Improved diffusion Monte Carlo

April 9, 2014

Martin Hairer[1] and Jonathan Weare[2]

[1] Mathematics Department, the University of Warwick
[2] Department of Statistics and James Franck Institute, the University of Chicago
Email: M.Hairer@Warwick.ac.uk, Email: weare@uchicago.edu

Abstract

We propose a modification, based on the RESTART (repetitive simulation trials after reaching thresholds) and DPR (dynamics probability redistribution) rare event simulation algorithms, of the standard diffusion Monte Carlo (DMC) algorithm. The new algorithm has a lower variance per workload, regardless of the regime considered. In particular, it makes it feasible to use DMC in situations where the "naïve" generalisation of the standard algorithm would be impractical, due to an exponential explosion of its variance. We numerically demonstrate the effectiveness of the new algorithm on a standard rare event simulation problem (probability of an unlikely transition in a Lennard-Jones cluster), as well as a high-frequency data assimilation problem.

Keywords: Diffusion Monte Carlo, quantum Monte Carlo, rare event simulation, sequential Monte Carlo, particle filtering, Brownian fan, branching process

Contents

1   Introduction

Diffusion Monte Carlo (DMC) is a well established method popularized within the Quantum Monte Carlo (QMC) community to compute the ground state energy (the lowest eigenvalue) of the Hamiltonian operator

$$\mathcal{H}\psi = -\Delta\psi + V\psi$$

as well as averages with respect to the square of the corresponding eigenfunction [Kal62, GS71, And75, CA80, KL11]. It is based on the fact that the Feynman–Kac formula

$$\int f(x)\psi(x,t)dx = \mathbf{E}\Big(f(B_t)\exp\Big(-\int_0^t V(B_s)ds\Big)\Big) , \qquad (1.1)$$

connects the solution, $\psi(x, t)$, of the partial differential equation

$$\partial_t \psi = -\mathcal{H}\psi$$
$$\psi(0, \cdot) = \delta_0 \, , \tag{1.2}$$

to expectations of a standard Brownian motion B_t starting at x. For large times t, suitably normalised integrals against the solution of (1.2) approximate normalised integrals against the ground state eigenfunction of $\mathcal{H}$.

As we will see in the next section, DMC is an extremely flexible tool with many potential applications going well beyond quantum Monte Carlo. In fact, the basic operational principles of DMC, as described in this article, can be traced back at least to what would today be called a rare event simulation scheme introduced in [HM54] and [RR55]. In the data assimilation context a popular variant of DMC, sequential importance sampling (see e.g. [dDG05, Liu02]), computes normalised expectations of the form

$$\frac{\mathbf{E}\Big(f(y_t) \exp\Big(-\sum_{t_k \leq t} V(k, y_{t_k}) \Big) \Big)}{\mathbf{E}\exp\Big(-\sum_{t_k \leq t} V(k, y_{t_k}) \Big)} \, ,$$

where the $V(k, \cdot)$ are functions encoding, for example, the likelihood of sequentially arriving observations (at times $t_1 < t_2 < t_3 < \cdots$) given the state of some Markov process y_t. A central component of a sequential importance sampling scheme is a so-called "resampling" step (see e.g. [dDG05, Liu02]) in which copies of a system are weighted by the ratio of two densities and then resampled to produce an unweighted set of samples. Some popular resampling algorithms are adaptations of the generalised version of DMC that we will present below in Algorithm DMC (e.g. residual resampling as described in [Liu02]). Our results suggest that sequential importance sampling schemes could be improved (in some cases dramatically) by building the resampling step from our modification of DMC in Algorithm TDMC.

One can imagine a wide range of potential uses of DMC. For example, we will show that the generalisation of DMC in Algorithm DMC below could potentially be used to compute approximations to quantities of the form

$$\mathbf{E}\Big(f(y_t) \exp\Big(-\int_0^t V(y_t) dy_t \Big) \Big) \, ,$$

or

$$\mathbf{E}(f(y_t) \exp\left(-V(y_t) \right)) \, . \tag{1.3}$$

where y_t is a diffusion. Continuous time data assimilation requires the approximation of expectations similar to first type above while, when applied to computing expectations of the type in (1.3), DMC becomes a rare event simulation technique (see e.g. [HM54, RR55, AFRtW06, JDMD06]), i.e. a tool for efficiently generating samples of extremely low probability events and computing their probabilities. Such tools can be used to compute the probability or frequency of dramatic fluctuations in a stochastic process. Those fluctuations might, for example, characterise a reaction in a chemical system or a failure event in an electronic device (see e.g. [FS96, Buc04]).

The mathematical properties of DMC have been explored in great detail (see e.g. [DM04]). Particularly relevant to our work is the thesis [Rou06] which studies the continuous time limit of DMC in the quantum Monte Carlo context and introduces schemes for that setting that bear some resemblance to our modified scheme Algorithm TDMC. The initial motivation for the present work is the observation that, in

some of the most interesting settings, the natural generalisation of DMC introduced in Algorithm DMC exhibits a dramatic instability. We show that this instability can be corrected by a slight enlargement of the state space to include a parameter dictating which part of the state space a sample is allowed to visit. The resulting method is introduced in Algorithm TDMC in Section 2 below. This modification is inspired by the RESTART and DPR rare event simulation algorithms [VAVA91, HT99, DD11]. The RESTART and DPR algorithms bias an underlying Markov process by splitting the trajectory of the process into multiple trajectories and then appropriately reweighing the resulting trajectories. The splitting occurs every time the process moves from one set to another in a predefined partition of the state space. The key feature of the algorithms that we borrow is a restriction placed on the trajectories resulting from a splitting event. The new trajectories are only allowed to explore a certain region of state space and are eliminated when they exit this region. Our modification of DMC is similar, except that our branching rule is not necessarily based on a fixed partition of state space.

In Section 3, we demonstrate the stability and utility of the new method in two applications. In the first one, we use the method to compute the probability of very unlikely transitions in a small cluster of Lennard-Jones particles. In the second example we show that the method can be used to efficiently assimilate high frequency observational data from a chaotic diffusion. In addition to these computational examples, we provide an in-depth mathematical study of the new scheme (Algorithm TDMC). We show that regardless of parameter regime and even underlying dynamics (i.e. y_t need not be related to a diffusion) the estimator generated by the new algorithm has lower variance per workload than the DMC estimator. In a companion article [HW14], by focusing on a particular asymptotic regime (small time-discretisation parameter) we are also able to rigorously establish several dramatic results concerning the stability of Algorithm TDMC in settings in which the straightforward generalisation of DMC is unstable. In particular, in that work we provide a characterization and proof of convergence in the continuous time limit to a well-behaved limiting Markov process which we call the Brownian fan.

1.1 Notations

In our description and discussion of the methods below we will use the standard notation $\lfloor a \rfloor = \max\{i \in \mathbb{Z} : i \leq a\}$.

Acknowledgements

2 The algorithm and a summary of our main results

With the potential term V removed, the PDE (1.2) is simply the familiar Fokker-Planck (or forward Kolmogorov) equation and one can of course approximate (1.1) (with $V = 0$) by

$$\widehat{f}_t = \frac{1}{M} \sum_{j=1}^{M} f(w_t^{(j)}) \,,$$

where the $w^{(j)}$ are M independent realisations of a Brownian motion. The probabilistic interpretation of the source term $V\psi$ in the PDE is that it is responsible for the killing and creation of sample trajectories of w. In practice one cannot exactly compute the integral appearing in the exponent in (1.1). Common practice (assuming that $t_k = k\varepsilon$) is to replace the integral by the approximation

$$\int_0^t V(w_s)\,ds \approx \sum_{k=0}^{\lfloor t/\varepsilon \rfloor - 1} \frac{1}{2}\big(V(w_{t_{k+1}}) + V(w_{t_k})\big)\varepsilon \,,$$

where $\varepsilon > 0$ is a small time-discretisation parameter. DMC then approximates

$$\langle f \rangle_t = \mathbf{E}\bigg(f(w_t)\exp\Big(-\sum_{k=0}^{\lfloor t/\varepsilon \rfloor - 1} \frac{1}{2}\big(V(w_{t_{k+1}}) + V(w_{t_k})\big)\varepsilon\Big)\bigg). \qquad (2.1)$$

The version of DMC that we state below in Algorithm DMC is a slight generalisation of the standard algorithm and approximates

$$\langle f \rangle_t = \mathbf{E}\bigg(f(y_t)\exp\Big(-\sum_{t_k \leq t} \chi(y_{t_k}, y_{t_{k+1}})\Big)\bigg)\,, \qquad (2.2)$$

where y_t is any Markov process and χ is any function of two variables. Strictly for convenience we will assume that the time t is among the discretization points $t_0, t_1, t_2, \ldots$. The DMC algorithm proceeds as follows:

Algorithm DMC *Slightly generalised DMC*

1. Begin with M copies $x_0^{(j)} = x_0$ and $k = 0$.

2. At step k there are N_{t_k} samples $x_{t_k}^{(j)}$. Evolve each of these $t_{k+1} - t_k$ units of time under the underlying dynamics to generate N_{t_k} values
$$\tilde{x}_{t_{k+1}}^{(j)} \sim \mathbf{P}(y_{t_{k+1}} \in dx \,|\, y_{t_k} = x_{t_k}^{(j)}).$$

3. For each $j = 1, \ldots, N_{t_k}$, let
$$P^{(j)} = e^{-\chi(x_{t_k}^{(j)}, \tilde{x}_{t_{k+1}}^{(j)})}$$
and set
$$N^{(j)} = \lfloor P^{(j)} + u^{(j)} \rfloor \,,$$
where the $u^{(j)}$ are independent $\mathcal{U}(0,1)$ random variables.

