



Published in final edited form as:

*Multiscale Model Simul.* 2020 ; 18(2): 646–673. doi:10.1137/18m1212100.

## OPTIMIZING WEIGHTED ENSEMBLE SAMPLING OF STEADY STATES

DAVID ARISTOFF<sup>†</sup>, DANIEL M. ZUCKERMAN<sup>‡</sup>

<sup>†</sup> Colorado State University, Fort Collins, CO 80523

<sup>‡</sup> Oregon Health & Science University, Portland, OR 97239

### Abstract

We propose parameter optimization techniques for weighted ensemble sampling of Markov chains in the steady-state regime. Weighted ensemble consists of replicas of a Markov chain, each carrying a weight, that are periodically resampled according to their weights inside of each of a number of bins that partition state space. We derive, from first principles, strategies for optimizing the choices of weighted ensemble parameters, in particular the choice of bins and the number of replicas to maintain in each bin. In a simple numerical example, we compare our new strategies with more traditional ones and with direct Monte Carlo.

### Keywords

Markov chains; resampling; sequential Monte Carlo; weighted ensemble; molecular dynamics; reaction networks; steady state; coarse graining

### AMS subject classifications.

65C05; 65C20; 65C40; 65Y05; 82C80

## 1. Introduction.

Weighted ensemble is a Monte Carlo method based on stratification and resampling, originally designed to solve problems in computational chemistry [8, 12, 15, 17, 18, 23, 34, 46, 47, 51, 61, 62, 64, 66]. Weighted ensemble currently has a substantial user base; see [65] for software and a more complete list of publications. In general terms, the method consists of periodically resampling from an ensemble of weighted replicas of a Markov process. In each of a number of *bins*, a certain number of replicas is maintained according to a prescribed replica *allocation*. The weights are adjusted so that the weighted ensemble has the correct distribution [62]. In the context of rare event or small probability calculations, weighted ensemble can significantly outperform direct Monte Carlo or independent replicas; the performance gain comes from allocating more particles to important or rare regions of state space. In this sense, weighted ensemble can be understood as an importance sampling

method. We assume the underlying Markov process is expensive to simulate, so that optimizing variance versus cost is critical. In applications, the Markov process is usually a high dimensional drift diffusion, such as Langevin molecular dynamics [50], or a continuous time Markov chain representing reaction network dynamics [3].

We focus on the computation of the average of a given function or *observable* with respect to the unique steady state of the Markov process, though many of our ideas could also be applied in a finite time setting. One of the most common applications of weighted ensemble is the computation of a *mean first passage time*, or the mean time for a Markov process to go from an initial state to some target set. The mean first passage time is an important quantity in physical and chemical processes, but it can be prohibitively large to compute using direct Monte Carlo simulation [1, 33, 42]. In weighted ensemble, the mean first passage time to a target set is reformulated, via the Hill relation [32], as the inverse of the steady state flux into the target. Here, the observable is the characteristic or indicator function of the target set, and the flux is the steady-state average of this observable. This steady state can sometimes be accessed on time scales much smaller than the mean first passage time [1, 16, 63]. On the other hand, when the mean first passage time is very large, the corresponding steady-state flux is very small and needs to be estimated with substantial precision, which is why importance sampling is needed. The steady state in this case, and in most weighted ensemble applications, is *not* known up to a normalization factor. As a result, many standard Monte Carlo methods, including those based on Metropolis–Hastings, do not apply.

In this article we consider weighted ensemble parameter optimization for steady-state averages. Our work here expands on related results in [5] for weighted ensemble optimization on finite time horizons. For a given Markov chain, weighted ensemble is completely characterized by the choice of resampling times, bins, and replica allocation. *In this article we discuss how to choose the bins and replica allocation.* We also argue that the frequency of resampling times is mainly limited by processor interaction cost and not variance. We use a first-principles, finite replica analysis based on Doob decomposing the weighted ensemble variance. Our earlier work [5] is based on this same mathematical technique, but the methods described there are not suitable for long-time computations or bin optimization [4]. As is usual in importance sampling, our parameter optimizations require estimates of the very variance terms we want to minimize. However, because *weighted ensemble is statistically exact no matter the parameters* [4, 62], these estimates can be crude. The choice of parameters affects the variance, not the mean so we only require parameters that are good enough to beat direct Monte Carlo.

From the point of view of applications, similar particle methods employing stratification include exact milestoning [7], nonequilibrium umbrella sampling [55, 22], transition interface sampling [53], trajectory tilting [54], and boxed molecular dynamics [30]. There are related methods based on sampling reactive paths, or paths going directly from a source to a target, in the context of the mean first passage time problem just cited. Such methods include forward flux sampling [2] and adaptive multilevel splitting [9, 10, 11]. These methods differ from weighted ensemble in that they estimate the mean first passage time directly from reactive paths rather than from steady state and the Hill relation. In contrast

with many of these methods, weighted ensemble is simple enough to allow for a relatively straightforward nonasymptotic variance analysis based on Doob decomposition [24].

From a mathematical viewpoint, weighted ensemble is simply a resampling-based evolutionary algorithm. In this sense it resembles particle filters, sequential importance sampling, and sequential Monte Carlo. For a review of sequential Monte Carlo, see the textbook [19], the articles [20, 21], or the compilation [28]. There is some recent work on optimizing the Gibbs–Boltzmann input potential functions in sequential Monte Carlo [6, 13, 36, 56, 60] (see also [21]). We emphasize that weighted ensemble is different from most sequential Monte Carlo methods, as it relies on a bin-based resampling mechanism rather than a globally defined fitness function like a Gibbs–Boltzmann potential. In particular, sequential Monte Carlo is more commonly used to sample rare events on finite time horizons, and may not be appropriate for very long-time computations of the sort considered here, as explained in [4].

To our knowledge, the binning and particle allocation strategies we derive here are new. A similar allocation strategy for weighted ensemble on finite time horizons was proposed in [5]. Our allocation strategy, which minimizes *mutation variance*—the variance corresponding to evolution of the replicas—extends ideas from [5] to our steady-state setup. We draw an analogy between minimizing mutation variance and minimizing sensitivity in an appendix. In most weighted ensemble simulations, and in [5], bins are chosen in an ad hoc way. We show below, however, that choosing bins carefully is important, particularly if relatively few can be afforded. We propose a new binning strategy based on minimizing *selection variance*—the variance associated with resampling from the replicas—in which weighted ensemble bins are aggregated from a collection of smaller *microbins* [15].

Formulas for the mutation and selection variance are derived in a companion paper [4] that proves an ergodic theorem for weighted ensemble time averages. These variance formulas, which we use to define our allocation and bin optimizations, involve the Markov kernel  $K$  that describes the evolution of the underlying Markov process between resampling times, as well as the solution,  $h$ , of a certain Poisson equation. We propose estimating  $K$  and  $h$  with techniques from Markov state modeling [35, 45, 48, 49]. In this formulation, the microbins correspond to the Markov states, and a microbin-to-microbin transition matrix defines the approximations of  $K$  and  $h$ .

This article is organized as follows. In section 2, we introduce notation, give an overview of weighted ensemble, and describe the parameters we wish to optimize. In section 3, we present weighted ensemble in precise detail (Algorithm 3.1), reproduce the aforementioned variance formulas (Lemmas 3.4 and 3.5) from our companion paper [4], and describe the solution,  $h$ , to a certain Poisson equation arising from these formulas ((3.15) and (3.16)). In section 4, we introduce novel optimization problems ((4.3) and (4.6)) for choosing the bins and particle allocation. The resulting allocation in (4.4) can be seen as a version of (6.4) in [5], modified for steady state and our new microbin setup. These optimizations are idealized in the sense that they involve  $K$  and  $h$ , which cannot be computed exactly. Thus in section 5, we propose using microbins and Markov state modeling to estimate their solutions (Algorithms 5.1 and 5.2). In section 6, we test our methods with a simple numerical

example. Concluding remarks and suggestions for future work are in section 7. In the appendix, we describe residual resampling [26], a common resampling technique, and draw an analogy between sensitivity and our mutation variance minimization strategy.

## 2. Algorithm.

A weighted ensemble consists of a collection of replicas, or *particles*, belonging to a common state space, with associated positive scalar weights. The particles repeatedly undergo resampling and evolution steps. Using a genealogical analogy, we refer to particles before resampling as *parents*, and just after resampling as *children*. A child is initially just a copy of its parent, though it evolves independently of other children of the same parent. The total weight remains constant in time, which is important for the stability of long-time averages [4] (and is a critical difference between the optimization algorithm described in [5] and the one outlined here). Between resampling steps, the particles evolve independently via the same Markov kernel  $K$ . The initial parents can be arbitrary, though their weights must sum to 1; see Algorithm 5.3 for a description of an initialization step tailored to steady-state calculations. The number  $N_{\text{init}}$  of initial particles can be larger than the number  $N$  of weighted ensemble particles after the first resampling step.

For the resampling or *selection* step, we require a collection of bins, denoted  $\mathcal{B}$ , and a *particle allocation*  $N_t(u)_{t \geq 0}^{u \in \mathcal{B}}$ , where  $N_t(u)$  is the (nonnegative integer) number of children in bin  $u \in \mathcal{B}$  at time  $t$ . We will assume the bins form a partition of state space (though in general they can be any grouping of the particles [4]). The total number of children is always  $N = \sum_{u \in \mathcal{B}} N_t(u)$ . *Both the bins and particle allocation are user-chosen parameters, and to a large extent this article concerns how to pick these parameters.* In general, the bins and particle allocation can be time dependent and adaptive. For simpler presentation, however, we assume that the bins are based on a fixed partition of state space. After selection, the weight of each child in bin  $u \in \mathcal{B}$  is  $\omega_t(u)/N_t(u)$ , where  $\omega_t(u)$  is the sum of the weights of all the parents in bin  $u$ . By construction, this preserves the total weight  $\sum_{u \in \mathcal{B}} \omega_t(u) = 1$ . In every occupied bin  $u$ , the  $N_t(u) - 1$  children are selected from the parents in bin  $u$  with probability proportional to the parents' weights. (Here, we mean bin  $u$  is occupied if  $\omega_t(u) > 0$ . In unoccupied bins, where  $\omega_t(u) = 0$ , we set  $N_t(u) = 0$ .)

In the evolution or *mutation* step, the time  $t$  advances, and all the children independently evolve one step according to a fixed Markov kernel  $K$ , keeping their weights from the selection step, and becoming the next parents. In practice,  $K$  corresponds to the underlying process evaluated at *resampling times*  $t$ . That is,  $K$  is a *t-skeleton* of the underlying Markov process [43]. For the mathematical analysis below, this will only be important when we consider varying the resampling times and, in particular, the  $t \rightarrow 0$  limit. (We think of this underlying process as being continuous in time, though of course time discretization is usually required for simulations. Weighted ensemble is used *on top* of an integrator of the underlying process; in particular weighted ensemble does not handle the time discretization. We will not be concerned with the unavoidable error resulting from time discretization.) *We assume that  $K$  is uniformly geometrically ergodic* [27] with respect to a stationary distribution, or steady state,  $\mu$ . Recall we are interested in steady-state averages. Thus for a

given bounded real-valued function or observable  $f$  on state space, we estimate  $\int f d\mu$  each time by evaluating the weighted sum of  $f$  on the current collection of parent particles.

We summarize weighted ensemble as follows:

- In the selection step at time  $t$ , inside each bin  $u$ , we resample  $N(u)$  children from the parents in  $u$ , according to the distribution defined by their weights. After selection, all the children in bin  $u$  have the same weight  $\omega(u)/N(u)$ , where  $\omega(u)$  is the total weight in bin  $u$  before and after selection.
- In each mutation step, the children evolve independently according to the Markov kernel  $K$ . After evolution, these children become the new parents.
- The weighted ensemble evolves by repeated selection and then mutation steps.

The time  $t$  advances after a single pair of selection and mutation steps.

See Algorithm 3.1 for a detailed description of weighted ensemble.

An important property of weighted ensemble is that it is unbiased no matter the choice of parameters: at time  $t$  the weighted particles have the same distribution as a Markov chain evolving according to  $K$ . See Theorem 3.1. With bad parameters, however, weighted ensemble can suffer from large variance, even worse than direct Monte Carlo. As there is no free lunch, choosing parameters cleverly requires either some information about  $K$ , perhaps gleaned from prior simulations or obtained adaptively during weighted ensemble simulations. We will assume we have a collection of *microbins* which we use to gain information about  $K$ . The microbins, like the weighted ensemble bins, form a partition of state space, and each bin will be a union of microbins. We use the term microbins because the microbins may be smaller than the actual weighted ensemble bins. We discuss the reasoning behind this distinction in section 5; see Remark 5.1.

### 3. Mathematical notation and algorithm.

We write  $\xi_t^1, \dots, \xi_t^N$  for the parents at time  $t$  and  $\omega_t^1, \dots, \omega_t^N$  for their weights. Their children are denoted  $\hat{\xi}_t^1, \dots, \hat{\xi}_t^N$  with weights  $\hat{\omega}_t^1, \dots, \hat{\omega}_t^N$ . Thus, weighted ensemble advances in time as follows:

$$\left\{ \xi_t^i \right\}_{i=1, \dots, N} \xrightarrow{\text{selection}} \left\{ \hat{\xi}_t^i \right\}_{i=1, \dots, N} \xrightarrow{\text{mutation}} \left\{ \xi_{t+1}^i \right\}_{i=1, \dots, N},$$

$$\left\{ \omega_t^i \right\}_{i=1, \dots, N} \xrightarrow{\text{selection}} \left\{ \hat{\omega}_t^i \right\}_{i=1, \dots, N} \xrightarrow{\text{mutation}} \left\{ \omega_{t+1}^i \right\}_{i=1, \dots, N}.$$

The particles belong to a common standard Borel state space [29]. This state space is divided into a finite collection  $\mathcal{B}$  of disjoint subsets (throughout we only consider measurable sets and functions). We define the bin weights at time  $t$  as

$$\omega_t(u) = \sum_{i: \xi_t^i \in u} \omega_t^i, \quad u \in \mathcal{B},$$

where the empty sum is zero (so an unoccupied bin  $u$  has  $\omega_t(u) = 0$ ).

