pi(y)q(y, x)}{\pi(x)q(x, y) + \pi(y)q(y, x) + d(x, y)} , \tag{3}$$

where $d(x, y) = d(y, x) \geq 0$. Then $\alpha(x, y)$ yields a π-reversible Markov chain and the symmetry of $d(x, y)$ is essential to this.

Theorem 1. *For a proposal density $q(x, y)$, consider a Markov chain with acceptance probability $\alpha(x, y)$ such that for some function $d(x, y)$*

$$\alpha(x, y) = \frac{\pi(y)q(y, x)}{\pi(x)q(x, y) + \pi(y)q(y, x) + d(x, y)}\,.$$

Then the Markov chain is π-reversible if and only if $d(x, y) = d(y, x)$.

Proof. An acceptance function yields a π-reversible Markov chain if and only if

$$\pi(y)q(y, x)\alpha(y, x) = \pi(x)q(x, y)\alpha(x, y)\,.$$

Let $d(x, y) = d(y, x)$. Consider,

$$\pi(x)q(x, y)\alpha(x, y) = \frac{\pi(y)q(y, x)\,\pi(x)q(x, y)}{\pi(x)q(x, y) + \pi(y)q(y, x) + d(x, y)} = \pi(y)q(y, x)\alpha(y, x)\,.$$

□

As mentioned in the introduction, $\alpha(x, y) \leq \alpha_{\mathrm{B}}(x, y)$ and by Peskun's ordering, Barker's method is more efficient. However, for a particular choice of $d(x, y)$ we present a Bernoulli factory that provides significant computational gains, enough to supersede the loss of statistical efficiency. For a user-chosen $0 < \beta \leq 1$, consider the following *portkey Barker's* acceptance probability:

$$\alpha_{(\beta)}(x, y) := \frac{\pi(y)q(y, x)}{\pi(y)q(y, x) + \pi(x)q(x, y) + \frac{(1-\beta)}{\beta}(c_x + c_y)}\,, \tag{4}$$

with c_x and c_y given by (2). To ensure $d(x, y)$ is small, $\beta \approx 1$.

To yield events of probability $\alpha_{(\beta)}(x, y)$, we modify the two-coin algorithm via, what we call a *portkey*[1] method. Our modified *portkey two-coin* algorithm in Algorithm 3 introduces a first step in the two-coin algorithm that allows immediate rejections with probability $1-\beta$. For a given proposal, whenever Algorithm 3 loops on Steps 9 and 17, an S is drawn, which if zero, rejects the proposed value immediately. Running Algorithm 3 with $\beta \approx 1$ avoids the large number of loops often witnessed in Algorithm 2.

Theorem 2. *Algorithm 3 yields output 1 with probability $\alpha_{(\beta)}(x, y)$.*

[1]We borrow the word *portkey* from the Harry Potter books by J.K. Rowling. As described on *Pottermore.com*, "The name portkey comes from the French porter – to carry – and the word key, in the sense of secret or trick".

Algorithm 3 Portkey two-coin algorithm

```
1:  Draw S ~ Bernoulli(β)
2:  if S = 0 then
3:      output 0
4:  if S = 1 then
5:      Draw C_1 ~ Bern( c_y / (c_x + c_y) )
6:      if C_1 = 1 then
7:          Draw C_2 ~ Bern(p_y)
8:          if C_2 = 1 then
9:              output 1
10:         if C_2 = 0 then
11:             go to Step 1
12:     if C_1 = 0 then
13:         Draw C_2 ~ Bern(p_x)
14:         if C_2 = 1 then
15:             output 0
16:         if C_2 = 0 then
17:             go to Step 1
```

Proof. Let r be the probability of no output in any given loop of the algorithm. Then,

$$r = \beta \frac{c_y(1-p_y) + c_x(1-p_x)}{c_x + c_y}$$

$$\Rightarrow \sum_{i=0}^{\infty} r^i = \frac{c_x + c_y}{(1-\beta)(c_x + c_y) + \beta(c_x p_x + c_y p_y)}.$$

For the algorithm to output 1, for all i, there should be no output in all loops up to i, and the ith loop should output 1. Thus, the probability that the algorithm outputs 1 is

$$\beta \frac{c_y p_y}{c_x + c_y} \sum_{i=0}^{\infty} r^i = \frac{c_y p_y}{c_x p_x + c_y p_y + \frac{1-\beta}{\beta}(c_x + c_y)}.$$

$\square$

From Theorem 1, the portkey Barker's algorithm is π-reversible, and if $\beta > 0$, it is π-ergodic since $\alpha_{(\beta)}(x, y) > 0$. It may be intuitive to see that Algorithm 3 will quite obviously lead to a smaller mean execution time. More specifically, the number of loops until the algorithm stops is distributed according to a Geom(s_β), where

$$s_\beta = (1-\beta) + \beta \cdot \frac{c_y p_y + c_x p_x}{c_x + c_y}.$$

We note that for $0 < \beta < 1$, $s_\beta > 1 - \beta$, which is bounded away from zero. Thus, the mean execution time is bounded above. A similar argument cannot be made for the original two-coin algorithm. Specifically, the ratio of the mean execution time of the two-coin algorithm to the portkey two-coin algorithm is

$$\frac{1/s_1}{1/s_\beta} = \frac{s_\beta}{s_1} = (1-\beta) \cdot \left(\frac{c_y p_y + c_x p_x}{c_x + c_y}\right)^{-1} + \beta \ \geq 1\,.$$

Thus, if $(c_y p_y + c_x p_x)/(c_x + c_y) \approx 0$, i.e., the original two-coin algorithm is highly inefficient, the ratio diverges to infinity. On the other hand, if $(c_y p_y + c_x p_x)/(c_x + c_y) \approx 1$, i.e., the original two-coin algorithm is efficient, the two algorithms have comparable expected number of loops.

Computational efficiency gained here is at the cost of statistical efficiency. Let $P_{(\beta)}$ and P_{B} denote the Markov operators for portkey Barker's and Barker's algorithms. In addition, for a function g, let $\bar{g}_n$ denote the Monte Carlo estimator of $\int g\pi(dx)$ obtained using a Markov kernel P and denote $\mathrm{var}(g, P) := \lim_{n\to\infty} n\mathrm{Var}_\pi(\bar{g}_n)$.