4. For $j = 1, \ldots, N_{t_k}$, and for $i = 1, \ldots, N^{(j)}$ set
$$x_{t_{k+1}}^{(j,i)} = \tilde{x}_{t_{k+1}}^{(j)}.$$

5. Finally, set $N_{t_{k+1}} = \sum_{j=1}^{N_{k\varepsilon}} N^{(j)}$ and list the $N_{t_{k+1}}$ vectors $\{x_{t_{k+1}}^{(j,i)}\}$ as $\{x_{t_{k+1}}^{(j)}\}_{j=1}^{N_{t_{k+1}}}$.

6. At time t, produce the estimate

$$\widehat{f_t} = \frac{1}{M} \sum_{j=1}^{N_t} f(x_t^{(j)}) \, .$$

Here, the notation $\mathcal{U}(a, b)$ was used to refer to the law of a random variable that is uniformly distributed in the interval (a, b). Below we will refer to the $x_t^{(j)}$ as *particles* and to a collection of particles as an *ensemble*.

Remark 2.1 If the goal is to compute the "conditional" quantity $\langle f \rangle_t / \langle 1 \rangle_t$ then one can simply redefine $P^{(j)}$ in Step 3 by

$$P^{(j)} = \frac{e^{-\chi(x_{t_k}^{(j)}, \tilde{x}_{t_{k+1}}^{(j)})}}{\frac{1}{N_t} \sum_{l=1}^{N_t} e^{-\chi(x_{t_k}^{(l)}, \tilde{x}_{t_{k+1}}^{(l)})}} \, .$$

With this choice the copy number N_t becomes a Martingale and will need to be controlled at the cost of some ensemble size dependent bias. The value of N_t can be maintained exactly at some predefined value M by uniformly upsampling or downsampling the ensemble after each iteration. Alternatively one can hold N_t near M by multiplying this $P^{(j)}$ by $(M/N_t)^\alpha$ for some $\alpha > 0$ as we do in Section 3.2.

Algorithm DMC results in an unbiased estimate of (2.2), i.e.

$$\mathbf{E}^1 \widehat{f_t} = \langle f \rangle_t \, .$$

For reasonable choices of χ and f (e.g. $\sup_{t_k \leq t} \langle 1 \rangle_{t_k} < \infty$ and $\langle |f| \rangle_t < \infty$) the law of large numbers then implies that

$$\lim_{M \to \infty} \widehat{f_t} = \langle f \rangle_t.$$

We use the symbol $\mathbf{E}^1$ for expectations under the rules of Algorithm DMC to distinguish them for expectations under the rules of our modified algorithm (Algorithm TDMC below) which will simply be denoted by $\mathbf{E}$.

Of course if we set

$$\chi(x, y) = \frac{1}{2}(V(x) + V(y))(t_{k+1} - t_k) \, , \tag{2.3}$$

then we are back to the quantum Monte Carlo setting. We might, however, choose

$$\chi(x, y) = V(x)(y - x) \, , \tag{2.4}$$

which, if y_t approximates a diffusion (which we also denote by y_t), formally corresponds to an approximation of

$$\langle f \rangle_t = \mathbf{E}\left(f(y_t) \exp\left(-\int_0^t V(y_s) dy_s \right) \right) \, ,$$

or

$$\chi(x, y) = V(y) - V(x) \, , \tag{2.5}$$

corresponding to an approximation of

$$\langle f \rangle_t = e^{V(x_0)} \mathbf{E}\Big(f(y_t) e^{-V(y_t)} \Big).$$

The generalisation of DMC resulting from various choices of χ is not frivolous. In Section 3 we will see that, with χ of form (2.5), one can design potentially very useful rare event and particle filtering schemes. Unfortunately, when y_t is a diffusion or a discretisation of a diffusion, the resulting Algorithms behave extremely poorly in the small discretisation size limit ($t_{k+1} - t_k \to 0$). The reason is simple: on a time interval $[t_k, t_{k+1})$, a typical step of Brownian motion moves by $\mathcal{O}(\sqrt{t_{k+1} - t_k})$. Therefore, unlike for (2.3), for both (2.4) and (2.5), the typical size of $\chi(x, y)$ is of order $\sqrt{t_{k+1} - t_k}$. As a consequence, at each step, the probability that a particle either dies or spawns a child is itself of order $\sqrt{t_{k+1} - t_k}$. Without modification, as $t_{k+1} - t_k \to 0$, the fluctuations in the number of particles, N_t, will grow wildly and the process will die out before a time of order 1 with higher and higher probability, thus causing the variance of our estimator to explode. This phenomenon is already evident in the simple case of a Brownian motion,

$$y_{(k+1)\varepsilon} = y_{k\varepsilon} + \sqrt{\varepsilon}\xi_{k+1} , \qquad X(0) = 0 , \tag{2.6}$$

with

$$\chi(x, y) = y - x \tag{2.7}$$

(a choice consistent with either (2.4) or (2.5)). Here the ξ_k are independent mean 0 and variance 1 Gaussian random variables. This example (with more general ξ_k) is analysed in great depth in the companion paper [HW14].

Figure 2 demonstrates the dramatic failure of the straightforward generalisation of DMC in Algorithm DMC on this simple example. There we plot the logarithm of the second moment of the number of particles in the ensemble at time 1 (i.e. after $1/\varepsilon$ steps of the algorithm) versus $-\log \varepsilon$ for several values of ε. Clearly, $\mathbf{E}\big(N_1^2\big)$ is growing rapidly as ε decreases. Indeed, for small enough ε and starting with a single initial copy at the origin, one can exploit the conditional independence of offspring in Algorithm DMC to show that for the simple random walk example, $\mathbf{E}\big[N_{k\varepsilon}^2\big] \geq \mathbf{E}\big[N_{(k-1)\varepsilon}^2\big] + \alpha\sqrt{\varepsilon}$ for a positive constant α. After iterating ε^{-1} times, this bound gives exactly the growth of $\mathbf{E}\big[N_1^2\big]$ illustrated in Figure 2 (i.e. $\mathcal{O}\big(\varepsilon^{-1/2}\big)$). This instability can be removed by carrying out the branching steps (Steps 3-5) only every $\mathcal{O}(1)$ units of time instead of every ε units of time and accumulating weights for the particles in the intervening time intervals. However, this small ε regime cleanly highlights a serious (and unnecessary) deficiency in DMC. In contrast to Algorithm DMC, the method that we will describe in Algorithm TDMC below, appears to be stable as ε vanishes (a property confirmed in [HW14]).

Our modification of DMC described in Algorithm TDMC below not only suppresses the small ε instability just described, but results in a more robust and efficient algorithm in any context. Inspired by the RESTART and DPR algorithms studied in [VAVA91, HT99, DD11] we append to each particle a "ticket" θ limiting the region that the particle is allowed to visit. Samples are eliminated when and only when they violate their ticket values. As a consequence, particles typically survive for a much longer time, and in many situations there is a well-defined limiting algorithm when $\varepsilon \to 0$. More precisely, the modified algorithm proceeds as follows:

Algorithm TDMC *Ticketed DMC*
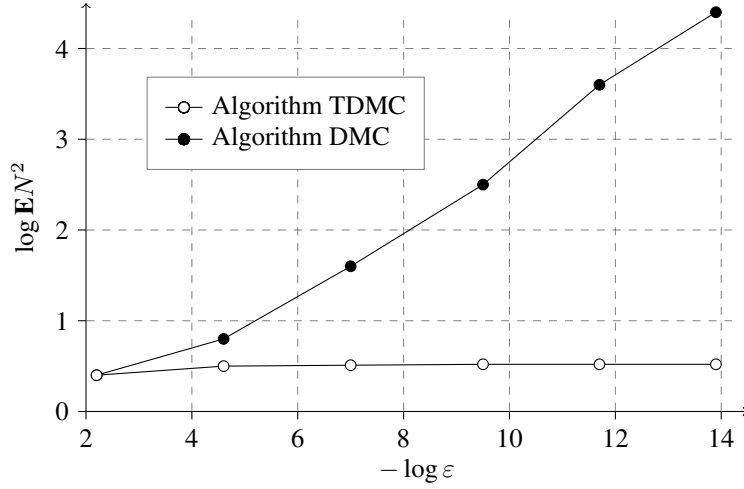
Figure 1: Second moment of N versus stepsize.

1. Begin with M copies $x_0^{(j)} = x_0$. For each $j = 1, \ldots, M$ choose an independent random variable $\theta_0^{(j)} \sim \mathcal{U}(0,1)$.

2. At step k there are N_{t_k} samples $(x_{t_k}^{(j)}, \theta_{t_k}^{(j)})$. Evolve each of the $x_{t_k}^{(j)}$ one step to generate N_{t_k} values

$$\tilde{x}_{t_k}^{(j)} \sim \mathbf{P}(y_{t_{k+1}} \in dx \mid y_{t_k} = x_{t_k}^{(j)}).$$

3. For each $j = 1, \ldots, N_{t_k}$, let

$$P^{(j)} = e^{-\chi(x_{t_k}^{(j)}, \tilde{x}_{t_{k+1}}^{(j)})}. \tag{2.8}$$

   If $P^{(j)} < \theta_{t_k}^{(j)}$ then set

$$N^{(j)} = 0.$$

   If $P^{(j)} \geq \theta_{t_k}^{(j)}$ then set

$$N^{(j)} = \max\{\lfloor P^{(j)} + u^{(j)} \rfloor, 1\}, \tag{2.9}$$

   where $u^{(j)}$ are independent $\mathcal{U}(0,1)$ random variables.