For the parent  $\xi_t^i$  of  $\hat{\xi}_t^j$ , we write  $\text{par}(\hat{\xi}_t^j) = \xi_t^i$ . A child is just a copy of its parent:

$$\text{par}(\hat{\xi}_t^j) = \xi_t^i \text{ implies } \hat{\xi}_t^j = \xi_t^i.$$

### Each child has a unique parent.

Setting the number of children of each parent completely defines the children, as the choices of the children's indices (the  $j$ 's in  $\hat{\xi}_t^j$ ) do not matter. The number of children of  $\xi_t^i$  will be written  $C_t^i$ :

$$C_t^i = \# \{j: \text{par}(\hat{\xi}_t^j) = \xi_t^i\}, \quad (3.1)$$

where  $\#S$  = number of elements in a set  $S$ .

Recall that  $N_t(u)$  is the number of children in bin  $u$  at time  $t$ . We require that there is at least one child in each occupied bin, no children in unoccupied bins, and  $N$  total children at each time  $t$ . Thus,

$$N_t(u) \geq 1 \text{ if } \omega_t(u) > 0, \quad N_t(u) = 0 \text{ if } \omega_t(u) = 0, \quad \sum_{u \in \mathcal{B}} N_t(u) = N. \quad (3.2)$$

We write  $\mathcal{F}_t$  for the  $\sigma$ -algebra generated by the weighted ensemble up to, but not including, the  $t$ th selection step. We will assume the particle allocation is known before selection. Similarly, we write  $\widehat{\mathcal{F}}_t$  for the  $\sigma$ -algebra generated by weighted ensemble up to and including the  $t$ th selection step. In detail,

$$\mathcal{F}_t = \sigma\left(\left(\xi_s^i, \omega_s^i\right)_{0 \leq s \leq t}^{i=1, \dots, N}, N_s(u)_{0 \leq s \leq t}^{u \in \mathcal{B}}, \left(\hat{\xi}_s^i, \hat{\omega}_s^i\right)_{0 \leq s \leq t-1}^{i=1, \dots, N}, \left(C_s^i\right)_{0 \leq s \leq t-1}^{i=1, \dots, N}\right),$$

$$\widehat{\mathcal{F}}_t = \sigma\left(\left(\xi_s^i, \omega_s^i\right)_{0 \leq s \leq t}^{i=1, \dots, N}, N_s(u)_{0 \leq s \leq t}^{u \in \mathcal{B}}, \left(\hat{\xi}_s^i, \hat{\omega}_s^i\right)_{0 \leq s \leq t}^{i=1, \dots, N}, \left(C_s^i\right)_{0 \leq s \leq t}^{i=1, \dots, N}\right).$$

In Algorithm 3.1, we do not explicitly say how we sample the  $N_t(u)$  children in each bin  $u$ . Our framework below allows for any unbiased resampling scheme. We give a selection variance formula that assumes residual resampling; see Lemma 3.5 and Algorithm 8.1. Residual resampling has performance on par with other standard resampling methods like

systematic and stratified resampling [26]. See [57] for more details on resampling in the context of sequential Monte Carlo.

### 3.1. Ergodic averages.

We are interested in using Algorithm 3.1 to estimate

$$\theta_T \approx \int f d\mu,$$

#### Algorithm 3.1

Weighted ensemble.

---

Pick initial parents and weights  $(\xi_0^i, \omega_0^i)_{i=1, \dots, N_{\text{init}}}$  with  $\sum_{i=1}^{N_{\text{init}}} \omega_0^i = 1$ , choose a collection  $\mathcal{B}$  of bins, and define a final time  $T$ . Then for  $t = 0$ , iterate the following:

- (*Selection step*) Each parent  $\xi_t^i$  is assigned a number  $C_t^i$  of children, as follows:

In each occupied bin  $u \in \mathcal{B}$ , conditional on  $\mathcal{F}_t$ , let  $(C_t^i)_{i: \text{bin}(\xi_t^i) = u}$  be  $N_t(u)$  samples from the distribution  $\{\omega_t^i / \omega_t(u); \xi_t^i \in u\}$ , where  $N_t(u) \in \mathcal{B}$  satisfies (3.2). The children  $(\hat{\xi}_t^i)_{i=1, \dots, N}$  are defined by (3.1) with weights

$$\hat{\omega}_t^i = \frac{\omega_t(u)}{N_t(u)} \text{ if } \hat{\xi}_t^i \in u. \quad (3.3)$$

Selections in distinct bins are conditionally independent.

- (*Mutation step*) Each child  $\hat{\xi}_t^i$  independently evolves one time step as follows.

Conditionally on  $\widehat{\mathcal{F}}_t$ , the children  $(\hat{\xi}_t^i)_{i=1, \dots, N}$  evolve independently according to the Markov kernel  $K$ , becoming the next parents  $(\xi_{t+1}^i)_{i=1, \dots, N}$ , with weights

$$\omega_{t+1}^i = \hat{\omega}_t^i, \quad i = 1, \dots, N. \quad (3.4)$$

Then time advances,  $t \leftarrow t + 1$ . Stop if  $t = T$ ; else return to the selection step.

Algorithm 5.1 outlines an optimization for the allocation  $N_t(u) \in \mathcal{B}$ , and a procedure for choosing the bins  $\mathcal{B}$  is in Algorithm 5.2. For the initialization, see Algorithm 5.3.

---

where we recall  $\mu$  is the stationary distribution of the Markov kernel  $K$ , and

$$\theta_T = \frac{1}{T} \sum_{t=0}^{T-1} \sum_{i=1}^N \omega_t^i f(\xi_t^i). \quad (3.5)$$

Note that  $\theta_T$  is simply the running average of  $f$  over the weighted ensemble up to time  $T-1$ . In particular, (3.5) is not a time average over ancestral lines that survive up to time  $T-1$ , but

rather it is an average over the weighted ensemble at each time  $0 \leq t \leq T-1$ . Our time averages (3.5) require no replica storage and should have smaller variances than averages over surviving ancestral lines [4].

### 3.2. Consistency results.

The next results, from a companion article [4], show that weighted ensemble is unbiased and that weighted ensemble time averages converge. The latter does not in general follow from the former, as standard unbiased particle methods such as sequential Monte Carlo can have variance explosion [4]. (The proofs in [4] have  $N_{\text{init}} = N$ , but they are easily modified for  $N_{\text{init}} \neq N$ .)

#### Theorem 3.1 (from [4]).

In Algorithm 3.1, suppose that the initial particles and weights are distributed as  $\nu$ , in the sense that

$$\mathbb{E} \left[ \sum_{i=1}^{N_{\text{init}}} \omega_0^i g(\xi_0^i) \right] = \int g d\nu \quad (3.6)$$

for all real-valued bounded functions  $g$  on state space. Let  $(\xi_t)_{t=0}$  be a Markov chain with kernel  $K$  and initial distribution  $\xi_0 \sim \nu$ . Then for any time  $T > 0$ ,

$$\mathbb{E} \left[ \sum_{i=1}^N \omega_T^i g(\xi_T^i) \right] = \mathbb{E}[g(\xi_T)]$$

for all real-valued bounded functions  $g$  on state space.

Recall we assume  $K$  is uniformly geometrically ergodic [27].

#### Theorem 3.2 (from [4]).

Weighted ensemble is ergodic in the following sense:

$$\lim_{T \rightarrow \infty} \theta_T \stackrel{a.s.}{=} \int f d\mu.$$

Theorem 3.1 does not use ergodicity of  $K$ , though Theorem 3.2 obviously does.

### 3.3. Variance analysis.

We will make use of the following analysis from [4] concerning the weighted ensemble variance. Define the Doob martingales [25]

$$D_t = \mathbb{E}[\theta_T | \mathcal{F}_t], \quad \widehat{D}_t = \mathbb{E}[\theta_T | \widehat{\mathcal{F}}_t]. \quad (3.7)$$



The Doob decomposition in Theorem 3.3 below filters the variance through the  $\sigma$ -algebras  $\mathcal{F}_t$  and  $\widehat{\mathcal{F}}_t$ . It is a way to decompose the variance into contributions from the initial condition and each time step. This type of telescopic variance decomposition is quite standard in sequential Monte Carlo, although it is usually applied at the level of measures on state space, which corresponds to the infinite particle limit  $N \rightarrow \infty$  [19]. We use the finite  $N$  formula directly to minimize variance, building on ideas in [5].

**Theorem 3.3 (from [4]).**

*By Doob decomposition,*

$$\theta_T^2 - \mathbb{E}[\theta_T]^2 + R_T = (D_0 - \mathbb{E}[\theta_T])^2 \quad (3.8)$$

$$+ \sum_{t=1}^{T-1} \mathbb{E}[(D_t - \widehat{D}_{t-1})^2 \mid \widehat{\mathcal{F}}_{t-1}] \quad (3.9)$$

$$+ \sum_{t=1}^{T-1} \mathbb{E}[(\widehat{D}_{t-1} - D_{t-1})^2 \mid \mathcal{F}_{t-1}], \quad (3.10)$$

where  $R_T$  is mean zero,  $\mathbb{E}[R_T] = 0$ .

The terms on the right-hand side of (3.8), (3.9), and (3.10) yield the contributions to the variance of  $\theta_T$  from, respectively, the initial condition, mutation steps, and selection steps of Algorithm 3.1. We thus refer to the summands of (3.9) and (3.10) as the mutation variance and selection variance of weighted ensemble.

Below, define

$$h_t = \sum_{s=0}^{T-t-1} K^s f, \quad (3.11)$$

and for any function  $g$  and probability distribution  $\eta$  on state space, let

$$\text{Var}_\eta g = \int g^2(\xi)\eta(d\xi) - \left( \int g(\xi)\eta(d\xi) \right)^2. \quad (3.12)$$

Above and below, the dependence of  $D_t$ ,  $\widehat{D}_t$ , and  $h_t$  on  $T$  is suppressed.

**Lemma 3.4 (from [4]).**

The mutation variance at time  $t$  is

$$\mathbb{E}[(D_{t+1} - \widehat{D}_t)^2 \mid \widehat{\mathcal{F}}_t] = \frac{1}{T^2} \sum_{i=1}^N (\widehat{\omega}_t^i)^2 \text{Var}_{K(\widehat{\xi}_t^i)} h_{t+1}. \quad (3.13)$$

To formulate the selection variance, we define

$$\delta_t^i = \frac{N_t(u)\omega_t^i}{\omega_t(u)} - \left\lfloor \frac{N_t(u)\omega_t^i}{\omega_t(u)} \right\rfloor \text{ if } \xi_t^i \in u, \quad \delta_t(u) = \sum_{i: \xi_t^i \in u} \delta_t^i,$$

where  $\lfloor x \rfloor$  is the floor function, or the greatest integer less than or equal to  $x$ . In Lemma 3.5, we assume that the  $(C_t^j)^{j=1, \dots, N}$  in Algorithm 3.1 are obtained using residual resampling. See [26, 57] or Algorithm 8.1 in the appendix below for a description of residual resampling.

**Lemma 3.5 (from [4]).**

The selection variance at time  $t$  is

$$\begin{aligned} \mathbb{E} \left[ (\widehat{D}_t - D_t)^2 \mid \mathcal{F}_t \right] &= \frac{1}{T^2} \sum_{u \in \mathcal{B}} \left( \frac{\omega_t(u)}{N_t(u)} \right)^2 \delta_t(u) \text{Var}_{\eta_t(u)} K h_{t+1}, \\ \eta_t(u) &:= \sum_{i: \binom{\xi_t^i}{\xi_t} = u} \frac{\delta_t^i}{\delta_t(u)} \delta_{\xi_t^i}. \end{aligned} \tag{3.14}$$

By definition (3.12), the variances in Lemmas 3.4 and 3.5 are rewritten as

$$\begin{aligned} \text{Var}_{K \left( \widehat{\xi}_t^i \cdot \right)} h_{t+1} &= K h_{t+1}^2 \left( \widehat{\xi}_t^i \right) - \left( K h_{t+1} \left( \widehat{\xi}_t^i \right) \right)^2, \\ \text{Var}_{\eta_t(u)} K h_{t+1} &= \sum_{i: \binom{\xi_t^i}{\xi_t} = u} \frac{\delta_t^i}{\delta_t(u)} \left( K h_{t+1} \left( \xi_t^i \right) \right)^2 - \left( \sum_{i: \binom{\xi_t^i}{\xi_t} = u} \frac{\delta_t^i}{\delta_t(u)} K h_{t+1} \left( \xi_t^i \right) \right)^2. \end{aligned}$$

### 3.4. Poisson equation.

Because we are interested in large time horizons  $T$ , below we will consider the mutation and selection variances in the limit  $T \rightarrow \infty$ . We will see that for any probability distribution  $\eta$  on state space,

$$\lim_{T \rightarrow \infty} \text{Var}_{\eta} h_{t+1} = \text{Var}_{\eta} h,$$

$$\lim_{T \rightarrow \infty} \text{Var}_{\eta} K h_{t+1} = \text{Var}_{\eta} K h,$$

where  $h$  is the solution to the Poisson equation [40, 44]

$$(Id - K)h = f - \int f d\mu, \quad \int h d\mu = 0, \quad (3.15)$$

where  $Id$  = the identity kernel. Existence and uniqueness of the solution  $h$  easily follow from uniform geometric ergodicity of the Markov kernel  $K$ . Indeed, if  $(\xi_t)_{t \geq 0}$  is a Markov chain with kernel  $K$ , then we can write

$$\begin{aligned} h(\xi) &= \sum_{t=0}^{\infty} \left( K^t f(\xi) - \int f d\mu \right) \\ &= \sum_{t=0}^{\infty} \left( \mathbb{E}[f(\xi_t) \mid \xi_0 = \xi] - \mathbb{E}[f(\xi_t) \mid \xi_0 \sim \mu] \right), \end{aligned} \quad (3.16)$$

where  $\xi_0 \sim \mu$  indicates  $\xi_0$  is initially distributed according to the steady state  $\mu$  of  $K$ . Uniform geometric ergodicity and the Weierstrass  $M$ -test show that the sums in (3.16) converge absolutely and uniformly. As a consequence,  $h$  in (3.16) solves (3.15).