Theorem 3. *For $0 < \beta \leq 1$, $\alpha_{(\beta)}(x,y) \leq \beta\, \alpha_B(x,y)$. As a consequence,*

$$var(g, P_B) \leq \beta\, var(g, P_{(\beta)}) + (\beta - 1)\, Var_\pi(g)\,.$$

Proof. Since $c_x + c_y \geq \pi(x)q(x,y) + \pi(y)q(y,x)$

$$\begin{aligned}
\alpha_{(\beta)}(x,y) &= \frac{\pi(y)q(y,x)}{\pi(x)q(x,y) + \pi(y)q(y,x) + \dfrac{(1-\beta)}{\beta}(c_x + c_y)} \\
&\leq \beta \cdot \frac{\pi(y)q(y,x)}{\beta\pi(x)q(x,y) + \beta\pi(y)q(y,x) + (1-\beta)(\pi(x)q(x,y) + \pi(y)q(y,x))} \\
&= \beta \cdot \alpha_{\mathrm{B}}(x,y)\,.
\end{aligned}$$

Combinig Łatuszyński and Roberts (2013, Corollary 1) and the ordering of Peskun (1973), or equivalently Zanella (2020, Theorem 2), yields,

$$\mathrm{var}(g, P_{\mathrm{B}}) \leq \beta \mathrm{var}(g, P_{(\beta)}) + (\beta - 1)\mathrm{Var}_\pi(g)\,.$$

□

A variance bound in the opposite direction can be obtained under specific conditions.

Theorem 4. *For $0 < \beta \leq 1$, if there exists $\delta > 0$ such that $p_x > \delta$ and $p_y > \delta$, then*

$$\alpha_B(x,y) \leq \left(1 + \frac{1-\beta}{\delta\beta}\right) \cdot \alpha_{(\beta)}(x,y)\,.$$

As a consequence,

$$var\left(g, P_{(\beta)}\right) \leq \left(1 + \frac{1-\beta}{\delta\beta}\right) var(g, P_B) + \frac{1-\beta}{\delta\beta} Var_\pi(g)\,.$$

Proof. Since $p_x \geq \delta$ and $p_y \geq \delta$, $c_x + c_y \leq (\pi(x)q(x,y) + \pi(y)q(y,x))/\delta$. So,

$$\begin{aligned}\alpha_{(\beta)}(x,y) &\geq \frac{\pi(y)q(y,x)}{\pi(x)q(x,y) + \pi(y)q(y,x) + \dfrac{1-\beta}{\delta\beta}(\pi(x)q(x,y) + \pi(y)q(y,x))} \\ &= \left(1 + \frac{1-\beta}{\delta\beta}\right)^{-1} \alpha_{\mathrm{B}}(x,y)\,.\end{aligned}$$

The variance ordering follows from Łatuszyński and Roberts (2013, Corollary 1). ☐

When p_x or p_y is small, it is desirable to set β to be large. Theorem 4 suggests that setting $\beta = 1 - \delta$ is desirable. However, δ will typically not be available.

4 Flipped portkey two-coin algorithm

A significant challenge in implementing the original two-coin algorithm is identifying a suitable c_x. In certain problems, it may be easier to lower bound $\pi(x)q(x,y)$, or in other words, upper bound $\pi(x)^{-1}q(x,y)^{-1}$. That is, suppose for $\tilde{c}_x > 0$ and $0 < \tilde{p}_x < 1$,

$$\pi(x)^{-1}q(x,y)^{-1} = \tilde{c}_x\tilde{p}_x\,.$$

Consider an acceptance probability denoted by $\alpha_{f,(\beta)}(x,y)$ of the form (3), with

$$d(x,y) := \tilde{d}(x,y) = \frac{(1-\beta)}{\beta}\frac{(\tilde{c}_x + \tilde{c}_y)}{\tilde{c}_x\tilde{p}_x\tilde{c}_y\tilde{p}_y}\,, \tag{5}$$

Algorithm 4 presents a *flipped portkey* two-coin algorithm for $\alpha_{f,(\beta)}(x,y)$.

Theorem 5. *Algorithm 4 yields output 1 with probability*

$$\alpha_{f,(\beta)}(x,y) = \frac{\pi(y)q(y,x)}{\pi(y)q(y,x) + \pi(x)q(x,y) + \dfrac{(1-\beta)}{\beta}\dfrac{(\tilde{c}_x + \tilde{c}_y)}{\tilde{c}_x\tilde{p}_x\tilde{c}_y\tilde{p}_y}}$$

Proof. Using $\pi(x)^{-1}q(x,y)^{-1} = \tilde{c}_x\tilde{p}_x$ and following the steps of Theorem 2,

$$\Pr(\,\text{output} = 1) = \frac{\tilde{c}_x\tilde{p}_x}{\tilde{c}_x\tilde{p}_x + \tilde{c}_y\tilde{p}_y + \dfrac{(1-\beta)}{\beta}(\tilde{c}_x + \tilde{c}_y)} = \alpha_{f,(\beta)}(x,y)\,.$$

Algorithm 4 Flipped portkey two-coin algorithm

```
Draw S ~ Bernoulli(β)
if S = 0 then
    output 0
if S = 1 then
    Draw C_1 ~ Bern( c̃_x / (c̃_x + c̃_y) )
    if C_1 = 1 then
        Draw C_2 ~ Bern(p̃_x)
        if C_2 = 1 then
            output 1
        if C_2 = 0 then
            go to Step 1
    if C_1 = 0 then
        Draw C_2 ~ Bern(p̃_y)
        if C_2 = 1 then
            output 0
        if C_2 = 0 then
            go to Step 1
```

□

There are two important consequences of Theorem 5. First, by a simple application of Theorem 1, Algorithm 4 yields a π-reversible Markov chain. Second, setting $\beta = 1$, yields the usual Barker's acceptance probability, providing a second Bernoulli factory for Barker's algorithm! Variance ordering results for the flipped portkey Barker's acceptance probability can be obtained as before and is given below for completeness.