4. For $j = 1, \ldots, N_{t_k}$, if $N^{(j)} > 0$ set

$$x_{t_{k+1}}^{(j,1)} = \tilde{x}_{t_{k+1}}^{(j)} \quad \text{and} \quad \theta_{t_{k+1}}^{(j,1)} = \frac{\theta_{t_k}^{(j)}}{P^{(j)}}$$

   and for $i = 2, \ldots, N^{(j)}$

$$x_{t_{k+1}}^{(j,i)} = \tilde{x}_{t_{k+1}}^{(j)} \quad \text{and} \quad \theta_{t_{k+1}}^{(j,i)} \sim \mathcal{U}((P^{(j)})^{-1}, 1).$$

5. Finally set $N_{t_{k+1}} = \sum_{j=1}^{N_{t_k}} N^{(j)}$ and list the $N_{t_{k+1}}$ vectors $\{x_{t_{k+1}}^{(j,i)}\}$ as $\{x_{t_{k+1}}^{(j)}\}_{j=1}^{N_{t_{k+1}}}$.

6. At time t produce the estimate

$$\widehat{f}_t = \frac{1}{M} \sum_{j=1}^{N_t} f(x_t^{(j)}) \, .$$

Remark 2.2 Both here and in Algorithm DMC, we could have replaced the random variable $\lfloor P^{(j)} + u^{(j)} \rfloor$ by any integer-valued random variable with mean $P^{(j)}$. The choice given here is natural, since it is the one that minimises the variance. From a mathematical perspective, the analysis would have been slightly easier if we chose instead to use Poisson distributed random variables.

We should first establish that this new algorithm "does the job" in the sense that if the steps in Algorithm DMC are replaced by those in Algorithm TDMC, the resulting ensemble of particles still produces an unbiased estimate of the quantity $\langle f \rangle_t$ defined in (2.2). This is the subject of Theorem 4.1 below which establishes that, indeed, for any bounded test function f and any choice of χ,

$$\mathbf{E}\widehat{f}_t = \langle f \rangle_t \, . \tag{2.10}$$

We have already mentioned that a companion article [HW14] is devoted to showing that our modification of DMC, Algorithm TDMC, does not suffer the small ε instability observed in Algorithm DMC. In that asymptotic regime Algorithm TDMC dramatically outperforms the straightforward generalisation of DMC in Algorithm DMC. However, a surprising side-effect of our modification is that Algorithm TDMC is superior to Algorithm DMC in all contexts. In this article we compare Algorithms DMC and TDMC independently of a specific regime (discretisation stepsize, choice of χ, etc).

The key tool in carrying out this comparison is Lemma 4.4 in Section 4. That lemma establishes that by simply randomising all tickets uniformly between 0 and 1 at each step of Algorithm TDMC, one obtains a process that, in law, is identical to the one generated by Algorithm DMC. With this observation in hand we are able to establish in Theorem 4.3 that the estimator produced by Algorithm TDMC always has lower variance than Algorithm DMC, i.e. we prove that

$$\mathbf{var}\,\widehat{f}_t \leq \mathbf{var}^1\,\widehat{f}_t \, , \tag{2.11}$$

holds for any bounded test function f, any underlying Markov chain Y, and any choice of χ. In expression (2.11) we have again distinguished expectations under the rules of Algorithm DMC from all other expectations by a superscript 1.

In comparing Algorithms DMC and TDMC, it is not enough to simply compare the variances of the corresponding estimators: one should also compare their respective costs. Since the dominant contribution to any cost difference in the two algorithms will come from differences in the number of times one must evolve a particle from one time step to the next we compare the expectations of *workload*

$$\mathcal{W}_t = \sum_{t_k \leq t} N_{t_k}$$

under the two rules. However, by (2.10) with $f \equiv 1$ we see that

$$\mathbf{E}^1 \mathcal{W}_t = \mathbf{E}\mathcal{W}_t = M \sum_{t_k \leq t} \mathbf{E} \exp\Big(-\sum_{j=0}^{k-1} \chi(y_{t_j}, y_{t_{j+1}})\Big) \, ,$$

so that the expected cost of both algorithms is the same. In fact, the proof of Theorem 4.3 can be modified slightly to show that the variance of the workload of Algorithm TDMC is always lower than the variance of workload of Algorithm DMC. These remarks leave little room for ambiguity. Algorithm TDMC is more efficient than Algorithm DMC in any setting.

The existence of a continuous-time limit and other robust, small-discretisation parameter characteristics established in [HW14] along with the other features of Algorithm TDMC mentioned above, strongly suggest that Algorithm TDMC has significant potential as a flexible and efficient computational tool. In the next section we provide numerical evidence to that effect.

3 Two examples

In the following subsections we apply our modified DMC algorithm to two simple problems. In Section 3.1 we consider the approximation of a small probability related to the escape of a diffusion from a deep potential well. We will use Algorithm TDMC to bias simulation of the diffusion so that an otherwise rare event (escape from the potential well) becomes common (and then reweight appropriately). We will see that extremely low probability events can be computed with reasonably low relative error. In Section 3.2 we demonstrate the performance of Algorithm TDMC in one of DMCs most common application contexts, particle filtering. We will reconstruct a hidden trajectory of a diffusion, given noisy but very frequent observations. Our comparison shows that the reconstruction produced by the modified method is dramatically more accurate than the reconstruction produced by straightforward generalisation of DMC.

3.1 Rare event simulation

In Quantum Monte Carlo applications, the function χ is (for the most part) specified by the potential V as in (2.3). In other applications however, there may be more freedom in the choice of χ to achieve some specific goal. As already mentioned earlier, the choice $\chi(x, y) = V(y) - V(x)$ turns Algorithm TDMC into an estimator of

$$\langle f \rangle_t = e^{V(x_0)} \mathbf{E}\big(f(y_t)e^{-V(y_t)}\big) \ .$$

The design of a "good" choice of V will be discussed below. By redefining f we find that $e^{-V(x_0)} \widehat{e^V f}_t$ is an unbiased estimate of $\mathbf{E}(f(y_t))$, i.e.

$$e^{-V(x_0)} \mathbf{E} \sum_{j=1}^{N_t} e^{V(x_t^{(j)})} f(x_t^{(j)}) = \mathbf{E}f(y_t) \ ,$$

where the samples $x_t^{(j)}$ are generated as in Algorithm TDMC. This suggests choosing V to minimise the variance of $\widehat{e^V f}_t$. Intuitively this involves choosing V to be smaller in regions where f is significant. The branching step in Algorithm TDMC will create more copies when V decreases, focusing computational resources in regions where V is small.

As an example, consider the overdamped Langevin dynamics

$$dy_t = -\nabla U(y_t)dt + \sqrt{2\gamma}dB_t$$

where y represents the positions of seven two-dimensional particles (i.e. $y \in \mathbb{R}^{14}$) and

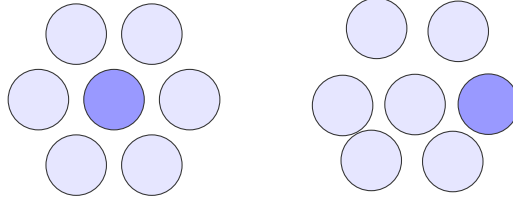$$U(x) = \sum_{i<j} 4\big(\|x_i - x_j\|^{-12} - \|x_i - x_j\|^{-6}\big) \ .$$

Figure 2: (**Left**) The initial condition x^A to generate the results in Table 1. (**Right**) A typical sample of the event $y_2 \in B$.

In this formula x_i represents the position of the ith particle (i.e. $x_i \in \mathbb{R}^2$). In various forms this Lennard-Jones cluster is a standard non-trivial test problem in rare event simulation (see [DBC98]).

After discretising this process by the Euler scheme

$$y_{(k+1)\varepsilon} = y_{k\varepsilon} - \nabla U(y_{k\varepsilon})\,\varepsilon + \sqrt{2\gamma\varepsilon}\,\xi_{k+1}$$

with a stepsize of $\varepsilon = 10^{-4}$ we apply Algorithm TDMC with

$$\chi(x,y) = V(y) - V(x), \qquad V(x) = \frac{\lambda}{kT} \min_{i \geq 2} \left\{ \left\| x_i - \frac{1}{7} \sum_j x_j \right\| \right\}.$$

The properties of the Euler discretisation in the context of rare event simulation are discussed in [VEW14].

Our goal is to compute

$$\mathbf{P}_{x^A}(y_2 \in B), \qquad B = \{x : V(x) < 0.1\},$$

where our initial configuration is given by $x_1^A = (0,0)$ and for $j = 2, 3, \ldots, 7$,

$$x_j^A = \left(\cos \frac{j\pi}{3}, \sin \frac{j\pi}{3} \right)$$

(see Figure 2). Starting from initial configuration $x_0 = x^A$, the particle initially at the centre of the cluster (x_1^A) will typically remain there for a length of time that increases exponentially in γ^{-1}. We are roughly computing the probability that, in 2 units of time, the particle at the centre of the cluster exchanges positions with one of the particles in the outer shell. This probability decreases exponentially in γ^{-1} as $\gamma \to 0$. In this case

$$f(x) = \mathbf{1}_B(x),$$

where $\mathbf{1}_B$ is the indicator function of B. In our simulations, λ is chosen so that the expected number of particles ending in B, $\mathbf{E}\widehat{f}_2$, is (very) roughly 1. The results for several values of the temperature γ are displayed in Table 1.