Interpreting (3.16),  $h(\xi)$  is the mean discrepancy of a time average of  $f(\xi_t)$  starting at  $\xi_0 = \xi$  with a time average of  $f(\xi_t)$  starting at steady state  $\xi_0 \sim \mu$ . This discrepancy in the time averages has been normalized so that it has a nontrivial limit and, in particular, does not vanish, as time goes to infinity.

The Poisson solution  $h$  is critical for understanding and estimating the weighted ensemble variance. Besides,  $h$  can be used to identify model features, such as sets that are *metastable* for the underlying Markov chain defined by  $K$ , as well as narrow pathways between these sets. Metastable sets are, roughly speaking, regions of state space in which the Markov chain tends to become trapped.

To understand the behavior of  $h$ , we define metastable sets more precisely. A region  $R$  in state space is metastable for  $K$  if  $(\xi_t)_{t \geq 0}$  tends to equilibrate in  $R$  much faster than it escapes from  $R$ . The rate of equilibration in  $R$  can be understood in terms of the *quasi-stationary distribution* [14] in  $R$ . See, e.g., [39] for more discussion on metastability. The Poisson solution  $h$  tends to be nearly constant in regions that are metastable for  $K$ . This is because the mean discrepancy in a time average of  $f(\xi_t)$  over two copies of  $(\xi_t)_{t \geq 0}$  with different starting points in the same metastable set  $R$  is small: both copies tend to reach the same quasi-stationary distribution in  $R$  before escaping from  $R$ . In the regions between metastable sets, however,  $h$  tends to have large variance. If  $f$  is a characteristic or indicator function, this variance tends to be larger the closer these regions are to the support of  $f$  (the set where  $f = 1$ ). More generally, the variance of  $h$  is larger near regions  $R$  where the stationary average  $\int_R f d\mu$  of  $f$  is large. See Figure 6.1 for an illustration of these features.

#### 4. Minimizing the variance.

Our strategy for minimizing the variance is based on *choosing the particle allocation to minimize mutation variance* and *picking the bins to mitigate selection variance*. Minimizing mutation variance is closely connected with minimizing a certain sensitivity; see the

appendix below. Both strategies require some coarse estimates of  $K$  and  $h$ . We propose using ideas from Markov state modeling to construct microbins from which we estimate  $K, h$ . The microbins can be significantly smaller than the weighted ensemble bins, as we discuss below.

#### 4.1. Resampling times.

Recall that weighted ensemble is fully characterized by the choice of resampling times, bins, and particle allocation. Though we focus on the latter two here, we briefly comment on the former. The resampling times are implicit in our framework. We assume here that  $K = K_t$  is a  $t$ -skeleton of an underlying Markov process, or a sequence of values of the underlying process at time intervals  $t$ . In this setup,  $t$  is a fixed resampling time, and we are ignoring the time discretization. (Actually, the resampling times need not be fixed—they can be any times at which the underlying process has the strong Markov property [5]. In practice, the underlying process must be discretized in time, and weighted ensemble is used with the discretized process.)

Suppose that the underlying Markov process is one of the ones mentioned in the introduction: either Langevin dynamics, or a reaction network modeled by a continuous time Markov chain on finite state space. Suppose, moreover, that microbins in continuous state space are domains with piecewise smooth boundaries (for instance, Voronoi regions), and that the bins are unions of microbins. Then the underlying process does not cross between distinct microbins, or between distinct bins, infinitely often in finite time. As a result, weighted ensemble should not degenerate in the limit as  $t \rightarrow 0$ , as we now show.

Consider the variance from selection in Lemma 3.5. By (3.3), the weights of all the children in each bin  $u \in \mathcal{B}$  are all equal to  $\omega_t(u)/N_t(u)$  after the selection step. If  $t$  is very small, then almost none of the children move to different microbins or bins in the mutation step. If exactly zero of the children change bins, then for residual resampling in the selection step at the next time  $t$ , provided the allocation  $N_t(u)^{u \in \mathcal{B}}$  has not changed, we have  $\delta_t^i = 0$  for all  $i = 1, \dots, N$ . (Note that with our optimal allocation strategy in Algorithm 5.1, if we avoid unnecessary resampling of  $R_t(u)^{u \in \mathcal{B}}$ , then  $N_t(u)^{u \in \mathcal{B}}$  does not change unless particles move between microbins.) Thus from Lemma 3.5 there is zero selection variance and, in fact, no resampling occurs in the selection step. Provided particles do not cross bins or microbins infinitely often in finite time, and the allocation only changes when particles move between microbins, this suggests there is *no variance blowup when  $t \rightarrow 0$* . We expect then that the frequency  $t$  of resampling should be driven not by variance cost but by computational cost, e.g., processor communication cost.

#### 4.2. Minimizing mutation variance.

The mutation variance depends on the choice of weighted ensemble bins as well as the particle allocation at each time  $t$ . In this section we focus on the particle allocation for an arbitrary choice  $\mathcal{B}$  of bins. To understand this relationship between the allocation and mutation variance, following ideas from [5], *we look at the mutation variance visible before*

*selection*. It is so named because, unlike the mutation variance in Lemma 3.4, it is a function of quantities  $\omega_t(u)$ ,  $N_t(u)$ ,  $(\omega_t^i, \xi_t^i)_{i=1, \dots, N}$  that are known at time  $t$  before selection.

**Proposition 4.1.**—*The mutation variance visible before selection satisfies*

$$\lim_{T \rightarrow \infty} T^2 \mathbb{E} \left[ (D_{t+1} - \widehat{D}_t)^2 \mid \mathcal{F}_t \right] = \sum_{u \in \mathcal{B}} \frac{\omega_t(u)}{N_t(u)} \sum_{i: \xi_t^i \in u} \omega_t^i \text{Var}_{K(\xi_t^i, \cdot)} h,$$

where  $h$  is defined in (3.15).

**Proof.:** By definition of the selection step (see Algorithm 3.1),

$$\mathbb{E} [C_t^i \mid \mathcal{F}_t] = \frac{N_t(u) \omega_t^i}{\omega_t(u)}. \quad (4.1)$$

From Lemma 3.4,

$$\begin{aligned} T^2 \mathbb{E} \left[ (D_{t+1} - \widehat{D}_t)^2 \mid \mathcal{F}_t \right] &= \mathbb{E} \left[ \sum_{i=1}^N (\widehat{\omega}_t^i)^2 \text{Var}_{K(\widehat{\xi}_t^i, \cdot)} h_{t+1} \mid \mathcal{F}_t \right] \\ &= \sum_{u \in \mathcal{B}} \left( \frac{\omega_t(u)}{N_t(u)} \right)^2 \mathbb{E} \left[ \sum_{i: \widehat{\xi}_t^i \in u} \text{Var}_{K(\widehat{\xi}_t^i, \cdot)} h_{t+1} \mid \mathcal{F}_t \right] \quad (\text{using (3.3)}) \\ &= \sum_{u \in \mathcal{B}} \left( \frac{\omega_t(u)}{N_t(u)} \right)^2 \sum_{i: \xi_t^i \in u} \mathbb{E} \left[ \sum_{j: \text{par}(\widehat{\xi}_t^j) = \xi_t^i} \left[ Kh_{t+1}^2(\widehat{\xi}_t^j) - (Kh_{t+1}(\widehat{\xi}_t^j))^2 \right] \mid \mathcal{F}_t \right] \\ &= \sum_{u \in \mathcal{B}} \frac{\omega_t(u)}{N_t(u)} \sum_{i: \xi_t^i \in u} \omega_t^i \left[ Kh_{t+1}^2(\xi_t^i) - (Kh_{t+1}(\xi_t^i))^2 \right] \quad (\text{using (4.1)}) \end{aligned} \quad (4.2)$$

In light of (3.11) and (3.16), and using the fact that

$$\text{Var}_{K(\xi_t^i, \cdot)} h = Kh^2(\xi_t^i) - (Kh(\xi_t^i))^2,$$

we get the result from letting  $T \rightarrow \infty$ .  $\square$

We let  $T \rightarrow \infty$  since we are interested in long-time averages. The simpler formulas that result, as they involve  $h$  instead of  $h_t$ , allow for strategies to estimate fixed optimal bins before beginning weighted ensemble simulations, which we discuss more below. Thus for minimizing the limiting mutation variance, we consider the following optimization:

$$\begin{aligned} & \text{minimize } \sum_{u \in \mathcal{B}} \frac{\omega_t(u)}{N_t(u)} \sum_{i: \xi_t^i \in u} \omega_t^i \text{Var}_{K(\xi_t^i, \cdot)} h, \\ & \text{over all choices of } N_t(u) \in \mathbb{R}^+ \text{ such that } \sum_{u \in \mathcal{B}} N_t(u) = N, \end{aligned} \quad (4.3)$$

where the bins  $\mathcal{B}$  are fixed, and we temporarily allow the allocation  $N_t(u)^{u \in \mathcal{B}}$  to be noninteger. A Lagrange multiplier calculation shows that the solution to (4.3) is

$$N_t(u) = \frac{N \sqrt{\omega_t(u) \sum_{i: \xi_t^i \in u} \omega_t^i \text{Var}_{K(\xi_t^i, \cdot)} h}}{\sum_{u \in \mathcal{B}} \sqrt{\omega_t(u) \sum_{i: \xi_t^i \in u} \omega_t^i \text{Var}_{K(\xi_t^i, \cdot)} h}}, \quad (4.4)$$

provided the denominator above is nonzero. *Note this solution is idealized, as  $N_t(u)$  must always be an integer, and  $h$  and  $K$  are not known exactly.* We explain a practical implementation of (4.4) in Algorithm 5.1.

Our choice of the particle allocation will be based on (4.4); see Algorithm 5.1. Notice that at each time  $t$  we only minimize one term, the summand in (3.9) corresponding to the mutation variance at time  $t$  in the Doob decomposition in Theorem 3.3. Later, when we optimize bins, we will optimize the summand in (3.10) corresponding to the selection variance at time  $t$ . In particular, we only minimize the mutation and selection variances at the current time, and not the sum of these variances over all times. In the  $T \rightarrow \infty$  limit, we expect that weighted ensemble reaches a steady state, provided the bin choice and allocation strategy (e.g., from Algorithms 5.1 and 5.2) do not change over time. If the weighted ensemble indeed reaches a steady state, then the mutation and selection variances become stationary in  $t$ , making it reasonable to minimize them only at the current time. (Under appropriate conditions, the variances in (3.9)–(3.10) should also become independent over time  $t$  in the  $N \rightarrow \infty$  limit. See [19] for related results in the context of sequential Monte Carlo.)

Note the term  $\text{Var}_{K(\xi_t^i, \cdot)} h = Kh^2(\xi_t^i) - (Kh(\xi_t^i))^2$  appearing in (4.4). As discussed in section 3,

*this variance tends to be large in regions between metastable sets.* The optimal allocation (4.4) favors putting children in such regions, increasing the likelihood that their descendants will visit both the adjacent metastable sets. See the appendix for a connection between minimizing mutation variance and minimizing the sensitivity of the stationary distribution  $\mu$  to perturbations of  $K$ .

#### 4.3. Mitigating selection variance.

We begin by observing that if bins are small, then so is the selection variance. In particular, if each bin is a single point in state space, then Lemma 3.5 shows that the selection variance is *zero*. One way, then, to get small selection variance is to have a lot of bins. When simulations of the underlying Markov chain are very expensive, however, we cannot afford a

large number of bins; see Remark 5.1 in section 5 below. As a result the bins are not so small, and we investigate the selection variance to decide how to construct them.

**Lemma 4.2.**—*The selection variance at time  $t \geq 1$  satisfies*

$$\lim_{T \rightarrow \infty} T^2 \mathbb{E} \left[ \left( \widehat{D}_t - D_t \right)^2 \mid \mathcal{F}_t \right] = \sum_{u \in \mathcal{B}} \left( \frac{\omega_t(u)}{N_t(u)} \right)^2 \delta_t(u) \text{Var}_{\eta_t(u)} Kh, \quad (4.5)$$

where  $\eta_t(u)$  is defined in Lemma 3.5 and  $h$  is defined in (3.15).

**Proof.** From Lemma 3.5, (3.11), and (3.16), letting  $T \rightarrow \infty$  gives the result.  $\square$

We choose to mitigate selection variance by choosing bins so that (4.5) is small. This likely requires some sort of search in bin space. For simplicity, we assume that the bins do not change in time and are chosen at the start of Algorithm 3.1. Of course it is possible to update bins adaptively, at a frequency that depends on how the cost of bin searches compares to that of particle evolution.

We will minimize an *agnostic* variant of (4.5), for which we make no assumptions about  $N_t(u)$ ,  $\omega_t(u)$ , and  $\delta_t(u)$ . This allows us to optimize fixed bins using a time-independent objective function, without taking the particle allocation into account. Our agnostic optimization also is not specific to residual resampling. Indeed, though the precise formula for the selection variance depends on the resampling method, the selection variance should always contain terms of the form  $\text{Var}_{\eta} Kh$ , where  $\eta$  are probability distributions in the individual weighted ensemble bins; see [4]. It is exactly these terms that we choose to minimize.

Thus for our agnostic selection variance minimization, we let

$$\eta_u^{\text{unif}}(d\xi) = \frac{\mathbb{1}_{\xi \in u} d\xi}{\int \mathbb{1}_{\xi \in u} d\xi}$$

be the uniform distribution in bin  $u \in \mathcal{B}$ , and consider the following problem:

$$\text{minimize} \sum_{u \in \mathcal{B}} \text{Var}_{\eta_u^{\text{unif}}} Kh$$

over choices of  $\mathcal{B}$ , subject to the constraint  $\#\mathcal{B} = M$ .

Like (4.3), *this is idealized because we cannot directly access  $K$  or  $h$* . We describe a practical implementation of (4.6) in Algorithm 5.2. Informally, solutions to (4.6) are characterized by the property that, inside each individual bin  $u$ , the value of  $Kh$  does not change very much.

Our choice of bins is based on (4.6). Here  $M$  is the desired total number of bins. Our agnostic perspective leads us to use the uniform distribution  $\eta_u^{\text{unif}}$  in each bin  $u$ . When bins are formed from combinations of a fixed collection of microbins, (4.6) is a discrete

optimization problem that usually lacks a closed form solution. Algorithm 5.2 below solves a discrete version of (4.6) by simulated annealing. Because of the similarity of (4.6) with the  $k$ -means problem [41], we expect that there are more efficient methods, but this is not the focus of the present work.