Theorem 6. *For $0 < \beta \leq 1$, $\alpha_{f,(\beta)}(x,y) \leq \beta\, \alpha_B(x,y)$. As a consequence,*

$$var(g, P_B) \leq \beta\, var(g, P_{f,(\beta)}) + (\beta - 1)\, Var_\pi(g)\,.$$

Further, if there exists $\delta > 0$ such that $\tilde{p}_x > \delta$ and $\tilde{p}_y > \delta$, then

$$\alpha_B(x,y) \leq \left(1 + \frac{1-\beta}{\delta\beta}\right) \cdot \alpha_{f,(\beta)}(x,y)\,.$$

As a consequence,

$$var\left(g, P_{f,(\beta)}\right) \leq \left(1 + \frac{1-\beta}{\delta\beta}\right) var(g, P_B) + \frac{1-\beta}{\delta\beta}\, Var_\pi(g)\,.$$

In addition to the availability of lower bounds on $\pi(x)q(y,x)$, the flipped portkey algorithm should be chosen if generating $\tilde{p}_x$ coins is easier than generating p_x coins. This advantage is demonstrated clearly in Section 5.2.

5 Examples

5.1 Gamma mixture of Weibulls

Consider a target distribution of the form

$$\pi(\theta) = \int \pi(\theta|\lambda)\,\nu(d\lambda)\,,$$

where ν is a mixing measure on λ. We assume that $\theta|\lambda \sim \text{Weibull}(\lambda, k)$, where λ is the scale parameter and k is a known shape parameter. The proposal distribution is the normal distribution centered at the current step with variance 4. It can be shown that,

$$\pi(\theta|\lambda) \leq \frac{k}{e\theta} := c_\theta .$$

Then $\pi(\theta) = c_\theta \cdot \pi(\theta)/c_\theta$, which is the desired form since events of probability $p_\theta = \pi(\theta)/c_\theta$ can be simulated. Specifically, draw $\lambda \sim \nu$ and $U \sim U[0,1]$ independently. Then $\Pr\{U \leq \pi(\theta|\lambda)/c_\theta\} = p_\theta$. We set $\nu = \text{Gamma}(10, 100)$ and $k = 10$.

We run Barker's and portkey Barker's algorithms for 10^5 steps using the two Bernoulli factories for various values of β. The true $\pi(\theta)$ in this case is unknown, however, its mean and variance are known. Trace plots and autocorrelation plots from one such run are shown in Figures 1 and 2, respectively. As expected, for smaller values of β, the acceptance probability is smaller, but the autocorrelations do not increase drastically.

We repeat the algorithms 1000 times comparing their performance. From the results in Table 1, unsurprisingly the effective sample size (ESS) (Gong and Flegal, 2016) decreases as β decreases. However, the ESS per second is significantly higher for portkey Barker's compared to Barker's. In fact, for $\beta = .90$, the portkey Barker's method is almost three times as efficient as the original Barker's method. This is a clear consequence of the large mean execution time of the two-coin algorithm. The two-coin algorithm required an average of 32 loops, while the portkey algorithms were significantly smaller. More notably, the two-coin algorithm's loops demonstrates heavy tailed behavior where the average maximum number of loops for MCMC run is 1.3 million.

5.2 MCMC on constrained spaces

Consider the following generic setup of Liechty et al. (2009). Let $f(y|\theta)$ be a likelihood and for a set $\mathcal{A}$, let $\pi(\theta|\eta)$ be the prior on θ constrained in $\mathcal{A}$, so that for a function

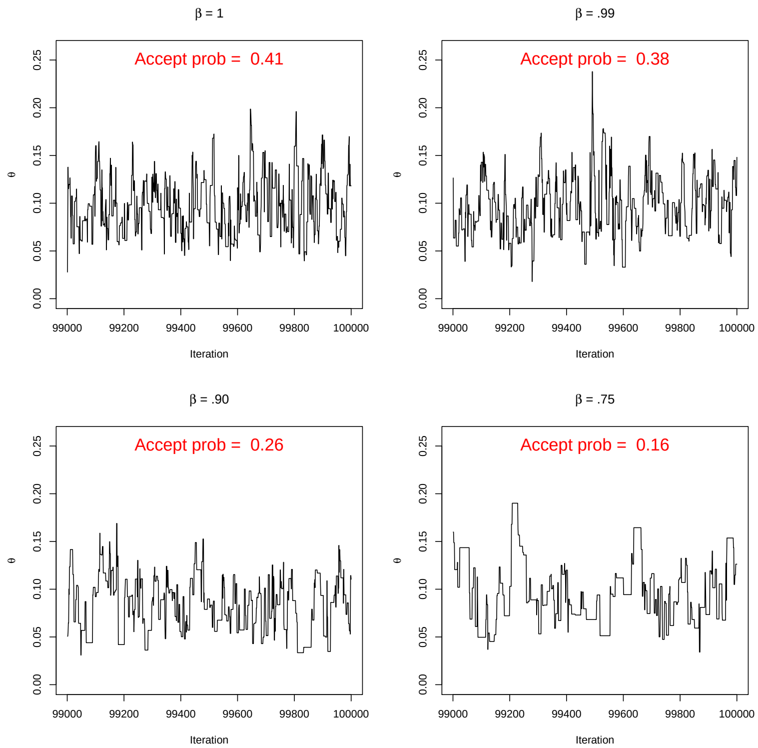

Figure 1: Trace plots of the last 1000 steps of the chains for four values of β, with acceptance probabilities are in text.

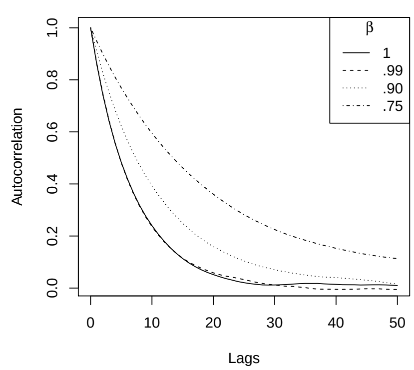

Figure 2: ACF plot for a run of length 1e5 for four β values.

$g(\cdot|\eta)$,

$$\pi(\theta|\eta) = \frac{g(\theta|\eta)I(\theta \in \mathcal{A})}{\int_{\mathcal{A}} g(\theta|\eta)d\theta} .$$

The set $\mathcal{A}$ can be complicated enough so that $\int_{\mathcal{A}} g(\theta|\eta)d\theta$ is not tractable. If η is a fixed hyperparameter, this intractability is not an issue. However, for a full Bayesian model, it is desirable to assign a hyperprior to η, and this leads to intractability in the posterior distribution for (θ, η). Chen et al. (2010) identify a shadow prior technique that modifies the Bayesian posterior to yield approximate inference. We describe this

Table 1: Averaged results from 1000 replications for sample size 1e5. Standard errors are in brackets.