The workload referenced in Table 1 is the (scaled) expected total number of ∇U evaluations per sample, i.e. the expectation of

$$\varepsilon \mathcal{W}_2 = \varepsilon \sum_{k=1}^{2/\varepsilon} N_{k\varepsilon}.$$

Note that when $V \equiv 0$, Algorithm TDMC reduces to straightforward (brute force) simulation and the workload is exactly 2. Increasing λ results in the creation of more

γ	λ	estimate	workload	$\frac{1}{2}\left(\frac{\text{variance}}{\times\text{workload}}\right)$	brute force variance
0.4	1.9	1.125×10^{-2}	12.90	4.732×10^{-3}	1.112×10^{-2}
0.2	1.3	2.340×10^{-3}	11.64	2.344×10^{-4}	2.335×10^{-3}
0.1	1	7.723×10^{-5}	13.87	7.473×10^{-7}	7.722×10^{-5}
0.05	0.85	9.290×10^{-8}	30.84	1.002×10^{-11}	9.290×10^{-8}
0.025	0.8	1.129×10^{-13}	204.8	1.311×10^{-21}	1.129×10^{-13}

Table 1: Performance of Algorithm TDMC on the Lennard-Jones cluster problem described in Section 3.1 at different temperatures (γ). A measure of the efficiency improvement of Algorithm TDMC over straightforward simulation is obtained by taking the ratio of the last two columns

particles. This has the competing effects of increasing the workload on the one hand but reducing the variance of the resulting estimator on the other. If we let σ^2 denote the variance of one sample run of Algorithm TDMC (i.e. $\mathbf{var}\, e^{-V(x^A)}\widehat{e^V f}_2$ with $M = 1$) and σ_{bf}^2 the variance of the random variable $f(y_2)$, then the number of samples required to compute an estimate with a statistical error of err is $M = \sigma^2/err$ and $M_{bf} = \sigma_{bf}^2/err$ for Algorithm TDMC and brute force respectively. Taking account of the expected cost (scaled by ε) per sample of Algorithm TDMC (reported in the "workload" column) and of brute force simulation (identically 2) we can obtain a comparison of the cost of Algorithm TDMC and brute force by comparing the brute force variance to the product of the variance of the estimate produced by Algorithm TDMC and one-half the workload reported in the fourth column of the table. These quantities are reported in the last two columns of the table. By taking the ratio of the two columns we obtain a measure of the relative cost per degree of accuracy of the two methods. One can see that the speedup provided by Algorithm TDMC becomes dramatic for smaller values of γ. The variance of the brute force estimator is computed from the values in the third column by

$$\text{brute force variance} = \mathbf{P}_{x^A}(y_2 \in B)(1 - \mathbf{P}_{x^A}(y_2 \in B)).$$

Only the estimates in the first two rows of Table 1 were compared against straightforward simulation. Those estimates agreed to the appropriate precision.

The choice of V used in the above simulations is certainly far from optimal. Given the similarities in this rare event context between Algorithm TDMC and the DPR algorithm, it should be straightforward to adapt the results of Dean and Dupuis in [DD11] identifying optimal choices of V in the small γ limit. In fact, we suspect (but do not prove) that, at least when the step-size parameter is small, the optimal choice of V at any temperature is given by the time-dependent function

$$V(k, x) = -\log \mathbf{P}(y_2 \in B \,|\, y_{k\varepsilon} = x).$$

This choice is clearly impractical as it requires knowing the probability that we are trying to compute. Even asymptotic estimates based on taking an appropriate low γ limit of this choice are often not practical so it is worth noting that, even with a rather clumsy choice of V, our scheme is still able to produce impressive results to fairly low temperatures. We fully expect however that without a very carefully chosen V, the cost of achieving a fixed relative accuracy for our scheme will grow exponentially with γ^{-1} as $\gamma \to 0$. This characteristic is common in rare event simulation methods.

3.2   Non-linear filtering

In this section our goal is to reconstruct a sample path of the solution to the stochastic differential equation

$$dy_t^{(1)} = 10(y_t^{(1)} - y_t^{(2)})\, dt + \sqrt{2} dB_t^{(1)}\,, \tag{3.1}$$

$$dy_t^{(2)} = (y_t^{(1)}(28 - y_t^{(3)}) - y_t^{(2)})\, dt + \sqrt{2} dB_t^{(2)}\,, \tag{3.2}$$

$$dy_t^{(3)} = \left(y_t^{(1)} y_t^{(2)} - \frac{8}{3} y_t^{(3)}\right) dt + \sqrt{2} dB_t^{(3)}\,, \tag{3.3}$$

with

$$y_0 = \begin{pmatrix} -5.91652 \\ -5.52332 \\ 24.57231 \end{pmatrix}.$$

The deterministic version of this system is the famous Lorenz 63 chaotic ordinary differential equation [Lor63]. The stochastic version above is commonly used in tests of on-line filtering strategies (see e.g. [MGG94, MTAC12]).

The path of y_t is revealed only through the 3-dimensional noisy observation process

$$dh_t = y_t\, dt + 0.1\, d\tilde{B}_t \tag{3.4}$$

where the 3-dimensional Brownian motion $\tilde{B}_t$ is independent of B_t. Our task is to reconstruct the path of y_t given a realisation of h_t. More precisely, our goal is to sample from (and compute moments of) the conditional distribution of y_t given $\mathcal{F}_t^h$, where $\mathcal{F}^h$ is the filtration generated by h. One can verify that expectations with respect to this conditional distribution can be written as

$$\frac{\mathbf{E}^B\left(f(y_t)\exp\left(-10\int_0^t \langle y_s, dh(s)\rangle - 50\int_0^t \|y_s\|^2 ds\right)\right)}{\mathbf{E}^B \exp\left(-10\int_0^t \langle y_s, dh_s\rangle - 50\int_0^t \|y_s\|^2 ds\right)}\,, \tag{3.5}$$

where the superscript B on the expectations indicates that they are expectations over B_t only and not over $\tilde{B}_t$, i.e. the trajectory of h_t is fixed. We will focus on estimating the mean of this distribution ($f(x) = x$) which we will refer to as the reconstruction of the hidden signal.

As always, we should first discretise the problem. The simplest choice is to again replace (3.1) by its Euler discretisation with parameter ε. The observation process (3.4) can be replaced by

$$h_{(k+1)\varepsilon} - h_{k\varepsilon} \approx y_{k\varepsilon}\, \varepsilon + 0.1\sqrt{\varepsilon}\, \eta_{k+1} \tag{3.6}$$

where the η_k are independent Gaussian random variables with mean 0 and identity covariance. We again set $\varepsilon = 10^{-4}$. With this choice for h_t, it is equivalent to assume that the observation process is the sequence of increments $h_{(k+1)\varepsilon} - h_{k\varepsilon}$ instead of H itself. To emphasise that we will be conditioning on the values of $h_{(k+1)\varepsilon} - h_{k\varepsilon}$ and not computing expectations with respect to these variables we will use the notation $\triangle_k$ to denote a specific realisation of $h_{(k+1)\varepsilon} - h_{k\varepsilon}$. In Figure 3.2 we plot a trajectory of the resulting discrete process y and observations $h_{(k+1)\varepsilon} - h_{k\varepsilon}$.

Note that given a particular state $y_{k\varepsilon} = x$, the probability of observing $h_{(k+1)\varepsilon} - h_{k\varepsilon} = \triangle$ is

$$\exp\left(-\frac{\|x\varepsilon - \triangle\|^2}{0.02\varepsilon}\right)$$
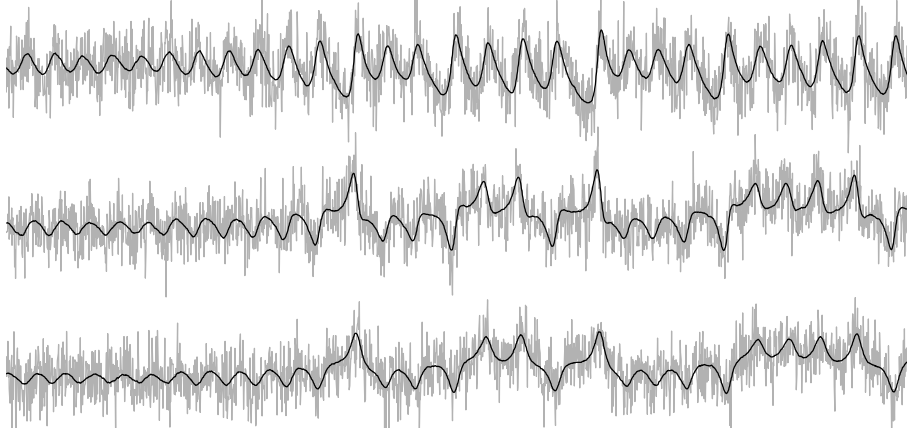
Figure 3: Hidden signal (black), together with the observed signal in gray.

and expectations of the discretised process y given the $h_{(k+1)\varepsilon} - h_{k\varepsilon} = \triangle_k$ observations can be computed by the formula

$$Z_k^{-1} \mathbf{E}\left(y_{k\varepsilon} \exp\left(-\sum_{j=1}^{k} \frac{\|y_{j\varepsilon}\,\varepsilon - \triangle_j\|^2}{0.02\varepsilon}\right)\right), \qquad (3.7)$$

where the normalisation constant Z_k is given by

$$Z_k = \mathbf{E} \exp\left(-\sum_{j=1}^{k} \frac{\|y_{j\varepsilon}\,\varepsilon - \triangle_j\|^2}{0.02\varepsilon}\right).$$

In the small ε limit, formula (3.7) (with $k = t/\varepsilon$) indeed converges to (3.5) (see [Cri11]).