## 5. Microbins and parameter choice.

To approximate the solutions to (4.3) and (4.6), we use microbins to gain information about  $K$  and  $h$ . The collection  $\mathcal{MB}$  of microbins is a finite partition of state space that refines the weighted ensemble bins  $\mathcal{B}$ , in the sense that every element of  $\mathcal{B}$  is a union of elements of  $\mathcal{MB}$ . Thus each bin is comprised of a number of microbins, and each microbin is inside exactly one bin. The idea is to use exploratory simulations, over short time horizons, to approximate  $K$  and  $h$  by observing transitions between microbins.

In more detail, we estimate the probability to transition from microbin  $p \in \mathcal{MB}$  to microbin  $q \in \mathcal{MB}$  by a matrix  $\tilde{K} = (\tilde{K}_{pq})$ ,

$$\tilde{K}_{pq} \approx \frac{\iint \nu(d\xi) K(\xi, d\xi') \mathbb{1}_{\xi \in p, \xi' \in q}}{\int \nu(d\xi) \mathbb{1}_{\xi \in p}}, \quad (5.1)$$

and we estimate  $f$  on microbin  $p \in \mathcal{MB}$  by a vector  $\tilde{f} = (\tilde{f}_p)$ ,

$$\tilde{f}_p \approx \frac{\int f(\xi) \mathbb{1}_{\xi \in p} \nu(d\xi)}{\int \mathbb{1}_{\xi \in p} \nu(d\xi)}. \quad (5.2)$$

Here  $\nu$  is some convenient measure, for instance an empirical measure obtained from preliminary weighted ensemble simulations. This strategy echoes work in the Markov state model community [35, 45, 48, 49].

For small enough microbins, we could replace  $\nu$  in (5.1) with any other measure without changing too much the value of the estimates on the right-hand sides of (5.1) and (5.2). Moreover, if  $f$  is the characteristic function of a microbin, then  $\nu$  could be replaced with any measure supported in microbin  $p$  *without changing at all* the value of the right-hand side of (5.2). This is the case for the mean first passage time problem mentioned in the introduction, and fleshed out in the numerical example in section 6; there,  $f$  is the characteristic function of the target set, which can be chosen to be a microbin.

### Algorithm 5.1

Optimizing the particle allocation.

---

Given the particles and weights  $(\xi_0^i, \omega_0^i)_{i=1, \dots, N_{\text{init}}}$  at  $t=0$  or  $(\xi_t^i, \omega_t^i)_{i=1, \dots, N}$  at  $t=1$ :

- Define the following approximate solution to (4.3):



$$\begin{aligned} \tilde{N}_t(u) &= \frac{N \sqrt{\omega_t(u) \sum_{i: \xi_t^i \in u} \omega_t^i [\tilde{K}\tilde{h}^2 - (\tilde{K}\tilde{h})^2]} p(\xi_t^i)}}{\sum_{u \in \mathcal{B}} \sqrt{\omega_t(u) \sum_{i: \xi_t^i \in u} \omega_t^i [\tilde{K}\tilde{h}^2 - (\tilde{K}\tilde{h})^2]} p(\xi_t^i)}, \end{aligned} \quad (5.3)$$

where  $p(\xi_t^i) \in \mathcal{MB}$  is the microbin containing  $\xi_t^i$

- Let  $\tilde{N}$  count the occupied bins,

$$\tilde{N} = \sum_{u \in \mathcal{B}} \mathbb{1}_{\omega_t(u) > 0}.$$

- Let  $R_t(u) \in \mathcal{B}$  be  $N - \tilde{N}$  samples from the distribution  $\{\tilde{N}_t(u)/N : u \in \mathcal{B}\}$ .
- In Algorithm 3.1, define the particle allocation as

$$N_t(u) = \mathbb{1}_{\omega_t(u) > 0} + R_t(u).$$

If the denominator of (5.3) is 0, we set  $N_t(u) = \#\{i: \xi_t^i \in u\}$ .

With  $\tilde{f}$  and the microbin-to-microbin transition matrix  $\tilde{K}$  in hand, we can obtain an approximate solution  $\tilde{h}$  to the Poisson equation (3.15), simply by replacing  $K$ ,  $f$ , and  $\mu$  in that equation with, respectively,  $\tilde{K}$ ,  $\tilde{f}$ , and the stationary distribution  $\tilde{\mu}$  of  $\tilde{K}$  (we assume  $\tilde{K}$  is aperiodic and irreducible). That is,  $\tilde{h}$  solves

$$(\tilde{I} - \tilde{K})\tilde{h} = \tilde{f} - \tilde{f}^T \tilde{\mu} \tilde{\mathbb{1}}, \quad \tilde{h}^T \tilde{\mu} = 0, \quad (5.4)$$

where  $\tilde{I}$  and  $\tilde{\mathbb{1}}$  are the identity matrix and all-ones column vector of the appropriate sizes, and  $\tilde{f}$ ,  $\tilde{\mu}$ , and  $\tilde{h}$  are column vectors. Then we can approximate the solutions to (4.3) and (4.6) by simply replacing  $K$  and  $h$  in those optimization problems with  $\tilde{K}$  and  $\tilde{h}$ . See Algorithms 5.1 and 5.2 for details. We can also use  $\tilde{\mu}$  to initialize weighted ensemble; see Algorithm 5.3.

### Algorithm 5.2

Optimizing the bins.

Choose an initial collection  $\mathcal{B}'$  of bins. Define an objective function on the bin space,

$$\mathcal{O}(\mathcal{B}') = \sum_{u \in \mathcal{B}'} \text{Var}(\tilde{K}\tilde{h}|_u), \quad (5.5)$$

where  $\tilde{K}\tilde{h}|_u$  is the restriction of  $\tilde{K}\tilde{h}$  to  $\{p \in \mathcal{M}\mathcal{B} : p \in u\}$ , and  $\text{Var}(\tilde{K}\tilde{h}|_u)$  is the usual vector population variance. Choose an annealing parameter  $\alpha > 0$ , set  $\mathcal{B}_{opt} = \mathcal{B}$ , and iterate the following for a user-prescribed number of steps:

1. Perturb  $\mathcal{B}$  to get new bins  $\mathcal{B}'$ .  
(Say by moving a microbin from one bin to another bin.)
2. With probability  $\min\{1, \exp[\alpha(\mathcal{O}(\mathcal{B}) - \mathcal{O}(\mathcal{B}'))]\}$ , set  $\mathcal{B} = \mathcal{B}'$ .
3. If  $\mathcal{O}(\mathcal{B}) < \mathcal{O}(\mathcal{B}_{opt})$ , then update  $\mathcal{B}_{opt} = \mathcal{B}$ . Return to step 1.

Once the bin search is complete, the output is  $\mathcal{B} = \mathcal{B}_{opt}$ .

---

We have in mind that the microbins are constructed using ideas from Markov state modeling [35, 45, 48, 49]. In this setup, the microbins are simply the Markov states. These could be determined from a clustering analysis (e.g., using  $k$ -means [41]) from preliminary weighted ensemble simulations with short time horizons. The resulting Markov state model can be crude: it will be used only for choosing parameters, and weighted ensemble is exact no matter the parameters. Indeed if our Markov state model was very refined, it could be used directly to estimate  $\int f d\mu$ . In practice, we expect our crude model could estimate  $\int f d\mu$  with significant bias. In our formulation, a bad set of parameters may lead to large variance, but there is never any bias. In short, the Markov state model should be good enough to pick reliable weighted ensemble parameters, but not necessarily good enough to accurately estimate  $\int f d\mu$ .

**Remark 5.1.**

We distinguish microbins from bins because the number of weighted ensemble bins is limited by the number of particles we can afford to simulate over the time horizon needed to reach  $\mu$ . As an extreme case, suppose we have many more bins than particles, so that all bins contain 0 or 1 particles at almost every time. Then, because Algorithm 3.1 requires at least one child per occupied bin, parents almost never have more than one child, and we recover direct Monte Carlo. This condition, that the collection of parents in a given bin must have at least one child, is essential for the stability of long-time calculations [4]. As a result, too many bins leads to poor weighted ensemble performance. A very rough rule of thumb is that the number  $M$  of bins should be not too much larger than the number  $N$  of particles.

The number of weighted ensemble bins is limited by the number  $N$  of particles. (In the references in the introduction,  $N$  is usually on the order of  $10^2$  to  $10^3$ .) The number of microbins, on the other hand, is limited primarily by the cost of the sampling that produces  $\tilde{h}$  and  $\tilde{K}$ . The microbins could be computed by postprocessing exploratory weighted ensemble data generated using larger bins. The quality of the microbin-to-microbin transition matrix  $\tilde{K}$  depends on the microbins and the number of particles used in these exploratory simulations. But these exploratory simulations, compared to our steady-state weighted ensemble simulations, could use more particles as their time horizons can be much shorter. As a result the number of microbins can be much greater than the number of bins.

In Algorithm 5.2, we could enforce an additional condition that the bins must be connected regions in state space. We do this in our implementation of Algorithm 5.2 in section 6 below.

Traditionally, bins are connected, not-too-elongated regions, e.g., Voronoi regions. Bins are chosen this way because resampling in bins with distant particles can lead to a large variance in the weighted ensemble. However, *since we are employing weighted ensemble only to compute  $\int f d\mu$  for a single observable  $f$* , a large variance in the full ensemble can be tolerated so long as the variance associated with estimating  $\int f d\mu$  is still small. This could be achieved with disconnected or elongated bins, or even a nonspatial assignment of particles to bins (based e.g., on the values of  $\tilde{K}\tilde{h}$  on the microbins containing the particles). We leave a more complete investigation to future work.

## 5.1. Initialization.

Note that the steady state  $\tilde{\mu}$  of  $\tilde{K}$  can be used to precondition the weighted ensemble simulations, so that they start closer to the true steady state. This basically amounts to adjusting the weights of the initial particles so that they match  $\tilde{\mu}$ . This is called *reweighting* in the weighted ensemble literature [8, 15, 51, 64]. One way to do this is the following. Take initial particles and weights  $(\xi_0^i, \omega^i)_{i=1, \dots, N_{\text{init}}}$  from the preliminary simulations that define  $\tilde{K}$ ,  $\tilde{\mu}$ ,  $\tilde{f}$ , and  $\tilde{h}$ . These initial particles can be a large subsample from these simulations; in particular we can have an initial number of particles  $N_{\text{init}} \gg N$  much greater than the number of particles in weighted ensemble simulations. (The large number can be obtained by sampling at different times along the the preliminary simulation particle ancestral lines.) We require that there is at least one initial particle in each microbin. The weights  $(\omega^i)_{i=1, \dots, N_{\text{init}}}$  of these particles are adjusted using  $\tilde{\mu}$  to get new weights  $(\omega_0^i)_{i=1, \dots, N_{\text{init}}}$ , such that the total adjusted weight in each microbin matches the value of  $\tilde{\mu}$  on the same microbin. Then these  $N_{\text{init}} \gg N$  initial particles are fed into the first ( $t = 0$ ) selection step of Algorithm 3.1. This selection step prunes the number of particles to a manageable number,  $N$ , for the rest of the main weighted ensemble simulation. See Algorithm 5.3 for a precise description of this initialization.

### Algorithm 5.3

Initializing weighted ensemble.

---

After the preliminary simulations that produce a collection of particles and weights  $(\xi_0^i, \omega^i)_{i=1, \dots, N_{\text{init}}}$ , together with approximations  $\tilde{K}$ ,  $\tilde{\mu}$ ,  $\tilde{f}$ , and  $\tilde{h}$  of  $K$ ,  $\mu$ ,  $f$ , and  $h$ :

- Adjust the weights of the particles in each microbin  $p$  according to  $\tilde{\mu}_p$ :

$$\omega_0^i = \omega^i \frac{\tilde{\mu}_p}{\sum_{j: \xi_0^j \in p} \omega^j} \text{ if } \xi_0^i \in p.$$

There should be at least one initial particle in each microbin,

$$\sum_{j: \xi_0^j \in p} \omega^j > 0, p \in \mathcal{MB}.$$

- Proceed to the selection step of Algorithm 3.1 at time  $t = 0$  with the initial particles  $\xi_0^1, \dots, \xi_0^{N_{\text{init}}}$  having the adjusted weights  $\omega_0^1, \dots, \omega_0^{N_{\text{init}}}$ .

The number,  $N_{\text{init}}$ , of initial particles can be much greater than the number,  $N$ , of particles in the weighted ensemble simulations of Algorithm 3.1.

## 5.2. Gain over naive parameter choices.

The gain of optimizing parameters, compared to naive parameter choices or direct Monte Carlo, comes from the larger number of particles that optimized weighted ensemble puts in important regions of state space, compared to a naive method. *These important regions are exactly the ones identified by  $h$* , roughly speaking they are regions  $R$  where the variance of  $h$  is large. A rule of thumb is that the variance can decrease by a factor of up to

$$\frac{\text{average \# of particles in } R \text{ with optimized parameters}}{\text{average \# of particles in } R \text{ with naive method}}. \quad (5.6)$$

To see why, consider the mutation variance from Proposition 4.1,

$$\lim_{T \rightarrow \infty} T^2 \mathbb{E} \left[ (D_{t+1} - \widehat{D}_t)^2 \mid \mathcal{F}_t \right] = \sum_{u \in \mathcal{B}} \frac{\omega_t(u)}{N_t(u)} \sum_{i: \xi_t^i \in u} \omega_t^i \text{Var}_{K(\xi_t^i, \cdot)} h. \quad (5.7)$$

The contribution to this mutation variance at time  $t$  from bin  $u$  is

$$\frac{\omega_t(u)}{N_t(u)} \sum_{i: \xi_t^i \in u} \omega_t^i \text{Var}_{K(\xi_t^i, \cdot)} h.$$

Increasing  $N_t(u)$  by some factor decreases the mutation variance from bin  $u$  at time  $t$  by the same factor. The mutation variance can be reduced by a factor of almost (5.6) if  $N_t(u)$  is increased by the factor (5.6) in the bins  $u$  where  $\text{Var}_{K(\xi_t^i, \cdot)} h$  is large, and if  $\text{Var}_{K(\xi_t^i, \cdot)} h$  is

small enough in the other bins that decreasing the allocation in those bins does not significantly increase the mutation variance.