β	1	.99	.90	.75
ESS	7484 (7.74)	6939 (12.18)	4320 (13.86)	2501 (9.19)
ESS/s	422.47 (3.87)	1052.66 (2.09)	1248.97 (4.15)	1159.68 (4.39)
Mean loops	32.00 (3.7)	7.63 (0.0)	3.97 (0.0)	2.55 (0.0)
Max loops	1315683 (366763.76)	604 (3.44)	78 (0.37)	32 (0.12)

technique as it relates to a Bayesian correlation estimation model and demonstrate how the flipped portkey algorithm can be used very naturally in this setting. A similar flipped portkey implementation can be implemented for other correlation estimation models of Hartman et al. (2020); Liechty et al. (2009); Philip et al. (2014); Wang and Pillai (2013), for the Bayesian graphical lasso (Wang et al., 2012), and for Bayesian semiparametric regression (Papageorgiou and Marshall, 2020).

Consider the Bayesian common correlation estimation model of Liechty et al. (2004). Suppose $y_1, \dots, y_n | R \overset{iid}{\sim} N(0, R)$ where R is a $p \times p$ correlation matrix. Liechty et al. (2004) assume the unique elements in R are normally distributed, restricted to R being positive-definite. That is, let S_p^+ be the set of $p \times p$ positive-definite matrices, then

$$f(R \mid \mu, \sigma^2) = L(\mu, \sigma^2) \prod_{i<j} \frac{1}{\sqrt{2\pi\sigma^2}} \exp\left\{-\frac{(r_{ij} - \mu)^2}{2\sigma^2}\right\} \mathbb{I}\{R \in S_p^+\}\,, \text{ where}$$

$$L^{-1}(\mu, \sigma^2) = \int_{R \in S_p^+} \prod_{i<j} \frac{1}{\sqrt{2\pi\sigma^2}} \exp\left\{-\frac{(r_{ij} - \mu)^2}{2\sigma^2}\right\} dr_{ij} \tag{6}$$

is typically not available in closed form due to the complex nature of S_p^+. For known τ^2, α, β, hyperpriors for $\mu \sim N(0, \tau^2)$ and $\sigma^2 \sim IG(a_0, b_0)$ are chosen. Interest is in the posterior distribution for (R, μ, σ^2) and Liechty et al. (2004) implement a componentwise Metropolis-within-Gibbs sampler, where for $i < j$, r_{ij} is updated conditional on all other correlations, r_{-ij}. Let $l = p(p-1)/2$. The full conditional densities are

$$f(r_{ij} \mid r_{-ij}, \mu, \sigma^2) \propto |R|^{-n/2} \exp\left\{-\frac{\text{tr}(R^{-1}Y^TY)}{2}\right\} \exp\left\{-\frac{(r_{ij} - \mu)^2}{2\sigma^2}\right\} \mathbb{I}\{l_{ij} \le r_{ij} \le u_{ij}\}\,,$$

$$f(\mu \mid R, \sigma^2) \propto L(\mu, \sigma^2) \prod_{i<j} \exp\left\{-\frac{(r_{ij} - \mu)^2}{2\sigma^2}\right\} \exp\left\{-\frac{\mu^2}{2\tau^2}\right\} := L(\mu, \sigma^2) g(\mu, R, \sigma^2)\,,$$

$$f(\sigma^2 \mid R, \mu) \propto L(\mu, \sigma^2) \prod_{i<j} \exp\left\{-\frac{(r_{ij} - \mu)^2}{2\sigma^2}\right\} \left(\frac{1}{\sigma^2}\right)^{a_0 + l/2 + 1} \exp\left\{-\frac{b_0}{\sigma^2}\right\}\,,$$

where the indicator variable $\mathbb{I}\{l_{ij} \leq r_{ij} \leq u_{ij}\}$ ensures positive-definiteness of R. The interval (l_{ij}, u_{ij}) can be obtained deterministically using the methods of Barnard et al. (2000). We use a random walk Metropolis-Hastings Gaussian proposal.

Updating μ and σ^2 requires the knowledge of $L(\mu, \sigma^2)$ which is unavailable. Liechty et al. (2004) implement a shadow prior technique that interjects the hierarchy such that $r_{ij} \sim N(\delta_{ij}, v^2)$ and $\delta_{ij} \sim N(\mu, \sigma^2)$, with unchanged hyper-priors on μ and σ^2. The resulting marginal posterior of (R, μ, σ^2) is different from the original model. The full conditionals of μ and σ^2 are available in closed-form, but the full conditionals for δ are intractable. However, Liechty et al. (2009) argue that for updating δ_{ij}, if $v^2 \approx 0$, the intractable constants can be ignored. Thus, the methodology, although convenient, is not asymptotically exact since the resulting Markov chain targets an approximation of a modified desired distribution.

The flipped portkey two-coin algorithm can easily be implemented for both μ and σ^2 and we only present details for μ. We use Gaussian random walk proposals for both components so that we can ignore the proposal density in the Bernoulli factories. Although $L^{-1}(\mu, \sigma^2) \leq 1$, we can obtain a tighter upper bound by noting that

$$g(\mu, R, \sigma^2)^{-1} L^{-1}(\mu, \sigma^2) \leq g(\mu, R, \sigma^2)^{-1} \left[\Phi\left(\frac{1-\mu}{\sigma}\right) - \Phi\left(\frac{-1-\mu}{\sigma}\right)\right]^l := \tilde{c}_\mu \,.$$

Set

$$f(\mu | R, \sigma^2)^{-1} = \tilde{c}_\mu \frac{f(\mu \mid R, \sigma^2)^{-1}}{\tilde{c}_\mu} := \tilde{c}_\mu \tilde{p}_\mu \,.$$

Generating coins of probability $\tilde{p}_\mu$ is straightforward: draw $z_{ij} \sim \text{TN}(-1, 1, \mu, \sigma^2)$ for $i < j$, and construct matrix Z with z_{ij} as the lower-triangular entries. If Z is positive-definite, return 1, else return 0. Notice here that the use of the flipped portkey algorithm over the portkey algorithm makes it much easier to simulate the p coin since $L^{-1}(\mu, \sigma^2)$ takes the form of an integral.