Expression (3.7) suggests applying Algorithms DMC or TDMC with

$$\chi(x, y) = \frac{\|y\varepsilon - \triangle_{k+1}\|^2}{0.02\varepsilon}$$

at each time step. Below it will be convenient to consider the resulting choice of $P^{(i)}$,

$$P^{(i)} = \exp\left(-\frac{\|\tilde{x}_{(k+1)\varepsilon}^{(i)}\varepsilon - \triangle_{k+1}\|^2}{0.02\varepsilon}\right)$$

in Algorithm TDMC. With this choice, the expected number of particles at the kth step would be

$$M\mathbf{E} \exp\left(-\sum_{j=1}^{k} \frac{\|y_{j\varepsilon}\,\varepsilon - \triangle_j\|^2}{0.02\varepsilon}\right),$$

a quantity that my decay very rapidly with k. Since our goal is to compute conditional expectations we are free to normalise $P^{(i)}$ by any constant. A tempting choice is

$$P^{(i)} = \frac{Z_k}{Z_{k+1}} \exp\left(-\frac{\|\tilde{x}_{(k+1)\varepsilon}^{(i)}\varepsilon - \triangle_{k+1}\|^2}{0.02\varepsilon}\right)$$

which would result in $\mathbf{E}N_{k\varepsilon} = M$ for all k. Unfortunately we do not know the conditional expectation in the denominator of this expression. However, for a large number of particles $N_{k\varepsilon}$ we have the approximation

$$\frac{1}{N_{k\varepsilon}} \sum_{i=1}^{N_{k\varepsilon}} \exp\left(-\frac{\|\tilde{x}^{(i)}_{(k+1)\varepsilon}\varepsilon - \triangle_{k+1}\|^2}{0.02\varepsilon}\right)$$

$$\approx \mathbf{E}\left(\exp\left(-\frac{\|y_{(k+1)\varepsilon}\varepsilon - \triangle_{k+1}\|^2}{0.02\varepsilon}\right) \,\Big|\, \triangle_1, \ldots, \triangle_k\right) = \frac{Z_{k+1}}{Z_k}.$$

This suggests using

$$P^{(i)} = \frac{\exp\left(-\frac{\|\tilde{x}^{(i)}_{(k+1)\varepsilon}\varepsilon - \triangle_{k+1}\|^2}{0.02\varepsilon}\right)}{\frac{1}{N_{k\varepsilon}} \sum_{l=1}^{N_{k\varepsilon}} \exp\left(-\frac{\|\tilde{x}^{(l)}_{(k+1)\varepsilon}\varepsilon - \triangle_{k+1}\|^2}{0.02\varepsilon}\right)}, \tag{3.8}$$

which will guaranty that $\mathbf{E}N_{k\varepsilon} = M$ for all k.

In fact, in practical applications it is important to have even more control over the population size. Stronger control on the number of particles can be achieved in many ways (e.g. by resampling strategies [dDG05, Liu02]). We choose to make the simple modification

$$P^{(i)} = M \frac{\exp\left(-\frac{\|\tilde{x}^{(i)}_{(k+1)\varepsilon}\varepsilon - \triangle_{k+1}\|^2}{0.02\varepsilon}\right)}{\sum_{l=1}^{N_{k\varepsilon}} \exp\left(-\frac{\|\tilde{x}^{(l)}_{(k+1)\varepsilon}\varepsilon - \triangle_{k+1}\|^2}{0.02\varepsilon}\right)}, \tag{3.9}$$

which results in an expected number of particles at step $k + 1$ of M independently of the details of the step k ensemble. Formula (3.9) depends on the entire ensemble of walkers and does not fit strictly within the confines of Algorithms DMC and TDMC. This choice will lead to estimators with a small, M-dependent, bias. All sequential Monte Carlo strategies with ensemble population control for computing conditional expectations that we are aware of suffer a similar bias.

The results of a test of Algorithm TDMC with this choice of $P^{(i)}$ are presented in Figure 5 for $M = 10$. The true trajectory of y is drawn as a dotted line, while our reconstruction is drawn as a solid line. Note that the dotted line is nearly completely hidden behind the solid line, indicating an accurate reconstruction. In Figure 4 we show the results of the same test with Algorithm TDMC replaced by Algorithm DMC. The method obtain in this way from Algorithm DMC is very similar to a standard particle filter with the common residual resampling step (see [Liu02]).

With our small choice of ε, one might expect that the number of particles generated by Algorithm DMC would explode or vanish almost instantly. Our choice of $P^{(i)}$ prevents large fluctuations in N. We expect instead that the number of truly distinguishable particles in the ensemble (in the sense that they are not very nearly in the same positions) will drop to 1 instantly, leading to a very low resolution scheme and a poor reconstruction. This is supported by the results shown in Figure 4.

The fact that we obtain such a good reconstruction in Figure 5 with only 10 particles indicates that this is not a particularly challenging filtering problem. Challenging filtering problems and more involved methods for dealing with them are discussed in [VEW]. The purpose of this example is to demonstrate that by simply removing unnecessary variance from a particle filter (via Algorithm TDMC) one can improve
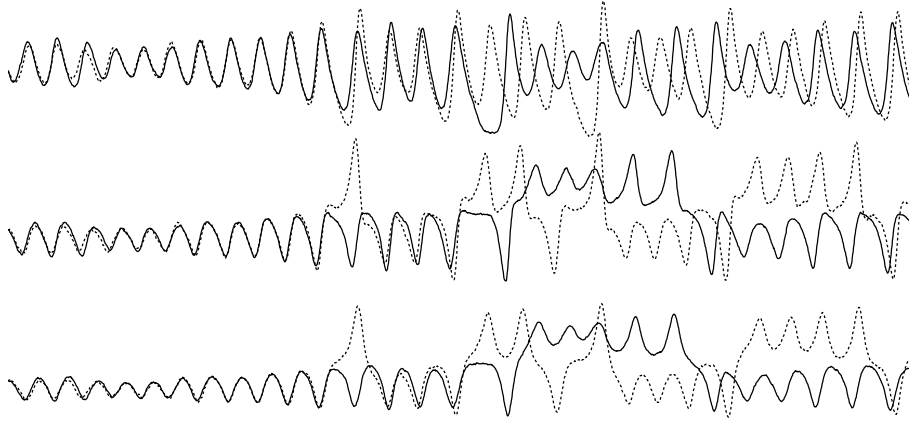
Figure 4: Reconstruction of the three components of the signal using Algorithm DMC. The hidden signal is shown as a dotted line.
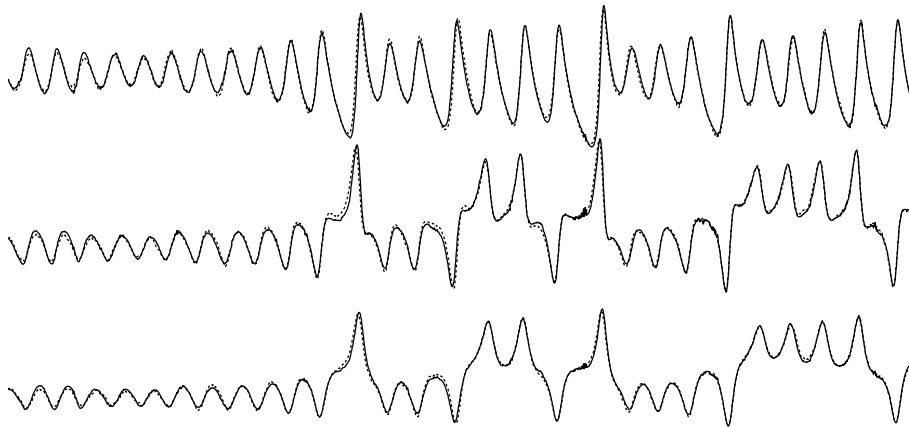

Figure 5: Reconstruction of the three components of the signal using Algorithm TDMC. The hidden signal is shown as a dotted line.

performance. This improvement is illustrated dramatically in our small ε setting. In practice it is advisable to carry out the copying and killing steps in Algorithm TDMC less frequently, accumulating particle weights in the intervening time intervals. Nonetheless, any variance in the resulting estimate will be unnecessarily amplified if one uses a method based on Algorithm DMC instead of Algorithm TDMC.

4 Bias and comparison of Algorithms DMC and TDMC

In this section we establish several general properties of Algorithms DMC and TDMC. In contrast to later sections we do not focus here on any specific choice of the Markov process y or the function χ. For simplicity in our displays we will assume in this section that y is time-homogenous, i.e. that its transition probability distribution is independent of time. As above we use $\mathbf{E}^1$ and $\mathbf{E}$ to denote expectations under the rules of Algorithms DMC and TDMC respectively. In the proof of each result we will assume that $M = 1$ in Algorithms DMC and TDMC. The results follow for $M > 1$ by the independence of the processes generated from each copy of the initial configuration. To keep the formulas slightly more compact and because the results in these section do not pertain to the continuous time limit, we will replace our subscript t_k notation with a subscript k (and t is now a non-negative integer).

Our first main result in this section establishes that Algorithms DMC and TDMC are unbiased in the following sense.

Theorem 4.1 *Algorithm TDMC produces an unbiased estimator, namely*

$$\mathbf{E}\widehat{f_t} = \mathbf{E}\left(f(y_t) \exp\left(-\sum_{k=0}^{t-1} \chi(y_k, y_{k+1})\right)\right) . \tag{4.1}$$

The expression also holds with $\mathbf{E}$ replaced by $\mathbf{E}^1$.

The proof of Theorem 4.1 is similar to the proof of Theorem 3.3 in [DD11] and requires the following lemma.

Lemma 4.2 *For any bounded, non-negative function F on $\mathbb{R}^d \times \mathbb{R}$ we have*

$$\mathbf{E}\left(\sum_{j=1}^{N_1} F(x_1^{(j)}, \theta_1^{(j)}) \,\Big|\, x_0, \tilde{x}_1^{(1)}\right) = e^{-\chi(x_0, \tilde{x}_1^{(1)})} \int_0^1 F(\tilde{x}_1^{(1)}, u)\, du . \tag{4.2}$$

The expression also holds with $\mathbf{E}$ replaced by $\mathbf{E}^1$.