Of course the variance formulas in Lemmas 3.4 and 3.5 can, in principle, more precisely describe the gain, although it is difficult to accurately estimate the values of these variances a priori outside of the  $N \rightarrow \infty$  limit. Since we focus on relatively small  $N$ , we do not go in this analytic direction. Instead we numerically illustrate the improvement from optimizing parameters in Figures 6.3 and 6.4.

### 5.3. Adaptive methods.

We have proposed handling the optimizations (4.3) and (4.6) by approximating  $K$  and  $h$  with  $\tilde{K}$  and  $\tilde{h}$ , where the latter are built by observing microbin-to-microbin transitions and solving the appropriate Poisson problem. Since  $K$  and  $h$  are fixed in time and we are doing long-time calculations, it is natural to estimate them adaptively, for instance, via stochastic approximation [38]. Thus both (4.3) and (4.6) could be solved adaptively, at least in principle. Depending on the number of microbins, it may be relatively cheap to compute  $h$  compared to the cost of evolving particles. If this is the case, it is natural to solve (4.3) on the fly. We could also perform bin searches intermittently, depending on their cost.

## 6. Numerical illustration.

In this section we illustrate the optimizations in Algorithms 5.1 and 5.2, for a simple example of a mean first passage time computation. Consider *overdamped Langevin dynamics*

$$dX_t = -V'(X_t)dt + \sqrt{2\beta^{-1}}dW_t, \quad (6.1)$$

where  $\beta = 5$ ,  $(W_t)_{t \geq 0}$  is a standard Brownian motion, and the potential energy is

$$V(x) = \begin{cases} 5(x - 7/12)^2 + 0.15 \cos(240\pi x), & x < 7/12, \\ -1 - \cos(12\pi x) + 0.15 \cos(240\pi x), & x \geq 7/12. \end{cases}$$

See Figure 6.1. We impose reflecting boundary conditions on the interval  $[0, 1]$ . We will estimate the mean first passage time of  $(X_t)_{t \geq 0}$  from  $1/2$  to  $[119/120, 1]$  using the Hill relation [5, 32]. The Hill relation reformulates the mean first passage time as a steady-state average, as we explain below.

We choose 120 uniformly spaced microbins between 0 and 1:

$$\mathcal{MB} = \{(p-1)/120, p/120\}; i = 1, \dots, 120\}.$$

(They are not actually disjoint, but they do not overlap so this is unimportant.)

The microbins correspond to the *basins of attraction* of  $V$ . A basin of attraction of  $V$  is a set of initial conditions  $x(0)$  for which  $dx(t)/dt = -V'(x(t))$  has a unique long-time limit. The microbins combine to make 3 larger *metastable sets*, defined in section 3 above. These larger metastable sets, each comprised of many smaller basins of attraction, will be called *superbasins*. *The microbins do not need to be basins of attraction: they only need to be sufficiently “small” to give useful estimates of  $K$  and  $h$ .* We choose microbins in this way to illustrate the qualitative features we expect from Markov state modeling, where the Markov states (or our microbins) are often basins of attraction. In this case, the bins might be clusters of Markov states corresponding to superbasins.

The kernel  $K$  is defined as follows. First, let  $K_{\delta t}$  be an Euler–Maruyama time discretization of (6.1) with time step  $\delta t = 2 \times 10^{-5}$ . We introduce a *sink* at the target state  $[119/120, 1]$  that recycles at a *source*  $x = 1/2$  via

$$\bar{K}_{\delta t}(x, dy) = \begin{cases} K_{\delta t}(\frac{1}{2}, dy), & x \in [119/120, 1], \\ K_{\delta t}(x, dy), & x \notin [119/120, 1]. \end{cases}$$

Then we define  $K$  as a  $t$ -skeleton of  $\bar{K}_{\delta t}$  where  $t = 10$ :

$$K = \bar{K}_{\delta t}^{10}. \quad (6.2)$$

This just means we take 10 Euler–Maruyama [37] time steps in the mutation step of Algorithm 3.1. The Hill relation [5] shows that, if  $(\bar{X}_n)_{n \geq 0}$  is a Markov chain with kernel either  $K_{\delta t}$  or  $\bar{K}_{\delta t}$  and  $\bar{\tau} = \inf\{n \geq 0 : \bar{X}_n \in [119/120, 1]\}$ , then

$$\mathbb{E}[\bar{\tau} \mid \bar{X}_0 = 1/2] = \frac{1}{\mu([119/120, 1])}.$$

where  $\mu$  is the stationary distribution of  $K$ . Thus if  $\tau = \inf\{t > 0 : X_t \in [119/120, 1]\}$ ,

$$\mathbb{E}[\tau \mid X_0 = 1/2] \approx \frac{\delta t}{\mu([119/120, 1])}.$$

By construction  $\mu([119/120, 1])$  is small (on the order  $10^{-7}$ ), so it must be estimated with substantial precision to resolve the mean first passage time.

We will estimate the mean first passage time from  $x = 1/2$  to  $x \in [119/120, 1]$ , the latter being the target state and rightmost microbin. Thus, we define  $f$  as the characteristic function of this microbin,  $f = \mathbb{1}_{[119/120, 1]}$ , so that

$$\theta_T \approx \int f d\mu = \mu([119/120, 1]).$$

Weighted ensemble then estimates the mean first passage time via

$$\text{Mean first passage time} = \mathbb{E}[\tau \mid X_0 = 1/2] \approx \frac{\delta t}{\theta_T}. \quad (6.3)$$

We illustrate Algorithm 3.1 combined with Algorithms 5.1–5.2 in Figures 6.1–6.4. For Algorithms 5.1 and 5.2, to construct  $\tilde{K}$ ,  $\tilde{\mu}$ ,  $\tilde{f}$ , and  $\tilde{h}$  as in section 5, we compute the matrix  $\tilde{K}$  using  $10^4$  trajectories starting from each microbin’s midpoint, and we define  $\tilde{f}_{120} = 1$ ,  $\tilde{f}_p = 0$  for  $1 \leq p < 119$ . The  $p$ th rows of  $\tilde{K}$ ,  $\tilde{\mu}$ ,  $\tilde{f}$ , and  $\tilde{h}$  correspond to the microbin  $[(p-1)/120, p/120]$ . In Algorithm 5.2, to simplify visualization we enforce a condition that the bins must

be connected regions. We initialize all our simulations with Algorithm 5.3, where  $N_{int} = 120$ ,  $\xi_0^i = (i - 1/2)/120$ , and  $\omega^i = 1/120$ .

In Figure 6.1, we show the bins resulting from Algorithm 5.2, and plot the terms  $\tilde{K}\tilde{h}$  and  $\tilde{K}\tilde{h}^2 - (\tilde{K}\tilde{h})^2$  appearing in the optimizations in Algorithms 5.1 and 5.2, along with the approximate steady state  $\tilde{\mu}$ . Note that the bins resolve the superbases of  $V$ . In Figure 6.2, we explore what happens when the number  $M$  of bins increases in Algorithm 5.2, finding that the bins begin to resolve the regions between superbases, favoring regions closer to the support of  $f$ . As  $M$  increases, the optimal allocation leads to particles having more children when they are near the dividing surfaces between the superbases; to see why, compare the particle allocation in Algorithm 5.1 with the bottom left of Figure 6.1.

In Figure 6.3, we illustrate weighted ensemble with the optimized allocation and binning of Algorithms 5.1 and 5.2 when the number of bins increases from  $M = 4$  to  $M = 16$ . Observe that  $M = 4$  bins is not enough to resolve the regions between the superbases, but we still get a substantial gain over direct Monte Carlo (compare with Figure 6.4). With  $M = 16$  bins we resolve the regions between superbases, further reducing the variance. The bins we use are exactly the ones in Figure 6.2.

In Figure 6.4, we compare weighted ensemble with direct Monte Carlo when  $M = 4$ . Direct Monte Carlo can be seen as a version of Algorithm 3.1 where each parent always has exactly one child,  $C_t^i = 1$  for all  $t, i$ . For weighted ensemble, we consider optimizing either or both of the bins and the allocation. When we do not optimize the bins, we consider *uniform bins*  $\mathcal{B} = \{[0, 40/120], [40/120, 80/120], [80/120, 1]\}$ . When we do not optimize the allocation, we consider *uniform allocation*, where we distribute particles evenly among the occupied bins,  $N_t(u) \approx N / \#\{u \in \mathcal{B} : \omega_t(u) > 0\}$ . Notice the order of magnitude reduction in standard deviation compared with direct Monte Carlo for this relatively small number of bins.

In Figure 6.5, to illustrate the Hill relation, we plot the weighted ensemble estimates of the mean first passage time from our data in Figures 6.3 and 6.4 against the (numerically) exact value. The weighted ensemble estimates at small  $T$  tend to exhibit a “bias” because the weighted ensemble has not yet reached steady state. As  $T$  grows, this bias vanishes and the weighted ensemble estimates converge to the true mean first passage time. In the simple example in this section, we can directly compute the mean first passage time. For complicated biological problems, however, the first passage time can be so large that it is difficult to directly sample even once. In spite of this, the Hill relation reformulation can lead to useful estimates on much shorter time scales than the mean first passage time [1, 16, 63]. Indeed this is the case in Figure 6.5, where we get accurate estimates in a time horizon orders of magnitude smaller than the mean first passage time. This speedup can be attributed in part to the initialization in Algorithm 3.1. In general, there can also be substantial speedup from the Hill relation itself, independently of the initial condition [63].

Optimizing the bins and allocation together has the best performance. We expect that, as in this numerical example, when the number  $M$  of bins is relatively small, optimizing the bins can be more important than optimizing the allocation. Optimizing only the allocation may

lead to a less dramatic gain when the bins poorly resolve the landscape defined by  $h$ . On the other hand, for a large enough number  $M$  of bins it may be sufficient just to optimize the allocation. We emphasize that *more bins is not always better*, since for a fixed number  $N$  of particles, with too many bins we end up recovering direct Monte Carlo. See Remark 5.1 above.

## 7. Remarks and future work.

This work presents new procedures for choosing the bins and particle allocation for weighted ensemble, building in part from ideas in [5]. The bins and particle allocation, together with the resampling times, completely characterize the method. Though we do not try to optimize the latter, we argue that there is no significant variance cost associated with taking small resampling times. Optimizing weighted ensemble is worthwhile when the optimized parameter choices lead to significantly more particles in important regions of state space, compared to a naive method. The corresponding gain, represented by the rule of thumb (5.6), can be expected to grow as the dimension increases. This is because in high dimensions, the pathways between metastable sets are narrow compared to the vast state space (see [58, 59] and the references in the introduction).

Though our interest is in steady-state calculations, many of our ideas could just as well be applied to a finite time setup. Our practical interest in steady state arises from the computation of mean first passage times via the Hill relation. On the mathematical side, general importance sampling particle methods with the unbiased property of Theorem 3.1 are often not appropriate for steady-state sampling. (By general methods, we mean methods for *nonreversible* Markov chains, or Markov chains with a steady state that is *not known up to a normalization factor* [40].) Indeed as we explain in our companion article [4], standard sequential Monte Carlo methods can suffer from variance explosion at large times, even for carefully designed resampling steps. So this article could be an important advance in that direction.

We conclude by discussing some open problems. The implementation of the algorithms in this article to complex, high dimensional problems will require substantial effort and modification to the software in [65]. The aim of this article is to lay the groundwork for such applications. On a more theoretical note, there remain some questions regarding parameter optimization. We could consider adaptive/nonspatial bins instead of fixed bins as in Algorithm 5.2, for instance, bins chosen via  $k$ -means on the values of  $Kh$  of the  $N$  particles. Also, we choose a number  $M$  of weighted ensemble bins and number  $N$  of particles a priori. It remains open how to pick the best value of  $M$  for a given number,  $N$ , of particles, and fixed computational resources. Using only the variance analysis above, a straightforward answer to this question can probably only be obtained in the large  $N$  asymptotic regime. More analysis is then needed, since we generally have small  $N$ . We could also optimize over both  $N$  and  $M$ , or over  $N$ ,  $M$ , and a total number  $S$  of independent simulations, subject to a fixed computational budget. We leave these questions to future work.



## Acknowledgments.

D. Aristoff thanks Peter Christman, Tony Lelièvre, Josselin Garnier, Matthias Rousset, Gabriel Stoltz, and Robert J. Webber for helpful suggestions and discussions. D. Aristoff and D.M. Zuckerman especially thank Jeremy Copperman and Gideon Simpson for many ongoing discussions related to this work.

**Funding:** This work was supported by the National Science Foundation via the awards NSF-DMS-1522398 and NSF-DMS-1818726 and the National Institutes of Health via the award GM115805.

## 8.: Appendix.

In this appendix, we describe residual resampling, and we draw a connection between mutation variance minimization and sensitivity.

### 8.1. Residual resampling.

Recall that in Algorithm 3.1 we assumed residual resampling is used to get the number of children  $(C_t^j)^{j=1, \dots, N}$  of each particle  $\xi_t^j$  at each time  $t$ . We could also use residual resampling to compute  $R_t(u) \in \mathcal{B}$  in Algorithm 5.1. For the reader's convenience, we describe residual resampling in Algorithm 8.1. Recall  $\lfloor x \rfloor$  is the floor function (the greatest integer  $x$ ).

### 8.2. Intuition behind mutation variance minimization.

The goal of this section is to give some intuition for the strategy (4.4), which chooses the particle allocation to minimize mutation variance, by introducing a connection with sensitivity of the stationary distribution  $\mu$  to perturbations of  $K$ . *In this section we make the simplifying assumption that state space is finite, and each point of state space is a microbin.* Thus  $K = (K_{pq})_{p, q \in \mathcal{M}\mathcal{B}}$  is a finite stochastic matrix. The Poisson solution

#### Algorithm 8.1

Residual resampling.