Consider the daily closing prices of major European stocks: Germany DAX, Switzerland SMI, France CAC, and United Kingdom FTSE on each day, not including weekends and holidays from 1991-1998. The data are available in the `datasets` R package and has 1860 observations. The goal is to estimate the correlation matrix of the four stock prices using the model specified above.

We set $\beta = .90$ for both the updates for μ and σ^2 and first compare the estimated posterior density of μ and σ^2 from a run of length 10^5 for both the shadow prior method and the two Bernoulli factory MCMC algorithms. Figure 3 presents the results, where it is evident that the shadow prior method yields a biased posterior distribution.

Figure 4 presents the autocorrelation and trace plots for standard Barker's ($\beta = 1$) and

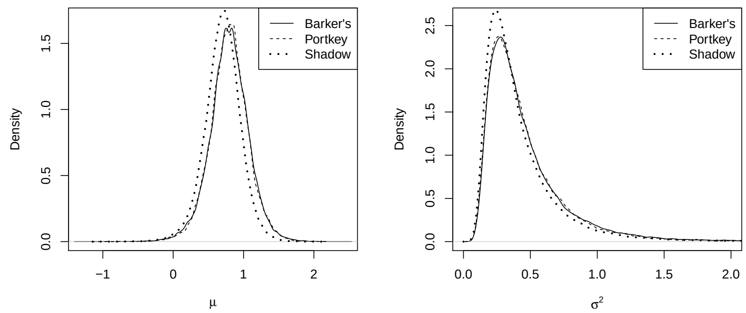

Figure 3: Estimated posterior density of μ and σ^2 from Markov chain lengths of 10^5 with $v^2 = .001$ as recommended by Chen et al. (2010).

the flipped portkey Barker's ($\beta = .90$). Portkey Barker's algorithm leads to only slightly slower mixing of the chain. However, the (log of) the number of Bernoulli factory loops required for the portkey two-coin algorithm is far less. Particularly for updating μ, the original two-coin algorithm requires an exponentially large number of Bernoulli factory loops whenever unlikely values are proposed. This makes it laborious to tune Barker's algorithm and encourages short jumps.

We repeat the above experiment 10 times for MCMC runs of length 10^4. Ideally we would have liked to increase the Monte Carlo replication size, but the original two-coin algorithm often got stuck in a loop for days. The results are presented in Table 2 with estimated ESS (Vats et al., 2019) calculated using the `R` package `mcmcse` (Flegal et al., 2017). Although a regular Barker's implementation yields a slightly higher ESS, the ESS per second for the flipped portkey Barker's algorithm is about 1.5 times higher on average. As further demonstrated in the table, this is clearly a consequence of the large number of loops required for the two-coin algorithm for updating μ.

Table 2: Averaged results from 10 replications for Monte Carlo sample sizes of 10^4. Standard errors are indicated in brackets.

β	1	.90
ESS	542 (13.50)	496 (9.00)
ESS/s	9.63 (1.992)	14.83 (0.279)
Mean loops μ	218.43 (148.89)	2.99 (0.010)
Mean loops σ^2	3.21 (0.02)	2.49 (0.010)
Max loops μ	2084195 (1491777)	34 (2.94)
Max loops σ^2	38 (1.13)	27 (1.51)

The original two-coin algorithm can work reasonably well here, however, every once in a while, an improbable far away value is proposed, particularly for μ, which leads to a