Proof. From the description of the algorithm and, in particular the fact that given x_0, and $\tilde{x}_1^{(1)}$, the tickets are all independent of each other, we obtain

$$\mathbf{E}\left(\sum_{j=1}^{N_1} F(x_1^{(j)}, \theta_1^{(j)}) \,\Big|\, x_0, \tilde{x}_1^{(1)}\right) = \int_0^1 F(\tilde{x}_1^{(1)}, ue^{\chi(x_0, \tilde{x}_1^{(1)})}) \mathbf{1}_{\left(e^{-\chi(x_0, \tilde{x}_1^{(1)})} \geq u\right)} du$$

$$+ \frac{e^{-\chi(x_0, \tilde{x}_1^{(1)})} - 1}{1 - e^{\chi(x_0, \tilde{x}_1^{(1)})}} \int_{e^{\chi(x_0, \tilde{x}_1^{(1)})}}^1 F(\tilde{x}_1^{(1)}, u)\, du \, \mathbf{1}_{\left(\chi(x_0, \tilde{x}_1^{(1)}) \leq 0\right)} .$$

The first term on the right can be rewritten as

$$e^{-\chi(x_0,\tilde{x}_1^{(1)})} \int_0^{e^{\chi(x_0,\tilde{x}_1^{(1)})}} F(\tilde{x}_1^{(1)}, u)\, du\, \mathbf{1}_{\left(\chi(x_0,\tilde{x}_1^{(1)})\leq 0\right)}$$

$$+ e^{-\chi(x_0,\tilde{x}_1^{(1)})} \int_0^1 F(\tilde{x}_1^{(1)}, u)\, du\, \mathbf{1}_{\left(\chi(x_0,\tilde{x}_1^{(1)})>0\right)}$$

while the second is

$$e^{-\chi(x_0,\tilde{x}_1^{(1)})} \int_{e^{\chi(x_0,\tilde{x}_1^{(1)})}}^1 F(\tilde{x}_1^{(1)}, u)\, du\, \mathbf{1}_{\left(\chi(x_0,\tilde{x}_1^{(1)})\leq 0\right)} \cdot$$

Combining terms we obtain

$$\mathbf{E}\left(\sum_{j=1}^{N_1} F(x_1^{(j)},\theta_1^{(j)}) \,\bigg|\, x_0, \tilde{x}_1^{(1)}\right) = e^{-\chi(x_0,\tilde{x}_1^{(1)})} \int_0^1 F(\tilde{x}_1^{(1)}, u)\, du\, \mathbf{1}_{\left(\chi(x_0,\tilde{x}_1^{(1)})\leq 0\right)}$$

$$+ e^{-\chi(x_0,\tilde{x}_1^{(1)})} \int_0^1 F(\tilde{x}_1^{(1)}, u)\, du\, \mathbf{1}_{\left(\chi(x_0,\tilde{x}_1^{(1)})>0\right)}$$

$$= e^{-\chi(x_0,\tilde{x}_1^{(1)})} \int_0^1 F(\tilde{x}_1^{(1)}, u)\, du\, ,$$

which establishes (4.2). A similar (but simpler) argument proves the result for $\mathbf{E}$ replaced by $\mathbf{E}^1$. $\qquad\square$

With Lemma 4.2 in hand we are ready to prove Theorem 4.1.

Proof of Theorem 4.1. First notice that by Lemma (4.2), we have the identity

$$\mathbf{E}\sum_{j=1}^{N_1} F(x_1^{(j)},\theta_1^{(j)}) = \mathbf{E}\left(e^{-\chi(x_0,y_1)} \int_0^1 F(y_1, u)\, du\right), \qquad (4.3)$$

for any function $F(x,\theta)$. In particular, the required identity holds for $k = 1$ by setting $F(x,\theta) = f(x)$.

Now we proceed by induction, assuming that expression (4.1) holds for step $k-1$ and prove that the relation also holds at k. Notice that, by the Markov property of the process,

$$\mathbf{E}\sum_{j=1}^{N_k} f(x_k^{(j)}) = \mathbf{E}\left(\sum_{j=1}^{N_1} \mathbf{E}_{x_1^{(j)},\theta_1^{(j)}}\left(\sum_{j=1}^{N_{k-1}} f(x_{k-1}^{(j)})\right)\right).$$

Setting

$$F(x,\theta) = \mathbf{E}_{x,\theta} \sum_{j=1}^{N_{k-1}} f(x_{k-1}^{(j)})$$

in expression (4.3), we obtain

$$\mathbf{E}\sum_{j=1}^{N_k} f(x_k^{(j)}) = \mathbf{E}\sum_{j=1}^{N_1} F(x_1^{(j)},\theta_1^{(j)})$$

$$= \mathbf{E}\left(e^{-\chi(x_0,y_1)} \int_0^1 \mathbf{E}_{y_1,u}\left(\sum_{j=1}^{N_{k-1}} f(x_{k-1}^{(j)})\right) du\right).$$

According to the rule for generating the initial ticket we have that

$$\int_0^1 \mathbf{E}_{y_1,u}\left(\sum_{j=1}^{N_{k-1}} f(x_{k-1}^{(j)})\right) du = \mathbf{E}_{y_1} \sum_{j=1}^{N_{k-1}} f(x_{k-1}^{(j)})$$

and we have therefore shown that

$$\mathbf{E}\sum_{j=1}^{N_k} f(x_k^{(j)}) = \mathbf{E}\left(e^{-\chi(x_0,y_1)}\mathbf{E}_{y_1}\left(\sum_{j=1}^{N_{k-1}} f(x_{k-1}^{(j)})\right)\right).$$

We can conclude, by our induction hypothesis, that

$$\mathbf{E}\sum_{j=1}^{N_k} f(x_k^{(j)}) = \mathbf{E}\left(e^{-\chi(x_0,y_1)}\mathbf{E}_{y_1}\left(f(y_{k-1})e^{-\sum_{k=0}^{t-2}\chi(y_k,y_{k+1})}\right)\right)$$

$$= \mathbf{E}\left(f(y_t)\exp\left(-\sum_{k=0}^{t-1}\chi(y_k,y_{k+1})\right)\right),$$

and the proof is complete. A similar argument proves the result for $\mathbf{E}$ replaced by $\mathbf{E}^1$. $\square$

The second main result of this section compares the variance of the estimators generated by Algorithms DMC and TDMC. We have that

Theorem 4.3 *For all bounded functions f, one has the inequality*

$$\mathbf{var}\,\widehat{f_t} \le \mathbf{var}^1\widehat{f_t}.$$

The inequality is strict when f is strictly positive.

The following straightforward observation is an important tool in comparing Algorithms DMC and TDMC.

Lemma 4.4 *After replacing the rules*

$$\theta_{k+1}^{(j,1)} = \frac{\theta_k^{(j)}}{P^{(j)}} \quad and \quad \theta_{k+1}^{(j,i)} \sim \mathcal{U}((P^{(j)})^{-1},1) \quad for\ i > 1\,,$$

in Step 4 of Algorithm TDMC by

$$\theta_{k+1}^{(j,i)} \sim \mathcal{U}(0,1) \quad for\ all\ i\,, \tag{4.4}$$

the process generated by Algorithm TDMC becomes identical in law to the process generated by Algorithm DMC.

Remark 4.5 Note that the resampling of the tickets in (4.4) applies to *all* particles, including $i = 1$.

As with the proof of Theorem 4.1, the proof of Theorem 4.3 is inductive. The most cumbersome component of that argument is obtaining a comparison of cross terms that appear when expanding the variance of the estimators produced by Algorithms DMC and TDMC. This is encapsulated in the following lemma.

Lemma 4.6 *Let F be any non-negative bounded function of $\mathbb{R}^d \times \mathbb{R}$, which is decreasing in its second argument. Then*

$$\mathbf{E}\left(\sum_{j=1}^{N_1}\sum_{i\neq j}^{N_1} F(x_1^{(j)}, \theta_1^{(j)})F(x_1^{(i)}, \theta_1^{(i)})\right) \leq \mathbf{E}^1\left(\sum_{j=1}^{N_1}\sum_{i\neq j}^{N_1} F(x_1^{(j)}, \theta_1^{(j)})F(x_1^{(i)}, \theta_1^{(i)})\right),$$

and the bound is strict when, for each x, $F(x, \theta)$ is a strictly decreasing function of θ.

Proof. First note that $x_1^{(j)} = \tilde{x}_1^{(1)}$ for each j. Furthermore, conditioned on x_0, $\tilde{x}_1^{(1)}$, and N_1, the tickets are independent of each other and for $j \geq 2$ they are identically distributed (for Algorithm DMC they are identically distributed for $j \geq 1$). These facts imply that, for the new scheme,

$$\mathbf{E}\left(\sum_{j=1}^{N_1}\sum_{i\neq j}^{N_1} F(\tilde{x}_1^{(1)}, \theta_1^{(j)})F(\tilde{x}_1^{(1)}, \theta_1^{(i)})\,\Big|\,\tilde{x}_1^{(1)}\right) =$$

$$\mathbf{E}\left(2\,\mathbf{1}_{(N_1>1)}(N_1-1)\,|\,\tilde{x}_1^{(1)}\right)\mathbf{E}\left(F(\tilde{x}_1^{(1)}, \theta_1^{(1)})\,|\,\tilde{x}_1^{(1)}\right)\mathbf{E}\left(F(\tilde{x}_1^{(1)}, \theta_1^{(2)})\,|\,\tilde{x}_1^{(1)}\right)$$

$$+ \mathbf{E}\left(\mathbf{1}_{(N_1>1)}(N_1-1)(N_1-2)\,|\,\tilde{x}_1^{(1)}\right)\mathbf{E}\left(F(\tilde{x}_1^{(1)}, \theta_1^{(2)})\,|\,\tilde{x}_1^{(1)}\right)^2. \quad (4.5)$$

For Algorithm DMC the tickets $\theta_1^{(j)}$ are i.i.d. conditioned on $\tilde{x}_1^{(1)}$ so that

$$\mathbf{E}^1\left(\sum_{j=1}^{N_1}\sum_{i\neq j}^{N_1} F(\tilde{x}_1^{(1)}, \theta_1^{(j)})F(\tilde{x}_1^{(1)}, \theta_1^{(i)})\,\Big|\,\tilde{x}_1^{(1)}\right) =$$

$$\mathbf{E}^1\left(N_1(N_1-1)\,|\,\tilde{x}_1^{(1)}\right)\mathbf{E}^1\left(F(\tilde{x}_1^{(1)}, \theta_1^{(1)})\,|\,\tilde{x}_1^{(1)}\right)^2. \quad (4.6)$$