---

To generate  $n$  samples  $\{N_i; i \in \mathcal{I}\}$  from a distribution  $\{d_i; i \in \mathcal{I}\}$  with  $\sum_{i \in \mathcal{I}} d_i = 1$ :

- Define  $\delta_i = nd_i - \lfloor nd_i \rfloor$  and let  $\delta = \sum_{i \in \mathcal{I}} \delta_i$ .
- Sample  $\{R_i; i \in \mathcal{I}\}$  from the multinomial distribution with  $\delta$  trials and event probabilities  $\delta_i/\delta$ ,  $i \in \mathcal{I}$ . In detail,

$$\mathbb{P}(R_i = r_i, i \in \mathcal{I}) = \mathbb{1}_{\sum_{i \in \mathcal{I}} r_i = \delta} \frac{\delta!}{\prod_{i \in \mathcal{I}} r_i!} \prod_{i \in \mathcal{I}} (\delta_i/\delta)^{r_i}.$$

- Define  $N_i = \lfloor nd_i \rfloor + R_i$ ,  $i \in \mathcal{I}$ .
- 

$h$  defined in (3.15) satisfies  $(I - K)h = f - (f^T \mu)\mathbb{1}$  and  $h^T \mu = 0$ , or equivalently

$$h = \sum_{s=0}^{\infty} K^s \bar{f}, \quad \bar{f} = f - (f^T \mu) \mathbb{1}, \quad (8.1)$$

where  $h = (h_p)_{p \in \mathcal{M}\mathcal{B}}$ ,  $f = (f_p)_{p \in \mathcal{M}\mathcal{B}}$ ,  $\bar{f} = (\bar{f}_p)_{p \in \mathcal{M}\mathcal{B}}$ , and  $\mu = (\mu_p)_{p \in \mathcal{M}\mathcal{B}}$  are column vectors, and  $\mathbb{1} = \sum_{p \in \mathcal{M}\mathcal{B}} e_p$  is the all-ones column vector. Here,  $I$  is the identity matrix, and  $(e_p)_{p \in \mathcal{M}\mathcal{B}}$  is the column vector with 1 in the  $p$ th entry and 0's elsewhere.

We write  $\mu(Q)$  for the stationary distribution of an irreducible stochastic matrix  $Q = (Q_{pq})_{p,q \in \mathcal{M}\mathcal{B}}$ . More precisely,  $\mu(Q)$  denotes a continuously differentiable extension of  $Q \mapsto \mu(Q)$  to an open neighborhood of the space of  $\# \mathcal{M}\mathcal{B} \times \# \mathcal{M}\mathcal{B}$  irreducible stochastic matrices which satisfies  $\mu(Q)^T Q = \mu(Q)^T$  and  $\mu(Q)^T \mathbb{1} = 1$  whenever  $Q \mathbb{1} = 1$ . See [52] for details and a proof of existence of this extension. Abusing notation, we still write  $\mu$  with no matrix argument to denote the stationary distribution  $\mu(K)$  of  $K$ .

### Theorem 8.1.

Let  $\lambda(u) > 0$  be such that  $\sum_{u \in \mathcal{B}} \lambda(u) = 1$ . For each  $u \in \mathcal{B}$ ,  $\nu(u) = (\nu(u)_p)_{p \in \mathcal{M}\mathcal{B}}$  be a vector satisfying  $\nu(u)_p = 0$  for  $p \in \mathcal{M}\mathcal{B}$ ,  $\nu(u)_p = 0$  for  $p \notin u$ , and  $\sum_{p \in u} \nu(u)_p = 1$ . Let  $A(u)$  be a random matrix with the distribution

$$\mathbb{P}(A(u) = e_p e_q^T - e_p e_p^T K) = \nu(u)_p K_{pq} \text{ if } p \in u, q \in \mathcal{M}\mathcal{B}. \quad (8.2)$$

Let  $A(u)^{(n)}$  be independent copies of  $A$  for  $u \in \mathcal{B}$  and  $n = 1, 2, \dots$ . Define

$$B^{(N)} = \sqrt{N} \sum_{u \in \mathcal{B}} \frac{1}{[N\lambda(u)]} \sum_{n=1}^{[N\lambda(u)]} A(u)^{(n)}. \quad (8.3)$$

Then

$$\begin{aligned} & \lim_{N \rightarrow \infty} \mathbb{E} \left[ \left( \frac{d}{d\epsilon} \mu(K + \epsilon B^{(N)})^T \Big|_{\epsilon=0} f \right)^2 \right] \\ &= \sum_{u \in \mathcal{B}} \lambda(u)^{-1} \sum_{p \in u} \nu(u)_p \mu_p^2 \left[ (Kh^2)_p - (Kh)_p^2 \right]. \end{aligned} \quad (8.4)$$

We now interpret Theorem 8.1 from the point of view of particle allocation. Note that  $A(u)$  is a centered sample of  $K$  obtained by picking a microbin  $p \in u$  according to the distribution  $\nu(u)$ , and then simulating a transition from  $p$  via  $K$ . By a centered sample, we mean that we adjust  $A(u)$  by subtracting by its mean, so that  $A(u)$  has mean zero. The mean of  $[(d/d\epsilon)\mu(K + \epsilon A(u))^T]_{\epsilon=0} f^2$  measures the sensitivity of  $\int f d\mu = f^T \mu$  to sampling from bin  $u$  according to the distribution  $\nu(u)$ . Similarly, the mean of  $[(d/d\epsilon)\mu(K + \epsilon B^{(N)})^T]_{\epsilon=0} f^2$  measures the sensitivity of  $\int f d\mu$  corresponding to sampling from each bin  $u \in \mathcal{B}$  exactly

$\lfloor N\lambda(u) \rfloor$  times according to the distributions  $\nu(u)$ . Appropriately normalizing the latter in the limit  $N \rightarrow \infty$  leads to (8.4). The sensitivity in (8.4) is minimized over  $\lambda(u)^u \in \mathcal{B}$  when

$$N\lambda(u) = \frac{N\sqrt{\sum_{p \in u} \nu(u)_p \mu_p^2 \left[ (Kh^2)_p - (Kh)_p^2 \right]}}{\sum_{u \in \mathcal{B}} \sqrt{\sum_{p \in u} \nu(u)_p \mu_p^2 \left[ (Kh^2)_p - (Kh)_p^2 \right]}}. \quad (8.5)$$

In light of the discussion above, we think of  $N\lambda(u)^u \in \mathcal{B}$  in (8.5) as a particle allocation. Note that this resembles our formula (4.4) for the optimal weighted ensemble particle allocation. We now try to make a connection between (4.4) and (8.5).

We first consider a simple case. Suppose the bins are equal to the microbins,  $\mathcal{MB} = \mathcal{B}$ . Since we assume here that every point in state space is a microbin, this means that every point of state space is also a bin. In particular the distributions  $\nu(u)$  are trivial:  $\nu(u)_p = 1$  whenever  $p \in \mathcal{MB}$  with  $p = u$ . To make (8.5) agree with (4.4), we need  $\mu_p = \omega(u)$  when  $p = u$ . Of course this equality does not hold. It is true that  $\mu_p \approx \omega(u)$  when  $p = u$  is a reasonable approximation, *but only in the asymptotic where  $N, t \rightarrow \infty$* . To see why this is so, note that the unbiased property and ergodicity show that  $\lim_{t \rightarrow \infty} \mathbb{E}[\omega_t(u)] = \mu_p$  when  $p = u$ ; provided an appropriate law of large numbers also holds,  $\lim_{t \rightarrow \infty} \lim_{N \rightarrow \infty} \omega(u) = \mu_p$ . Recall, though, that we are interested in relatively small  $N$ , due to the high cost of particle evolution.

For the general case, where  $\mathcal{MB} \neq \mathcal{B}$  and bins contain multiple points in state space, we see no direct connection between the allocation formulas (8.5) and (4.4). Indeed, to make an analogy between this sensitivity calculation and weighted ensemble, we should have  $\nu(u)_p = \sum_{i: \xi_t^i \in p} \omega_t^i / \omega_t(u)$ . Or in other words, we should consider perturbations that

correspond to sampling from the bins according to particle weights, in accordance with Algorithm 3.1. But putting  $\nu(u)_p = \sum_{i: \xi_t^i \in p} \omega_t^i / \omega_t(u)$  in (8.5) gives something quite different from (4.4). So while the sensitivity minimization formula (8.5) is qualitatively similar to our mutation variance minimization formula (4.4), the two are not actually the same.

### 8.3. Proof of Theorem 8.1.

We begin by noting the following.

$$\text{If } v^T \mathbb{1} = 0, \text{ then } g^T(I - K) = v^T, g^T \mathbb{1} = 0 \Leftrightarrow g^T = \sum_{s=0}^{\infty} v^T K^s. \quad (8.6)$$

Like (8.1), this follows from the ergodicity of  $K$ . See, for instance, [31].

#### Lemma 8.2.

Suppose  $A$  is a matrix with  $A\mathbb{1} = 0$ . Then

$$\left. \frac{d}{d\epsilon} \mu(K + \epsilon A) \right|_{\epsilon=0} f = \mu^T A h.$$

**Proof.**—Since  $\mu(K + \epsilon A)^T(K + \epsilon A) = \mu(K + \epsilon A)^T$ , we have

$$0 = \left. \frac{d}{d\epsilon} \mu(K + \epsilon A)^T(I - K - \epsilon A) \right|_{\epsilon=0} = \left. \frac{d}{d\epsilon} \mu(K + \epsilon A)^T \right|_{\epsilon=0} (I - K) - \mu^T A.$$

Thus

$$\left. \frac{d}{d\epsilon} \mu(K + \epsilon A)^T \right|_{\epsilon=0} (I - K) = \mu^T A. \quad (8.7)$$

Moreover since  $\mu(Q)^T \mathbf{1} = 1$  for any stochastic matrix  $Q$ ,

$$0 = \left. \frac{d}{d\epsilon} \mu(K + \epsilon A)^T \mathbf{1} \right|_{\epsilon=0} = \left. \frac{d}{d\epsilon} \mu(K + \epsilon A)^T \right|_{\epsilon=0} \mathbf{1}. \quad (8.8)$$

Now by (8.1), (8.6), (8.7), and (8.8),

$$\begin{aligned} \left. \frac{d}{d\epsilon} \mu(K + \epsilon A)^T \right|_{\epsilon=0} f &= \sum_{s=0}^{\infty} \mu^T A K^s f \\ &= \mu^T A \sum_{s=0}^{\infty} K^s \bar{f} = \mu^T A h, \end{aligned}$$

where the last line above uses  $AK^s \mathbf{1} = A \mathbf{1} = 0$  to replace  $f$  with  $\bar{f} = f - \mu^T f \mathbf{1}$ .  $\square$

### Lemma 8.3.

Suppose  $(A(n))^{n=1, \dots, N}$  are matrices such that (i)  $(A(n))^{n=1, \dots, N}$  are independent over  $n$ , (ii)  $A^{(n)} \mathbf{1} = 0$  for all  $n$ , and (iii)  $\mathbb{E}[A^{(n)}] = 0$  for all  $n$ . Then

$$\mathbb{E} \left[ \left( \left. \frac{d}{d\epsilon} \mu \left( K + \epsilon \sum_{n=1}^N A^{(n)} \right)^T \right|_{\epsilon=0} f \right)^2 \right] = \sum_{n=1}^N \mathbb{E} \left[ \left( \left. \frac{d}{d\epsilon} \mu \left( K + \epsilon A^{(n)} \right)^T \right|_{\epsilon=0} f \right)^2 \right].$$

**Proof.**—Since  $\mu$  is continuously differentiable,

$$\left. \frac{d}{d\epsilon} \mu \left( K + \epsilon \sum_{n=1}^N A^{(n)} \right)^T \right|_{\epsilon=0} f = \sum_{n=1}^N \left. \frac{d}{d\epsilon} \mu \left( K + \epsilon A^{(n)} \right)^T \right|_{\epsilon=0} f. \quad (8.9)$$

By (ii)–(iii) and Lemma 8.2,  $\mathbb{E} \left[ \left. \frac{d}{d\epsilon} \mu \left( K + \epsilon A^{(n)} \right)^T \right|_{\epsilon=0} f \right] = 0$  for all  $n$ . So by (i),

$$\mathbb{E} \left[ \left. \frac{d}{d\epsilon} \mu(K + \epsilon A^{(n)})^T \right|_{\epsilon=0} f \frac{d}{d\epsilon} \mu(K + \epsilon A^{(m)})^T \right|_{\epsilon=0} f \right] = 0 \text{ if } n \neq m. \quad (8.10)$$

The result follows by combining (8.9) and (8.10).  $\square$

#### Lemma 8.4.

Let  $A(u)$  be a random matrix with the distribution (8.2). Then

$$\mathbb{E} \left[ \left. \frac{d}{d\epsilon} \mu(K + \epsilon A(u))^T \right|_{\epsilon=0} f \right]^2 = \sum_{p \in u} \nu(u)_p \mu_p^2 \left[ (Kh^2)_p - (Kh)_p^2 \right].$$

**Proof.**—Note that  $A(u)\mathbb{1} = (e_p e_q^T - e_p e_p^T K)\mathbb{1} = 0$  since  $K\mathbb{1} = 1$ . So by Lemma 8.2,

$$\frac{d}{d\epsilon} \mu(K + \epsilon A(u))^T \Big|_{\epsilon=0} f = \mu^T A(u) h.$$

From this and (8.2),

$$\begin{aligned} \mathbb{E} \left[ \left. \frac{d}{d\epsilon} \mu(K + \epsilon A(u))^T \right|_{\epsilon=0} f \right]^2 &= \sum_{p \in u} \sum_{q \in \mathcal{M}\mathcal{B}} \nu(u)_p K_{pq} \left[ \mu^T (e_p e_q^T - e_p e_p^T K) h \right]^2 \\ &= \sum_{p \in u} \nu(u)_p \mu_p^2 \sum_{q \in \mathcal{M}\mathcal{B}} K_{pq} \left[ (e_q^T - e_p^T K) h \right]^2 \\ &= \sum_{p \in u} \nu(u)_p \mu_p^2 \sum_{q \in \mathcal{M}\mathcal{B}} K_{pq} \left[ (Kh)_p^2 + h_q^2 - 2(Kh)_p h_q \right] \\ &= \sum_{p \in u} \nu(u)_p \mu_p^2 \left[ (Kh^2)_p - (Kh)_p^2 \right]. \end{aligned}$$

We are now ready to prove Theorem 8.1.