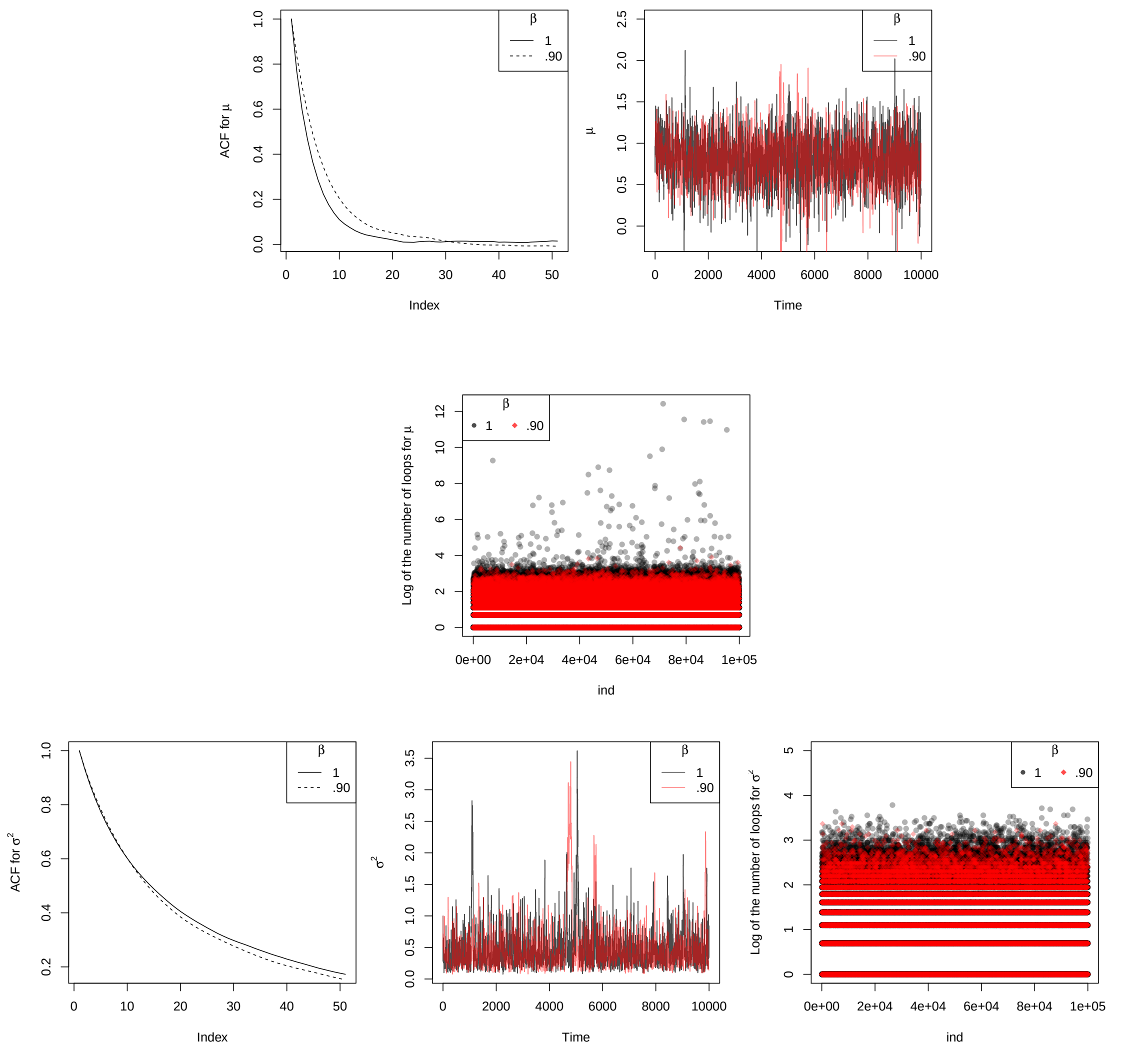

Figure 4: ACF, trace, and (log of) Bernoulli factory loops for Barker's and portkey Barker's for μ (right) and σ^2 (left).

large number of loops in its Bernoulli factory. The portkey trick stabilizes this behavior, with little loss in statistical efficiency, and an overall two-fold gain in efficiency.

5.3 Bayesian inference for the Wright-Fisher diffusion

A collection of methodologies to perform exact inference for discretely-observed diffusions has been proposed in the last 15 years (see Beskos et al., 2009, 2006; Sermaidis et al., 2013). These are all based on algorithms for exact simulation of diffusions (Beskos et al., 2008). Although these methodologies can be applied to a wide class of univariate diffusion models, some important ones are left out. For example, the Cox-Ingersoll-Ross model used to describe the evolution of interest rates, and the Wright-Fisher diffusion, which is widely used in genetics to model the evolution of the frequency of a genetic variant or allele, in a large randomly mating population.

The methodologies for exact inference for jump-diffusion models based on the exact simulation of jump-diffusions are considerably more restrict (see Gonçalves et al., 2017b; Gonçalves and Roberts, 2014). Let α be the drift function of the Lamperti transform of the original diffusion. Whilst for diffusion models $\alpha^2 + \alpha'$ is required to be bounded below, for jump-diffusions it is also required to have uniformly bounded drift and jump rate of the Lamperti transformed process.

As demonstrated in Gonçalves et al. (2017a,b), the two-coin Barker's MCMC algorithm performs exact inference for diffusions and jump-diffusions without requiring the above conditions. The Portkey two-coin Barker's MCMC proposed here will typically provide considerable gain when compared to the original two-coin Barker's, allowing for a wider applicability of the methodology in terms of model complexity and data size.

Consider the neutral Wright-Fisher family of diffusions with mutations

$$dY_s = \frac{1}{2}(\theta_1(1 - Y_s) - \theta_2 Y_s) + \sqrt{Y_s(1 - Y_s)}dW_s \qquad Y_0 = y_0, \quad \theta_1, \theta_2 > 0\,, \tag{7}$$

where W_s is a standard Brownian motion. The process Y lives in $[0, 1]$ and we assume it is observed at a finite set of time points, $0 = t_0, t_1, \dots, t_n = T$ so that the observed data are $Y_{\text{obs}} = (y_0, y_1, \dots, y_n)^T$. The parameters of interest are (θ_1, θ_2) which describe the drift coefficient of the process and Bayesian estimation yields desirable theoretical properties (Sant et al., 2020). Yet, their transition densities $p(\gamma|y_{i-1}, y_i) := \Pr_\gamma(Y_{t_i} \in dy_i|Y_{t_i-1} = y_{i-1})/dy_i$, that determine the likelihoods, are not available in closed-form and are characterised by poor numerical properties near the 0 and 1 barriers (see Jenkins and Spano, 2017) that undermine reliable Bayesian inference based on approximations.

Inference will be conducted for the following reparameterization

$$\gamma_1 = \theta_1 + \theta_2 \quad \text{and} \quad \gamma_2 = \frac{\theta_1}{\theta_1 + \theta_2},$$

with uniform priors on $\gamma = (\gamma_1, \gamma_2)$. The first step is to apply the Lamperti transform to Y in (7), which leads to a unit diffusion coefficient process X. Now we specify a Gibbs sampler alternating between updating γ and updating the missing paths, X_{mis}, of the transformed diffusion X given the transformed observations X_{obs}.

For the X_{mis} update, we use standard Brownian bridge proposals restricted to $[0, 1]$ and implement both Barker's and portkey Barker's MCMC. The Brownian bridges are sampled using a layered Brownian bridge construction, which allows lower and upper bounds on the likelihood that are crucial to devise an efficient Bernoulli factory.

For updating γ, component-wise updates are done with uniform random walk proposal distributions, $U(\gamma_1 \pm 0.3)$ and $U(\gamma_2 \pm 0.01)$ for γ_1 and γ_2, respectively. It can be shown that $\pi(\gamma | X_{\text{mis}}, X_{\text{obs}}) = c_\gamma \, p_\gamma$, where the forms of c_γ and p_γ can be found in Gonçalves et al. (2017a). A realization of probability p_γ is obtained using the layer refinement strategy described in Gonçalves et al. (2017b). Here each interval (t_{i-1}, t_i) is broken into refinements, and a layered Brownian bridge is constructed within each refinement. Without the refinement, the bound c_γ is too loose and thus p_γ is often too small for the Bernoulli factory to be efficient. However, finer refinements require simulating more layered Brownian bridges leading to much larger computation times to even draw from the proposal distribution. It is precisely this characteristic of the sampling process that dictates the superior performance of the portkey two-coin algorithm.

We simulate a Wright-Fisher diffusion with $\gamma_1 = 8$ and $\gamma_2 = 0.5$ and observe the process at times $\{0, 1, \ldots, 50\}$. The data are shown in Figure 5. We use the component-wise sampler described above to sample from the posterior distribution of (γ_1, γ_2). We run the two samplers for 10000 steps where the Barker's algorithm takes about 34.5 hours and the portkey Barker's algorithm takes about 3.5 hours.