Let

$$P = e^{-\chi(x_0, \tilde{x}_1^{(1)})}.$$

Note that on the set $\{P \leq 1\}$, we have $N_1 \leq 1$, so that expressions (4.5) and (4.6) vanish. On the set $\{P > 1\}$, we have

$$\mathbf{E}\left(F(\tilde{x}_1^{(1)}, \theta_1^{(2)})\,|\,\tilde{x}_1^{(1)}\right) = \frac{1}{1-P^{-1}}\int_{P^{-1}}^1 F(\tilde{x}_1^{(1)}, u)\,du,$$

$$\mathbf{E}\left(F(\tilde{x}_1^{(1)}, \theta_1^{(1)})\,|\,\tilde{x}_1^{(1)}\right) = P\int_0^{P^{-1}} F(\tilde{x}_1^{(1)}, u)\,du,$$

and

$$\mathbf{E}^1\left(F(\tilde{x}_1^{(1)}, \theta_1^{(1)})\,|\,\tilde{x}_1^{(1)}\right) = \int_0^1 F(\tilde{x}_1^{(1)}, u)\,du,$$

so that

$$\mathbf{E}^1\left(F(\tilde{x}_1^{(1)}, \theta_1^{(1)})\,|\,\tilde{x}_1^{(1)}\right) = \frac{1}{P}\mathbf{E}\left(F(\tilde{x}_1^{(1)}, \theta_1^{(1)})\,|\,\tilde{x}_1^{(1)}\right) + \frac{(P-1)}{P}\mathbf{E}\left(F(\tilde{x}_1^{(1)}, \theta_1^{(2)})\,|\,\tilde{x}_1^{(1)}\right).$$

In other words, if we set

$$A = \mathbf{E}\left(F(\tilde{x}_1^{(1)}, \theta_1^{(2)})\,|\,\tilde{x}_1^{(1)}\right)$$

and

$$B = \mathbf{E}\left(F(\tilde{x}_1^{(1)}, \theta_1^{(1)})\,|\,\tilde{x}_1^{(1)}\right) - \mathbf{E}\left(F(\tilde{x}_1^{(1)}, \theta_1^{(2)})\,|\,\tilde{x}_1^{(1)}\right),$$

we have the identity

$$\mathbf{E}^1\left(F(\tilde{x}_1^{(1)}, \theta_1^{(1)}) \,|\, \tilde{x}_1^{(1)}\right) = A + \frac{1}{P}B.$$

In Algorithm TDMC, $\theta_1^{(1)} \le \theta_1^{(2)}$ so that, if F is a decreasing function of θ, then

$$B \ge 0.$$

Now note that the variable N_1 is the same for both algorithms and is given explicitly by the formula

$$N_1 = \mathbf{1}_{\left(\theta_1^{(1)} \le e^{-\chi(x_0, \tilde{x}_1^{(1)})} \le 1\right)} + \mathbf{1}_{\left(\chi(x_0, \tilde{x}_1^{(1)}) < 0\right)}$$
$$\times \left(\left\lfloor e^{-\chi(x_0, \tilde{x}_1^{(1)})} \right\rfloor + \mathbf{1}_{\left(U < e^{-\chi(x_0, \tilde{x}_1^{(1)})} - \left\lfloor e^{-\chi(x_0, \tilde{x}_1^{(1)})} \right\rfloor\right)}\right),$$

where U is a uniform random variable in $[0,1]$. Using this formula we can compute the expectations involving N_1 explicitly. Defining

$$R = P - \lfloor P \rfloor,$$

on the event $\{P > 1\}$ we have that

$$\mathbf{E}^1\left(N_1(N_1 - 1) \,|\, \tilde{x}_1^{(1)}\right) = (P - R)(P - 1 + R),$$
$$\mathbf{E}\left(\mathbf{1}_{(N_1 > 1)}(N_1 - 1) \,|\, \tilde{x}_1^{(1)}\right) = P - 1,$$

and

$$\mathbf{E}\left(\mathbf{1}_{(N_1 > 1)}(N_1 - 1)(N_1 - 2) \,|\, \tilde{x}_1^{(1)}\right) = (P - 1 - R)(P - 2 + R).$$

The difference of terms (4.6) and (4.5), which vanishes on $P < 2$, can now be written as

$$2(P - 1)AB - 2(P - R)(P - 1 + R)AB\frac{1}{P} - (P - R)(P - 1 + R)B^2\frac{1}{P^2}$$

on the set $\{P \ge 2\}$. Recalling that $B \ge 0$ (we can drop the last term) and rearranging we see that this expression is bounded above by

$$2\frac{AB}{P}(P(P - 1) - (P - R)(P - 1 + R)). \tag{4.7}$$

Note that since $0 \le R \le 1$ we have $P - 1 \le P - 1 + R \le P - R \le P$. The difference in the parenthesis in (4.7) is the difference in the area of two squares with equal perimeter, the second of which (in the sense of the last sequence of inequalities) is closer to square. The difference is therefore non-positive. All bounds are strict whenever, for each x, $F(x, \theta)$ is a strictly decreasing function of θ. □

Finally we complete the proof that the variance of the estimator generated by Algorithm TDMC is bounded above by the variance of the estimator generated by Algorithm DMC.

Proof of Theorem 4.3. By Theorem 4.1 we have that

$$\mathbf{E}\widehat{f_t} = \mathbf{E}^1 \widehat{f_t} \,,$$

so it suffices to show that

$$\mathbf{E}(\widehat{f_t})^2 \leq \mathbf{E}^1(\widehat{f_t})^2 \,. \tag{4.8}$$

Furthermore, since the variance does not change by adding a constant to f we can, and will from now on, assume that $f(x) \geq 0$ for every x. In order to prove (4.8), we will proceed by induction. Since the random variables N_1 and $\tilde{x}_1^{(1)}$ are the same in both schemes, the result is true for $k = 1$. Now assume that (4.8) holds through step $k - 1$. We will show that

$$\mathbf{E}\Big(\sum_{j=1}^{N_k} f(x_k^{(j)})\Big)^2 \leq \mathbf{E}^1\Big(\sum_{j=1}^{N_k} f(x_k^{(j)})\Big)^2 \,. \tag{4.9}$$

Observe that we can write

$$\sum_{j=1}^{N_k} f(x_k^{(j)}) = \sum_{j=1}^{N_1} \sum_{i=1}^{N_{1,k}^{(j)}} f(x_{1,k}^{(j,i)}) \,,$$

where we used $N_{1,k}^{(j)}$ to denote the number of particles at time k whose ancestor at time 1 is $x_1^{(j)}$. The descendants of $x_1^{(j)}$ are enumerated by $x_{1,k}^{(j,i)}$ for $i = 1, 2, \ldots, N_{1,k}^{(j)}$.

By the conditional independence of the descendants of the particles at time k when conditioned on x_0, $\tilde{x}_1^{(1)}$, and N_1, we have that

$$\mathbf{E}\Big(\sum_{j=1}^{N_k} f(x_k^{(j)})\Big)^2 = \mathbf{E}\Big(\sum_{j=1}^{N_1} \mathbf{E}_{x_1^{(j)},\theta_1^{(j)}}\Big(\sum_{i=1}^{N_{k-1}} f(x_{k-1}^{(i)})\Big)^2\Big)$$

$$+ \mathbf{E}\Big(\sum_{j=1}^{N_1} \sum_{i \neq j}^{N_1} \mathbf{E}_{x_1^{(j)},\theta_1^{(j)}}\Big(\sum_{l=1}^{N_{k-1}} f(x_{k-1}^{(l)})\Big)$$

$$\times \mathbf{E}_{x_1^{(i)},\theta_1^{(i)}}\Big(\sum_{l=1}^{N_{k-1}} f(x_{k-1}^{(l)})\Big)\Big) \,. \tag{4.10}$$

An identical expression (with $\mathbf{E}$ replaced by $\mathbf{E}^1$) holds for Algorithm DMC. Setting

$$F_1(x,\theta) = \mathbf{E}_{x,\theta} \sum_{l=1}^{N_{k-1}} f(x_{k-1}^{(l)}) \,,$$

and

$$F_2(x,\theta) = \mathbf{E}_{x,\theta}\Big(\sum_{i=1}^{N_{k-1}} f(x_{k-1}^{(i)})\Big)^2 \,,$$

we can rewrite the last identity in the case of Algorithm TDMC as

$$\mathbf{E}\Big(\sum_{j=1}^{N_k} f(x_k^{(j)})\Big)^2 = \mathbf{E}\sum_{j=1}^{N_1} F_2(x_1^{(j)}, \theta_1^{(j)}) + \mathbf{E}\sum_{j=1}^{N_1} \sum_{i \neq j}^{N_1} F_1(x_1^{(j)}, \theta_1^{(j)}) F_1(x_1^{(i)}, \theta_1^{(i)}).$$

In the case of Algorithm DMC with

$$F_1^0(x, \theta) = \mathbf{E}_{x,\theta}^1 \sum_{l=1}^{N_{k-1}} f(x_{k-1}^{(l)})$$

and

$$F_2^0(x, \theta) = \mathbf{E}_{x,\theta}^1 \left(\sum_{i=1}^{N_{k-1}} f(x_{k-1}^{(i)}) \right)^2 ,$$

we have

$$\mathbf{E}^1 \left(\sum_{j=1}^{N_k} f(x_k^{(j)}) \right)^2 = \mathbf{E}^1 \sum_{j=1}^{N_1} F_2^0(x_1^{(j)}, \theta_1^{(j)}) + \mathbf{E}^1 \sum_{j=1}^{N_1} \sum_{i \neq j}^{N_1} F_1^0(x_1^{(j)}, \theta_1^{(j)}) F_1^0(x_1^{(i)}, \theta_1^{(i)}) .$$