**Proof of Theorem 8.1.**—By Lemmas 8.3 and 8.4,

$$\begin{aligned} &\mathbb{E} \left[ \left. \frac{d}{d\epsilon} \mu(K + \epsilon B^{(N)})^T \right|_{\epsilon=0} f \right]^2 \\ &= N \sum_{u \in \mathcal{B}} \frac{1}{[N\lambda(u)]^2} \sum_{n=1}^{\lfloor N\lambda(u) \rfloor} \mathbb{E} \left[ \left. \frac{d}{d\epsilon} \mu(K + \epsilon A(u)^{(n)})^T \right|_{\epsilon=0} f \right]^2 \\ &= \sum_{u \in \mathcal{B}} \frac{N}{[N\lambda(u)]} \sum_{p \in u} \nu(u)_p \mu_p^2 \left[ (Kh^2)_p - (Kh)_p^2 \right]. \end{aligned} \quad (8.11)$$

The result follows by letting  $N \rightarrow \infty$ .  $\square$

## REFERENCES

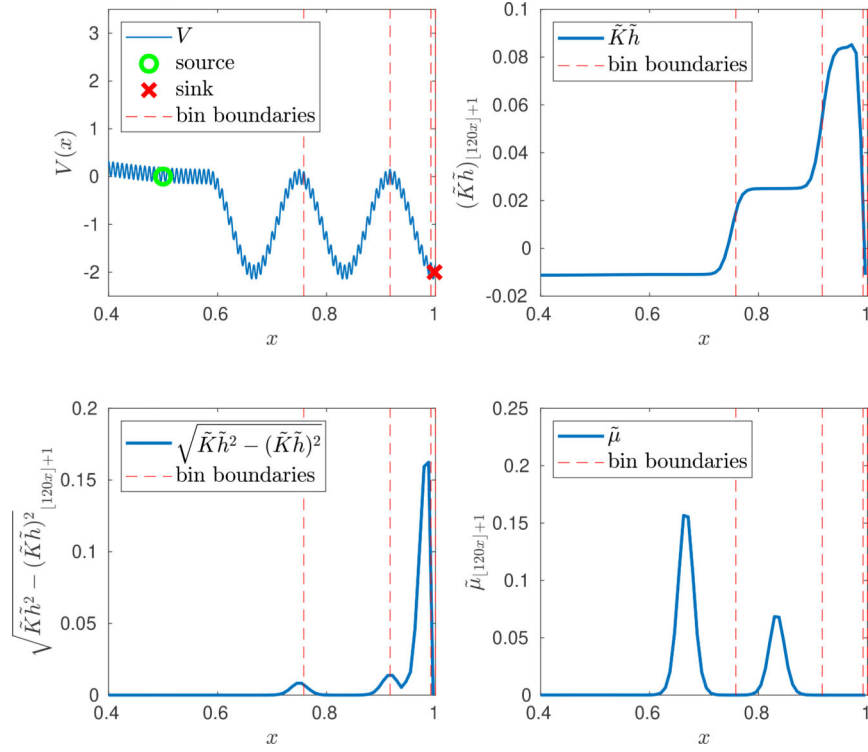
- [1]. Adhikari U, Mostofian B, Copperman J, Subramanian SR, Petersen AA, and Zuckerman DM, Computational estimation of microsecond to second atomistic folding times, *J. Amer. Chem. Soc.* 141 (2019), pp. 6519–6526. [PubMed: 30892023]

- [2]. Allen RJ, Frenkel D, and ten Wolde PR, Forward flux sampling-type schemes for simulating rare events: Efficiency analysis, *J. Chem. Phys.*, 124 (2006), 194111. [PubMed: 16729807]
- [3]. Anderson DF and Kurtz TG, Continuous time Markov chain models for chemical reaction networks, in *Design and Analysis of Biomolecular Circuits*, Springer, New York, 2011, pp. 3–42.
- [4]. Aristoff D, An Ergodic Theorem for Weighted Ensemble, preprint, <https://arxiv.org/abs/1906.00856> (2019).
- [5]. Aristoff D, Analysis and optimization of weighted ensemble sampling, *ESAIM Math. Model. Numer. Anal.*, 52 (2018), pp. 1219–1238.
- [6]. Balesdent M, Marzat J, Morio J, and Jacquemart D, Optimization of interacting particle systems for rare event estimation, *Comput. Statist. Data Anal.*, 66 (2013), pp. 117–128.
- [7]. Bello-Rivas JM and Elber R, Exact milestoneing, *J. Chem. Phys.*, 142 (2015), 03B602\_1.
- [8]. Bhatt D, Zhang BW, and Zuckerman DM, Steady-state simulations using weighted ensemble path sampling, *J. Chem. Phys.*, 133 (2010), 014110. [PubMed: 20614962]
- [9]. Bréhier C-E, Lelièvre T, and Rousset M, Analysis of adaptive multilevel splitting algorithms in an idealized case, *ESAIM Probab. Stat.*, 19 (2015), pp. 361–394.
- [10]. Bréhier C-E, Gazeau M, Goudenège L, Lelièvre T, and Rousset M, Unbiasedness of some generalized adaptive multilevel splitting algorithms, *Ann. Appl. Probab.*, 26 (2016), pp. 3559–3601.
- [11]. Cérou F, Guyader A, Lelièvre T, and Pommier D, A multiple replica approach to simulate reactive trajectories, *J. Chem. Phys.*, 134 (2011), 054108. [PubMed: 21303093]
- [12]. Chong LT, Saglam AS, and Zuckerman DM, Path-sampling strategies for simulating rare events in biomolecular systems, *Curr. Opinion Struc. Biol.*, 43 (2017), pp. 88–94.
- [13]. Chraïbi H, Dutfoy A, Galtier T, and Garnier J, Optimal Input Potential Functions in the Interacting Particle System Method, preprint, <https://arxiv.org/abs/1811.10450> (2018).
- [14]. Collet P, Martínez S, and San Martín J, *Quasi-Stationary Distributions: Markov Chains, Diffusions and Dynamical Systems*, Springer, Berlin, 2012.
- [15]. Copperman J and Zuckerman DM, Accelerated Estimation of Long-Timescale Kinetics by Combining Weighted Ensemble Simulation with Markov Model “microstates” Using Non-Markovian Theory, preprint, <https://arxiv.org/abs/1903.04673>.
- [16]. Copperman J, Aristoff D, Makarov DE, Simpson G, and Zuckerman DM, Transient probability currents provide upper and lower bounds on non-equilibrium steady-state currents in the Smoluchowski picture, *J. Chem. Phys.*, 151 (2019), 174108. [PubMed: 31703496]
- [17]. Costaoeuc R, Feng H, Izaguirre J, and Darve E, Analysis of the accelerated weighted ensemble methodology, *Discrete Contin. Dyn. Syst.*, 2013 (2013), pp. 171–181.
- [18]. Darve E and Ryu E, Computing reaction rates in bio-molecular systems using discrete macrostates, in *Innovations in Biomolecular Modeling and Simulations*, Vol. 1, Schlick T, ed., Royal Society of Chemistry, Cambridge, 2012, pp. 138–206.
- [19]. Del Moral P, *Feynman-Kac formulae: Genealogical and interacting particle approximations*, Probab. Appl., Springer, New York, 2004.
- [20]. Del Moral P and Doucet A, Particle methods: An introduction with applications, *ESAIM: Proc* 44 (2014), pp. 1–46.
- [21]. Del Moral P and Garnier J, Genealogical particle analysis of rare events, *Ann. Appl. Probab.*, 15 (2005), pp. 2496–2534.
- [22]. Dinner AR, Mattingly JC, Tempkin JO, Van Koten B, and Weare J, Trajectory stratification of stochastic dynamics, *SIAM Rev.*, 60 (2018), pp. 909–938.
- [23]. Donovan RM, Sedgewick AJ, Faeder JR, and Zuckerman DM, Efficient stochastic simulation of chemical kinetics networks using a weighted ensemble of trajectories, *J. Chem. Phys.*, 139 (2013), 09B642 1.
- [24]. Doob JL, *Stochastic Processes*, Wiley, New York, 1953.
- [25]. Doob JL, Regularity properties of certain families of chance variables, *Trans. Amer. Math. Soc.*, 47 (1940), pp. 455–486.

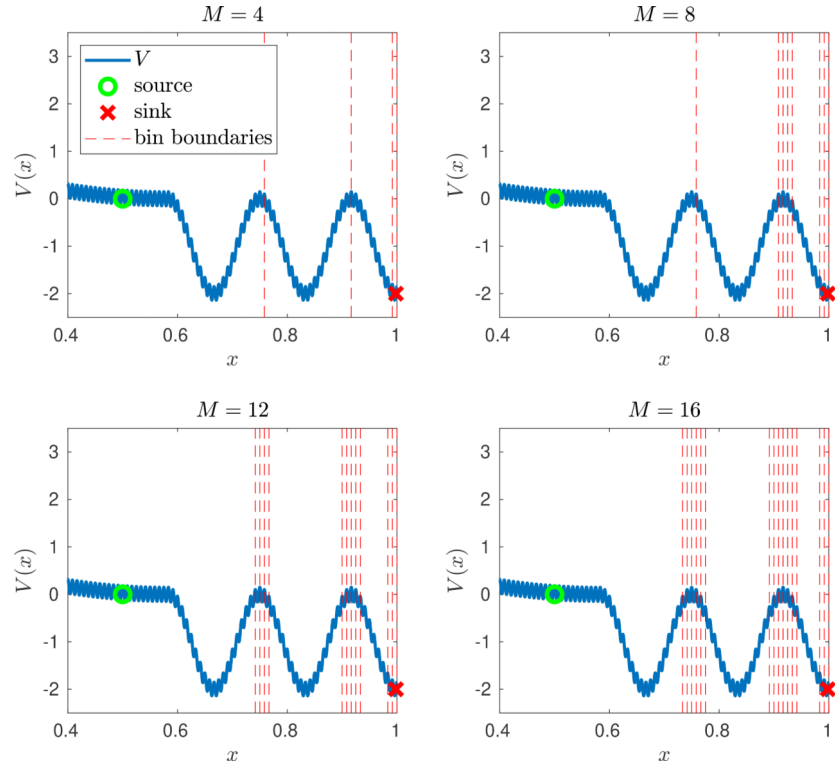
- [26]. Douc R and Cappé O, Comparison of resampling schemes for particle filtering, Proceedings of the 4th International Symposium on Image and Signal Processing and Analysis, 2005. ISPA 2005., IEEE, Piscataway, NJ, 2005, pp. 64–69.
- [27]. Douc R, Moulines E, and Stoffer D, *Nonlinear Time Series Theory, Methods, and Applications with R Examples*, CRC Press, Boca Raton, FL, 2014.
- [28]. Doucet A, De Freitas N, and Gordon N, *Sequential Monte Carlo Methods in Practice*, Stat. Eng. Inform. Sci, Springer, New York, 2001.
- [29]. Durrett R, *Probability: Theory and Examples*, Cambridge University Press, Cambridge, 2019.
- [30]. Glowacki DR, Paci E, and Shalashilin DV, Boxed molecular dynamics: Decorrelation time scales and the kinetic master equation, *J. Chem. Theory Comput*, 7 (2011), pp. 1244–1252. [PubMed: 26610120]
- [31]. Golub GH and Meyer CD, Using the QR factorization and group inversion to compute, differentiate, and estimate the sensitivity of stationary probabilities for Markov chains, *SIAM J. Algebraic Discrete Methods*, 7 (1986), pp. 273–281.
- [32]. Hill TL, *Free Energy Transduction and Biochemical Cycle Kinetics*, Dover, Mineola, NY, 2005.
- [33]. Hofmann H and Ivanyuk FA, Mean first passage time for nuclear fission and the emission of light particles, *Phys. Rev. Lett*, 90 (2003), 132701. [PubMed: 12689280]
- [34]. Huber GA and Kim S, Weighted-ensemble Brownian dynamics simulations for protein association reactions, *Biophys. J*, 70 (1996), pp. 97–110. [PubMed: 8770190]
- [35]. Husic BE and Pande VS, Markov state models: From an art to a science, *J. Amer. Chem. Soc*, 140 (2018), pp. 2386–2396. [PubMed: 29323881]
- [36]. Jacquemart D and Morio J, Tuning of adaptive interacting particle system for rare event probability estimation, *Simul. Model. Practice Theory*, 66 (2016), pp. 36–49.
- [37]. Kloeden PE and Platen E, *Numerical Solution of Stochastic Differential Equations*, Springer, Berlin, 1992.
- [38]. Kushner H and Yin GG, *Stochastic Approximation and Recursive Algorithms and Applications*, Appl. Math 35, Springer, New York, 2003.
- [39]. Lelièvre T, Two mathematical tools to analyze metastable stochastic processes, in *Numerical Mathematics and Advanced Applications*, Springer, Berlin, 2011, pp. 791–810.
- [40]. Lelièvre T and Stoltz G, Partial differential equations and stochastic methods in molecular dynamics, *Acta Numer*, 25 (2016), pp. 681–880.
- [41]. Lloyd SP, Least squares quantization in PCM, *IEEE Trans. Inform. Theory*, 28 (1982), pp. 129–137.
- [42]. Metzler R, Oshanin G, and Redner S, eds., *First-Passage Phenomena and their Applications*, World Scientific, Hackensack, NJ, 2014.
- [43]. Meyn SP and Tweedie RL, *Stability of Markovian processes II: Continuous-time processes and sampled chains*, *Adv. Appl. Probab*, 25 (1993), pp. 487–517.
- [44]. Nummelin E, On the Poisson equation in the potential theory of a single kernel, *Math. Scand*, 68 (1991), pp. 59–82.
- [45]. Pande VS, Beauchamp K, and Bowman GR, Everything you wanted to know about Markov state models but were afraid to ask, *Methods*, 52 (2010), pp. 99–105. [PubMed: 20570730]
- [46]. Rojnuckarin A, Kim S, and Subramaniam S, Brownian dynamics simulations of protein folding: Access to milliseconds time scale and beyond, *Proc. Natl. Acad. Sci. USA*, 95 (1998), pp. 4288–4292. [PubMed: 9539729]
- [47]. Rojnuckarin A, Livesay DR, and Subramaniam S, Bimolecular reaction simulation using weighted ensemble Brownian dynamics and the University of Houston Brownian dynamics program, *Biophys. J*, 79 (2000), pp. 686–693. [PubMed: 10920003]
- [48]. Sarich M, Noe F, and Schütte C, On the approximation quality of Markov state models, *Multiscale Model. Simul*, 8 (2010), pp. 1154–1177.
- [49]. Schütte C and Sarich M, *Metastability and Markov State Models in Molecular Dynamics*, Courant Lect. Notes 24, American Mathematical Society, Providence, RI, 2013.
- [50]. Stoltz G, Rousset M, and Lelièvre T, *Free Energy Computations: A Mathematical Perspective*, World Scientific, Hackensack, NJ, 2010.