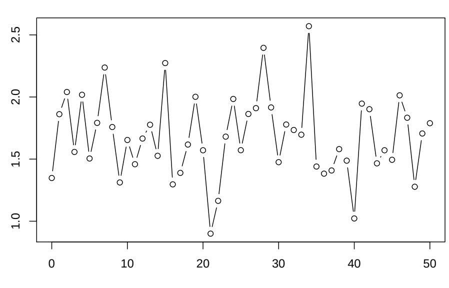

Figure 5: Simulated data observed at 51 time points.

First, the update for X_{mis} performs similarly for both Barker's and portkey Barker's, where we set $\beta = 0.98$. The average number of loops over $1e4$ samples for the two-coin and the portkey two-coin algorithms are 1.698 and 1.645, respectively, indicating that the original two-coin algorithm already works well enough for this component. We note specifically that close to nothing is lost by using portkey Barker's instead of Barker's acceptance probability.

For updating the parameters, the βs are set to be .99995 and .9995 for γ_1 and γ_2, respectively. In Figure 6 are the number of loops of the two Bernoulli factories for each iteration of the two Markov chains for both γ_1 and γ_2. The γ_1 update of the Barker's two-coin Markov chain requires an average of 489 loops with a maximum of 183077 loops. In comparison, the portkey two-coin algorithm runs an average of 43 loops with a maximum of 12622. The long tails of the number of loops for the original two-coin algorithm mean that the average run-time is quite slow, as evidenced from the computation times. However, since both βs are close to 1, this gain in computational

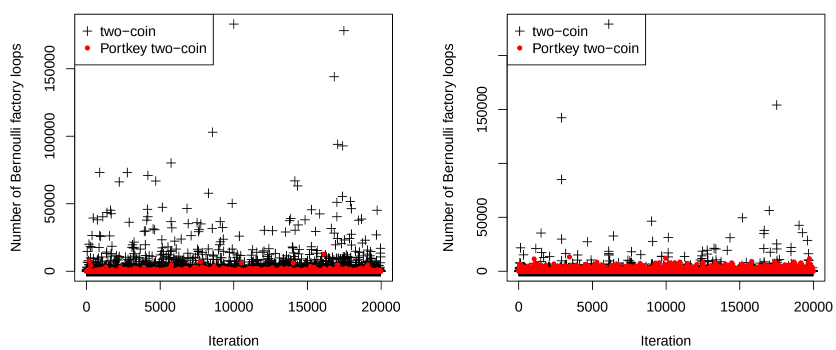

Figure 6: Comparing the number of loops of the Bernoulli factory for two-coin and portkey two-coin algorithm. Left is for the γ_1 update and right is for the γ_2 update.

efficiency mainly impacts the Bernoulli factory, and has only a small effect on the Markov chain as witnessed by the autocorrelation plots in Figure 7. The estimated

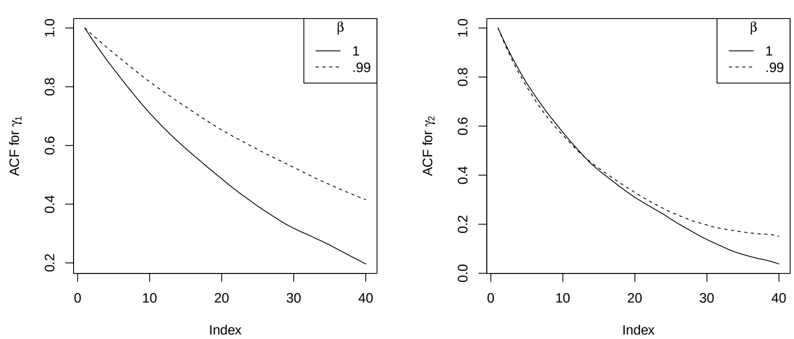

Figure 7: Autocorrelation plots for γ_1 and γ_2.

ESS for the posterior mean of γ is 255 and 197, respectively for Barker's and portkey Barker's. However, the ESS per hour is 7.36 and 56.45, respectively. Thus, the portkey Barker's algorithm is ≈ 7 times more efficient that the original Barker's algorithm.

Finally, we stress that implementation of the portkey two-coin algorithm requires only minor changes to the code for any two-coin algorithm.

6 Discussion

Motivated by Bernoulli factories, we introduce a family of MCMC acceptance probabilities for intractable target distributions[2]. We argue that the (flipped) portkey two-coin algorithm is a robust alternative to the two-coin algorithm; when the two-coin algorithm is efficient, the portkey two-coin algorithm is essentially similar to the two-coin algorithm, but when the two-coin algorithm is computationally infeasible, the portkey two-coin algorithm is significantly more stable, having a smaller mean execution time. The resulting Markov chain is exact and for values of β close to 1, the loss in statistical efficiency is small compared to the gain in computational efficiency. Moreover, tuning the portkey Barker's sampler is far easier since feedback from the algorithm is significantly faster than the original Barker's sampler.

Finally, the (portkey) two-coin algorithm finds immediate use in Bayesian inference for diffusions and jump-diffusions, as illustrated in Section 5.3 and Bayesian models with intractable priors. To implement these algorithms more generally requires reasonable bounds on the target distribution. We believe the solution to this is to construct local bounds compatible with these algorithms and this makes for important future work.

7 Acknowledgements

Dootika Vats was supported by the National Science Foundation (DMS/1703691). Flávio Gonçalves would like to thank FAPEMIG, CNPq and the University of Warwick for financial support. Krzysztof Łatuszyński is supported by the Royal Society through the Royal Society University Research Fellowship. Gareth Roberts is supported by the EPSRC grants: *ilike* (EP/K014463/1), CoSInES (EP/R034710/1) and Bayes for Health (EP/R018561/1).

[2]An `R` package that implements the Bernoulli factories is here: `https://github.com/dvats/portkey`

References

Andrieu, C. and Roberts, G. O. (2009). The pseudo-marginal approach for efficient Monte Carlo computations. *The Annals of Statistics*, 37(2):697–725.

Asmussen, S., Glynn, P. W., and Thorisson, H. (1992). Stationarity detection in the initial transient problem. *ACM Transactions on Modeling and Computer Simulation (TOMACS)*, 2:130–157.

Barker, A. A. (1965). Monte Carlo calculations of the radial distribution functions for a proton-electron plasma. *Australian Journal of Physics*, 18:119–134.

Barnard, J., McCulloch, R., and Meng, X.-L. (2000). Modeling covariance matrices in terms of standard deviations and correlations, with application to shrinkage. *Statistica Sinica*, pages 1281–1311.