Note that Lemma 4.2 implies that

$$\mathbf{E} \sum_{j=1}^{N_1} F_2(x_1^{(j)}, \theta_1^{(j)}) = \mathbf{E}^1 \sum_{j=1}^{N_1} F_2(x_1^{(j)}, \theta_1^{(j)}).$$

Under the rules of Algorithm DMC, conditioned on x_0 and $\tilde{x}_1^{(1)}$, the ticket $\theta_1^{(j)}$ is independent of N_1. So we can write the last equality as

$$\mathbf{E} \sum_{j=1}^{N_1} F_2(x_1^{(j)}, \theta_1^{(j)}) = \mathbf{E}^1 \left(\sum_{j=1}^{N_1} \mathbf{E}_{x_1^{(j)}} \left(F_2(x_1^{(j)}, \theta_1^{(j)}) \right) \right).$$

By our inductive hypothesis we then have that

$$\mathbf{E} \sum_{j=1}^{N_1} F_2(x_1^{(j)}, \theta_1^{(j)}) \leq \mathbf{E}^1 \left(\sum_{j=1}^{N_1} \mathbf{E}_{x_1^{(j)}} \left(F_2^0(x_1^{(j)}, \theta_1^{(j)}) \right) \right)$$

Appealing again to the conditional independence of $\theta_1^{(j)}$ and N_1 under Algorithm DMC we have that

$$\mathbf{E} \sum_{j=1}^{N_1} F_2(x_1^{(j)}, \theta_1^{(j)}) \leq \mathbf{E}^1 \sum_{j=1}^{N_1} F_2^0(x_1^{(j)}, \theta_1^{(j)}) .$$

We now move on to the second term in (4.10). It follows from the definition of Algorithm TDMC that the function F_1 is strictly decreasing in θ, so that Lemma 4.6 yields

$$\mathbf{E} \left(\sum_{j=1}^{N_1} \sum_{i \neq j}^{N_1} F_1(x_1^{(j)}, \theta_1^{(j)}) F_1(x_1^{(i)}, \theta_1^{(i)}) \right) \leq \mathbf{E}^1 \left(\sum_{j=1}^{N_1} \sum_{i \neq j}^{N_1} F_1(x_1^{(j)}, \theta_1^{(j)}) F_1(x_1^{(i)}, \theta_1^{(i)}) \right).$$

Under the rules of Algorithm DMC, conditional on x_0 and $\tilde{x}_1^{(1)}$, the $\theta_1^{(1)}$ are all independent of each other and of N_1. Therefore we can integrate over the tickets to obtain

$$\mathbf{E} \left(\sum_{j=1}^{N_1} \sum_{i \neq j}^{N_1} F_1(x_1^{(j)}, \theta_1^{(j)}) F_1(x_1^{(i)}, \theta_1^{(i)}) \right)$$

$$\leq \mathbf{E}^1 \left(\sum_{j=1}^{N_1} \sum_{i \neq j}^{N_1} \mathbf{E}_{x_1^{(j)}} \left(F_1(x_1^{(j)}, \theta_1^{(j)}) \right) \mathbf{E}_{x_1^{(i)}} \left(F_1(x_1^{(i)}, \theta_1^{(i)}) \right) \right).$$

By Theorem 4.1 we have that

$$\mathbf{E}_{x_1^{(j)}} \sum_{l=1}^{N_{k-1}} f(x_{k-1}^{(l)}) = \mathbf{E}_{x_1^{(j)}}^0 \sum_{l=1}^{N_{k-1}} f(x_{k-1}^{(l)})$$

or

$$\mathbf{E}_{x_1^{(j)}} F_1(x_1^{(j)}, \theta_1^{(j)}) = \mathbf{E}_{x_1^{(j)}}^1 F_1^0(x_1^{(j)}, \theta_1^{(j)}) \,.$$

This yields

$$\mathbf{E}\bigg(\sum_{j=1}^{N_1} \sum_{i \neq j}^{N_1} F_1(x_1^{(j)}, \theta_1^{(j)}) F_1(x_1^{(i)}, \theta_1^{(i)}) \bigg)$$
$$\leq \mathbf{E}^1 \bigg(\sum_{j=1}^{N_1} \sum_{i \neq j}^{N_1} \mathbf{E}_{x_1^{(j)}}^1 \bigg(F_1^0(x_1^{(j)}, \theta_1^{(j)}) \bigg) \mathbf{E}_{x_1^{(i)}}^1 \bigg(F_1^0(x_1^{(i)}, \theta_1^{(i)}) \bigg) \bigg).$$

Reinserting the tickets we obtain

$$\mathbf{E}\bigg(\sum_{j=1}^{N_1} \sum_{i \neq j}^{N_1} F_1(x_1^{(j)}, \theta_1^{(j)}) F_1(x_1^{(i)}, \theta_1^{(i)}) \bigg) \leq \mathbf{E}^1 \bigg(\sum_{j=1}^{N_1} \sum_{i \neq j}^{N_1} F_1^0(x_1^{(j)}, \theta_1^{(j)}) F_1^0(x_1^{(i)}, \theta_1^{(i)}) \bigg) \,,$$

which completes the proof. $\qquad\square$

References

[AFRtW06] R. J. ALLEN, D. FRENKEL, and P. REIN TEN WOLDE. Simulating rare events in equilibrium or non equilibrium stochastic systems. *J. Chem. Phys.* **124**, (2006), 024102.

[And75] J. ANDERSON. A random-walk simulation of the Schrödinger equation: H_3^+. *J. Chem. Phys.* **63**, no. 4, (1975), 1499–1503.

[Buc04] J. BUCKLEW. *Introduction to Rare Event Simulation*. Springer, 2004.

[CA80] D. CEPERLEY and B. ALDER. Ground state of electron gas by a stochastic method. *Phys. Rev. Lett.* **45**, no. 7, (1980), 566–569.

[Cri11] D. CRISAN. Discretizing the continuous time filtering problem. order of convergence. In D. CRISAN and B. ROSOVSKII, eds., *The Oxford Handbook of Nonlinear Filtering*, 572–597. Oxford University Press, 2011.

[DBC98] C. DELLAGO, P. BOLHUIS, and D. CHANDLER. Efficient transition path sampling: Application to lennard-jones cluster rearrangements. *J. Chem. Phys.* **108**, no. 22, (1998), 9236–9245.

[DD11] T. DEAN and P. DUPUIS. The design and analysis of a generalized RESTART/DPR algorithm for rare event simulation. *Ann. Oper. Res.* **189**, (2011), 63–102.

[dDG05] N. DE FREITAS, A. DOUCET, and N. E. GORDON. *Sequential Monte Carlo Methods in Practice*. Springer, 2005.

[DM04] P. DEL MORAL. *Feynman-Kac formulae*. Probability and its Applications (New York). Springer-Verlag, New York, 2004. Genealogical and interacting particle systems with applications.

[FS96] D. FRENKEL and B. SMIT. *Understanding Molecular Simulation*. Academic Press, 1996.

[GS71]   R. GRIMM and R. STORER. Monte-Carlo solution of Schrödinger's equation. *J. Comp. Phys.* **7**, no. 1, (1971), 134–156.

[HM54]   J. HAMMERSLEY and K. MORTON. Poor man's Monte Carlo. *J. R. Stat. Soc. B* **16**, no. 1, (1954), 23–38.

[HT99]   Z. HARASZTI and J. TOWNSEND. The theory of direct probability redistribution and its application to rare event simulation. *ACM Transactions on Modelling and Computer Simulation* **9**, (1999), 105–140.

[HW14]   M. HAIRER and J. WEARE. The Brownian fan. *Commun. Pure Appl. Math.* **?**, no. ?, (2014), ??–??

[JDMD06]   A. JOHANSEN, P. DEL MORAL, and A. DOUCET. Sequential Monte Carlo samplers for rare events. In *Proceedings of the 6th International Workshop on Rare Event Simulation*. Bramberg, 2006.

[Kal62]   M. KALOS. Monte Carlo calculations of the ground state of three- and four-body nuclei. *Phys. Rev.* **128**, no. 4, (1962), 1791–1795.

[KL11]   J. KOLORENČ and M. LUBOS. Applications of quantum Monte Carlo methods in condensed systems. *Rep. Prog. Phys.* **74**, (2011), 1–28.

[Liu02]   J. LIU. *Monte Carlo Strategies in Scientific Computing*. Springer, 2002.

[Lor63]   E. N. LORENZ. Deterministic nonperiodic flow. *J. Atmos. Sci.* **20**, (1963), 130–141.

[MGG94]   R. N. MILLER, M. GHIL, and F. GAUTHIEZ. Advanced data assimilation in strongly nonlinear dynamical systems. *J. Atmospheric Sci.* **51**, no. 8, (1994), 1037–1056.

[MTAC12]   M. MORZFELD, X. TU, E. ATKINS, and A. J. CHORIN. A random map implementation of implicit filters. *J. Comp. Phys.* **231**, no. 4, (2012), 2049 – 2066.

[Rou06]   M. ROUSSET. *Méthods Population Monte-Carlo en Temps Continu pour la Physique Numérique*. Ph.D. thesis, L'Université Paul Sabatier Toulouse III, 2006.

[RR55]   M. ROSENBLUTH and A. ROSENBLUTH. Monte Carlo calculation of the average extension of molecular chains. *J. Chem. Phys.* **23**, no. 2, (1955), 356–359.

[VAVA91]   M. VILLEN-ALTAMIRANO and J. VILLEN-ALTAMIRANO. RESTART: A method method for accelerating rare event simulations. *Proc. of the 13th international teletraffic congress, queuing, performance and control in ATM* **9**, (1991), 71–76.

[VEW]   E. VANDEN-EIJNDEN and J. WEARE. Data assimilation in the low noise, accurate observation regime with application to the kuroshio current. *Submitted* .

[VEW14]   E. VANDEN-EIJNDEN and J. WEARE. Rare event simulation for small noise diffusions. *Commun. Pure Appl. Math.* **?**, no. ?, (2014), ??–??