- [51]. Suárez E, Lettieri S, Zwier MC, Stringer CA, Subramanian SR, Chong LT, and Zuckerman DM, Simultaneous computation of dynamical and equilibrium information using a weighted ensemble of trajectories, *J. Chem. Theory. Comput*, 10 (2014), pp. 2658–2667. [PubMed: 25246856]
- [52]. Thiede E, Van Koten B, and Weare J, Sharp entrywise perturbation bounds for Markov chains, *SIAM J. Matrix Anal. Appl*, 36 (2015), pp. 917–941. [PubMed: 26491218]
- [53]. van Erp TS, Moroni D, and Bolhuis PG, A novel path sampling method for the calculation of rate constants, *J. Chem. Phys*, 118 (2003), pp. 7762–7774.
- [54]. Vanden-Eijnden E and Venturoli M, Exact rate calculations by trajectory parallelization and tilting, *J. Chem. Phys*, 131 (2009), 044120. [PubMed: 19655850]
- [55]. Warmflash A, Bhimalapuram P, and Dinner AR, Umbrella sampling for nonequilibrium processes, *J. Chem. Phys*, 127 (2007), 154112. [PubMed: 17949137]
- [56]. Webber RJ, Plotkin DA, O’Neill ME, Abbot DS, and Weare J, Practical rare event sampling for extreme mesoscale weather, *Chaos*, 29 (2019), 053109. [PubMed: 31154764]
- [57]. Webber RJ, Unifying Sequential Monte Carlo with Resampling Matrices, preprint, <https://arxiv.org/abs/1903.12583> (2019).
- [58]. Weinan E, Ren W, and Vanden-Eijnden E, Transition pathways in complex systems: Reaction coordinates, isocommittor surfaces, and transition tubes, *Chem. Phys. Lett*, 413 (2005), pp. 242–247.
- [59]. Weinan E and Vanden-Eijnden E, Transition-path theory and path-finding algorithms for the study of rare events, *Annu. Rev. Phys. Chem*, 61 (2010), pp. 391–420. [PubMed: 18999998]
- [60]. Wouters J and Bouchet F, Rare event computation in deterministic chaotic systems using genealogical particle analysis, *J. Phys. A*, 49 (2016), 374002.
- [61]. Zhang BW, Jasnow D, and Zuckerman DM, Efficient and verified simulation of a path ensemble for conformational change in a united-residue model of calmodulin, *Proc. Natl. Acad. Sci. USA*, 104 (2007), pp. 18043–18048. [PubMed: 17984047]
- [62]. Zhang BW, Jasnow D, and Zuckerman DM, The “weighted ensemble” path sampling method is statistically exact for a broad class of stochastic processes and binning procedures, *J. Chem. Phys*, 132 (2010), 054107. [PubMed: 20136305]
- [63]. Zuckerman DM, Discrete-State Kinetics and Markov Models, <http://www.physicallensonthecell.org/discrete-state-kinetics-and-markov-models>, equation (34).
- [64]. Zuckerman DM and Chong LT, Weighted ensemble simulation: Review of methodology, applications, and software, *Annu. Rev. Biophys*, 46 (2017), pp. 43–57. [PubMed: 28301772]
- [65]. Zwier MC, Adelman JL, Kaus JW, Pratt AJ, Wong KF, Rego NS, Suárez E, Lettieri S, Wang DW, Grabe M, Zuckerman DM, and Chong LT, Westpa, <https://westpa.github.io/westpa/>, (2015).
- [66]. Zwier MC, Adelman JL, Kaus JW, Pratt AJ, Wong KF, Rego NB, Suárez E, Lettieri S, Wang DW, Grabe M, Zuckerman DM, and Chong LT, Westpa: An interoperable, highly scalable software package for weighted ensemble simulation and analysis, *J. Chem. Theory Comput*, 11 (2015), pp. 800–809. [PubMed: 26392815]

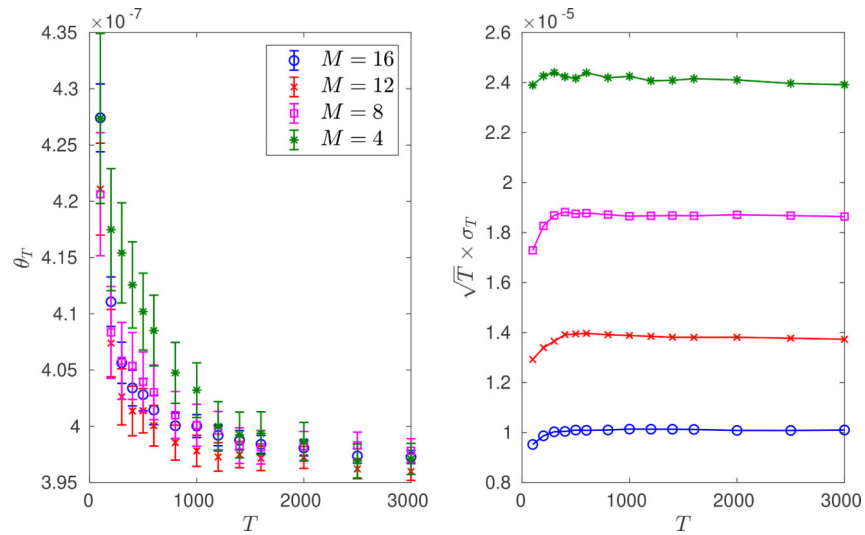


**Fig. 6.1.**

Using Algorithm 5.2 to compute the weighted ensemble bins  $\mathcal{B}$  when the number of bins is  $M = 4$ . We use  $10^6$  iterations of Algorithm 5.2 with  $\alpha = 10^5$ . Top left: Potential energy  $V$  and bin boundaries when  $M = 4$ . Top right: The vector  $\tilde{K}\tilde{h}$  defining the objective function in Algorithm 5.2, where  $\tilde{h}$  is the approximate Poisson solution. Note that  $\tilde{K}\tilde{h}$  is nearly constant on each superbasin. Bottom left: (square root of) the vector  $\tilde{K}\tilde{h}^2 - (\tilde{K}\tilde{h})^2$  involved in the mutation variance optimization in Algorithm 5.1. Bottom right: Approximate steady state distribution  $\tilde{\mu}$ . All plots have been cropped at  $x > 0.4$ , where the values of  $\tilde{K}\tilde{h}^2 - (\tilde{K}\tilde{h})^2$  and  $\tilde{\mu}$  are negligible and  $\tilde{K}\tilde{h}$  is nearly constant.

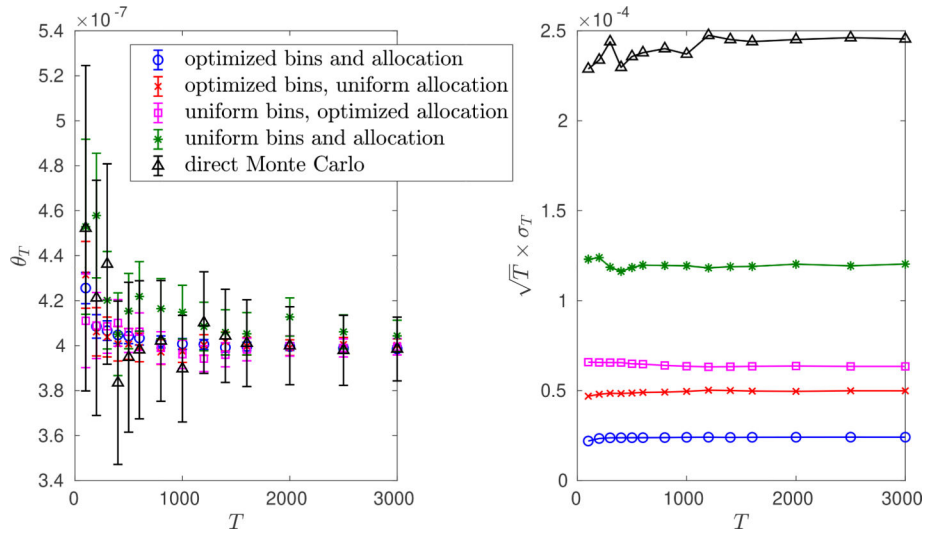
**Fig. 6.2.**

Increasing the value of  $M$  in Algorithm 5.2. Plotted are weighted ensemble bins  $\mathcal{B}$  computed using Algorithm 5.2 with  $M = 4, 8, 12, 16$ . For each value of  $M$ , we use  $10^6$  iterations of Algorithm 5.2, with  $\alpha$  tuned between  $10^5$  and  $10^6$ . Note that with increasing  $M$ , additional bins are initially devoted to resolving the energy barrier between the two rightmost superbins. Since the observable  $f$  is the rightmost microbin, this is the most important energy barrier for the bins to resolve. Note that the multiple adjacent small bins for  $M=16$  correspond to the steepest gradients of  $\tilde{K}\tilde{h}$  in the top right of Figure 6.1. All plots have been cropped at  $x > 0.4$ .

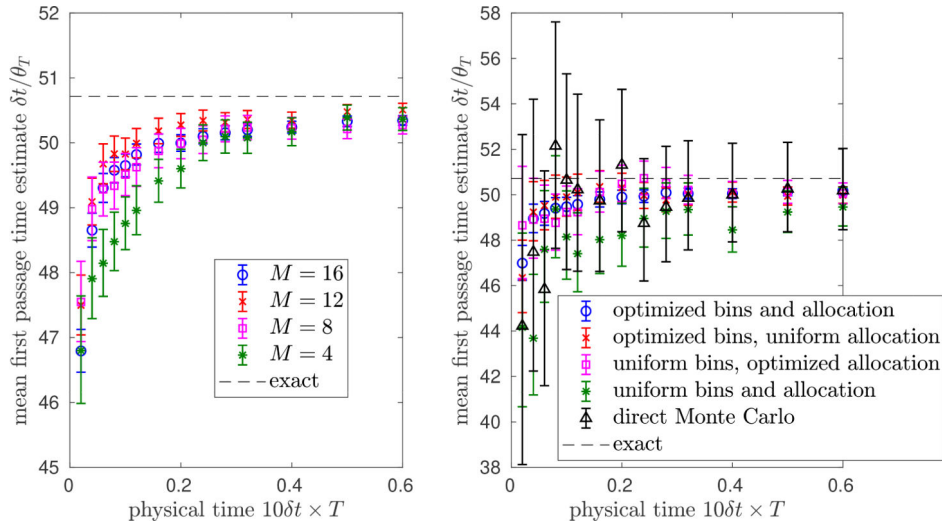
**Fig. 6.3.**

Varying the number  $M$  of weighted ensemble bins in Algorithm 5.2. Left: Weighted ensemble running means  $\theta_T$  versus  $T$  for Algorithm 3.1 with the optimal allocation and binning of Algorithms 5.1 and 5.2, when  $M = 4, 8, 12, 16$ . Values shown are averages over  $10^5$  independent trials. Error bars are  $\sigma_T/\sqrt{10^5}$ , where  $\sigma_T$  are the empirical standard deviations.

Right: Scaled empirical standard deviations  $\sqrt{T} \times \sigma_T$  versus  $T$  in the same setup. We use  $N = 40$  particles, and the bins correspond exactly to the ones in Figure 6.2. More bins are not always better, since with too many bins we return to the direct Monte Carlo regime; see Remark 5.1.

**Fig. 6.4.**

Comparison of direct Monte Carlo with weighted ensemble, where the bins and/or allocation are optimized,  $M = 4$ , and  $N = 40$ . Left: Weighted ensemble running means  $\theta_T$  versus  $T$  for Algorithm 3.1 with the indicated choices for allocation and binning. Values shown are averages over  $10^5$  independent trials. Error bars are  $\sigma_T / \sqrt{10^5}$ , where  $\sigma_T$  are the empirical standard deviations. Right: Scaled empirical standard deviations  $\sqrt{T} \times \sigma_T$  versus  $T$  in the same setup. By optimized bins, we mean the weighted ensemble bins  $\mathcal{B}$  are chosen using Algorithm 5.2 with  $M = 4$ . These optimized bins are exactly the ones plotted in the top left of Figure 6.2. Optimized allocation means the particle allocation follows Algorithm 5.1. Uniform bins means that the bins are uniformly spaced on  $[0, 1]$ , while uniform allocation means the particles are distributed uniformly among the occupied bins.

**Fig. 6.5.**

*Illustration of the Hill relation for estimating the mean first passage time via (6.3). Left: The same data as in Figure 6.3, but  $\theta_T$  is inverted and multiplied by  $\delta t$  to estimate the mean first passage time, and error bars are adjusted accordingly. Right: The same data as in Figure 6.4, but  $\theta_T$  is inverted and multiplied by  $\delta t$  to estimate the mean first passage time, and error bars are adjusted accordingly. So that the  $x$  and  $y$  axes have the same units, we consider the “physical time” on the  $x$ -axis, defined as the total number,  $T$ , of selection steps of Algorithm 3.1 multiplied by the time,  $10\delta t$ , between selection steps (see (6.2)). In both plots, the “exact” value of the mean first passage time is obtained from  $10^5$  independent samples of the first passage time  $\bar{\tau}$ . As in Figures 6.3 and 6.4, optimizing the bins and allocation has the best performance, and  $M=16$  bins is better than  $M=4, 8, 12$  bins, though more bins are not always better; see Remark 5.1.*