Beskos, A., Papaspiliopoulos, O., and Roberts, G. O. (2008). A new factorisation of diffusion measure and sample path reconstruction. *Methodology and Computing in Applied Probability*, 10:85–104.

Beskos, A., Papaspiliopoulos, O., and Roberts, G. O. (2009). Monte Carlo maximum likelihood estimation for discretely observed diffusion processes. *The Annals of Statistics*, 37:223–245.

Beskos, A., Papaspiliopoulos, O., Roberts, G. O., and Fearnhead, P. (2006). Exact and computationally efficient likelihood-based inference for discretely observed diffusion processes (with discussion). *Journal of the Royal Statistical Society. Series B*, 68:333–382.

Billera, L. J. and Diaconis, P. (2001). A geometric interpretation of the Metropolis algorithm. *Statistical Science*, 16:335–339.

Chen, M.-H., Müller, P., Sun, D., Ye, K., and Dey, D. K. (2010). *Frontiers of statistical decision making and Bayesian analysis: In Honor of James O. Berger*. Springer Science & Business Media.

Flegal, J. M., Hughes, J., Vats, D., and Dai, N. (2017). *mcmcse: Monte Carlo Standard Errors for MCMC*. Riverside, CA and Minneapolis, MN. R package version 1.3-2.

Gonçalves, F. B., Łatuszyński, K., Roberts, G. O., et al. (2017a). Barker's algorithm for Bayesian inference with intractable likelihoods. *Brazilian Journal of Probability and Statistics*, 31:732–745.

Gonçalves, F. B., Łatuszyński, K. G., and Roberts, G. O. (2017b). Exact Monte Carlo likelihood-based inference for jump-diffusion processes. *arXiv preprint arXiv:1707.00332*.

Gonçalves, F. B. and Roberts, G. O. (2014). Exact simulation problems for jump-diffusions. *Methodology and Computing in Applied Probability*, 16:907–930.

Gong, L. and Flegal, J. M. (2016). A practical sequential stopping rule for high-dimensional Markov chain Monte Carlo. *Journal of Computational and Graphical Statistics*, 25:684–700.

Hartman, B., Groendyke, C., and Engler, D. (2020). Bayesian multivariate regime-switching models and the impact of correlation structure misspecification in variable annuity pricing. *Scandinavian Actuarial Journal*, 2020(2):152–171.

Hastings, W. K. (1970). Monte Carlo sampling methods using Markov chains and their applications. *Biometrika*, 57(1).

Huber, M. (2017). Optimal linear Bernoulli factories for small mean problems. *Methodology and Computing in Applied Probability*, 19:631–645.

Jenkins, P. A. and Spano, D. (2017). Exact simulation of the Wright–Fisher diffusion. *The Annals of Applied Probability*, 27:1478–1509.

Keane, M. and O'Brien, G. L. (1994). A Bernoulli factory. *ACM Transactions on Modeling and Computer Simulation (TOMACS)*, 4:213–219.

Łatuszyński, K., Kosmidis, I., Papaspiliopoulos, O., and Roberts, G. O. (2011). Simulating events of unknown probabilities via reverse time martingales. *Random Structures Algorithms*, 38(4):441–452.

Łatuszyński, K. and Roberts, G. O. (2013). CLTs and asymptotic variance of time-sampled Markov chains. *Methodology and Computing in Applied Probability*, 15:237–247.

Liang, F., Liu, C., and Carroll, R. J. (2007). Stochastic approximation in Monte Carlo computation. *Journal of the American Statistical Association*, 102(477):305–320.

Liechty, J. C., Liechty, M. W., and Müller, P. (2004). Bayesian correlation estimation. *Biometrika*, 91(1):1–14.

Liechty, M. W., Liechty, J. C., and Müller, P. (2009). The shadow prior. *Journal of Computational and Graphical Statistics*, 18:368–383.

Metropolis, N., Rosenbluth, A. W., Rosenbluth, M. N., Teller, A. H., and Teller, E. (1953). Equations of state calculations by fast computing machine. *Journal of Chemical Physics*, 21.

Morina, G., Łatuszyński, K., Nayar, P., and Wendland, A. (2019). From the Bernoulli factory to a dice enterprise via perfect sampling of Markov chains. *arXiv preprint arXiv:1912.09229*.

Nacu, c. and Peres, Y. (2005). Fast simulation of new coins from old. *The Annals of Applied Probability*, 15(1A):93–115.

Papageorgiou, G. and Marshall, B. C. (2020). Bayesian semiparametric analysis of multivariate continuous responses, with variable selection. *Journal of Computational and Graphical Statistics*, pages 1–14.

Park, J. and Haran, M. (2018). Bayesian inference in the presence of intractable normalizing functions. *Journal of the American Statistical Association*, 113(523):1372–1390.

Peskun, P. (1973). Optimum Monte-Carlo sampling using Markov Chains. *Biometrika*, 89:745–754.

Philip, L., Li, W. K., and Ng, F. (2014). Formulating hypothetical scenarios in correlation stress testing via a Bayesian framework. *The North American Journal of Economics and Finance*, 27:17–33.

Sant, J., Jenkins, P. A., Koskela, J., and Spano, D. (2020). Convergence of Bayesian estimators for diffusions in genetics. *arXiv preprint arXiv:2001.03527*.

Sermaidis, G., Papaspiliopoulos, O., Roberts, G. O., Beskos, A., and Fearnhead, P. (2013). Markov chain Monte Carlo for exact inference for diffusions. *Scandinavian Journal of Statistics*, 40:294–321.

Tierney, L. (1994). Markov chains for exploring posterior distributions (with discussion). *The Annals of Statistics*, 22:1701–1762.

Vats, D., Flegal, J. M., and Jones, G. L. (2019). Multivariate output analysis for Markov chain Monte Carlo. *Biometrika*, 106(2):321–337.

Wang, H. et al. (2012). Bayesian graphical lasso models and efficient posterior computation. *Bayesian Analysis*, 7:867–886.

Wang, H. and Pillai, N. S. (2013). On a class of shrinkage priors for covariance matrix estimation. *Journal of Computational and Graphical Statistics*, 22:689–707.

Zanella, G. (2020). Informed proposals for local MCMC in discrete spaces. *Journal of the American Statistical Association*, 115:852–865.