THE UNIVERSITY OF CHICAGO

# SHORT TRAJECTORY METHODS FOR RARE EVENT ANALYSIS AND SAMPLING

A DISSERTATION SUBMITTED TO
THE FACULTY OF THE DIVISION OF THE PHYSICAL SCIENCES
IN CANDIDACY FOR THE DEGREE OF
DOCTOR OF PHILOSOPHY

DEPARTMENT OF CHEMISTRY

BY
JOHN CHAMPLIN STRAHAN

CHICAGO, ILLINOIS
MARCH 2024

# TABLE OF CONTENTS

# LIST OF FIGURES

# LIST OF TABLES

# ACKNOWLEDGMENTS

I would like to thank my advisors Aaron Dinner and Jonathan Weare. They both have been incredible mentors, and have pushed me to produce the best work I can. I also want to thank Chatipat Lorpaiboon for innumerable helpful discussions. This work would not have been nearly as successful without his help. Finally, I want to thank my co-authors Adam Antozewski, Justin Finkel, Spencer Guo, and Bodhi Vani for their help.

# ABSTRACT

A fundamental problem in computer simulation of systems of biophysical interest is the separation of timescales. This refers to the fact that stable integration of the equations that describe molecular motion requires timesteps on the order of the fastest motion, on the order of a few femtoseconds, while events of interest, such as ligand binding, protein conformational rearrangements, etc. take place on timescales of milliseconds or more. Since even the fastest computers can barely reach the millisecond timescale by brute force, a naive computation of most 'textbook' quantities of interest such as on and off rates for ligands or free energy differences fail due to sampling error. The main focus of this dissertation is to develop novel computational methods to attack the rare-event problem and compute such quantities and more.

Our approach throughout is general: for each of the methods, we attempt to make as few assumptions on the underlying dynamics beyond Markovianity, while at the same time attempting to compute as broad a class of statistics as possible. To this end, our main tool is unbiased short trajectories. We generate a swarm of unbiased, short trajectories, each of which may be run in parallel, and then develop algorithms to recombine these short trajectories to compute our kinetic statistics of interest. We begin with a simple linear basis expansion method known as the dynamical Galerkin approximation. We reformulate the method to account for finite lag times and also develop new estimators for the reaction rate and reactive current. Next, we introduce two machine learning based alternatives to the DGA method. These use the universal function approximation properties of certain neural networks to overcome limitations in the basis set construction required for DGA. Finally, we combine our short trajectory methods with the weighted ensemble path sampling scheme to develop a rapidly converging algorithm for computing arbitrary stationary distributions for Markov processes. We illustrate our methods on well-characterized low-dimensional test systems, as well as models of both subseasonal weather and protein folding.

# CHAPTER 1

# INTRODUCTION

Molecular dynamics (MD) simulations enable atomic-resolution investigation of complex processes. These investigations are often carried out by direct simulation: the equations of motion are numerically integrated forward in time to generate trajectories (times series of atomic positions and, as needed, momenta) for as long as possible given available computational resources. Since most events of interest occur on timescales longer than those accessible by direct simulation, many enhanced sampling schemes have been developed to allow more extensive interrogation of an event of interest without sacrificing model fidelity.

The simplest and oldest class of methods restrict themselves to computing only equilibrium statistics. That is, they assume that the system obeys detailed balance, and typically that the equilibrium distribution is the Boltzmann distribution, and then the algorithm uses those assumptions to sample the configurational integral. There are a number of examples, but the two most prevalent are umbrella sampling and metadynamics [6, 7]. In umbrella sampling, one samples a large number of restricted distributions. These restricted distributions are formed by adding a series of harmonic restraints along the coordinates which resolve the slowest motions of the dynamics, such that the timescale problem is avoided in each restricted distribution and the sampling is efficient. Then, the data from each restricted distribution is combined using a number of different methods such as WHAM or MBAR [8, 7, 9]. Another common choice is metadynamics [6]. Metadynamics adds a positive Gaussian bias term at the states that the molecule has visited. This fills up metastable basins, and rapidly pushes the system to explore more basins. Both metadynamics and umbrella sampling are useful for computing free energies and some other equilibrium statistics. They are not able to compute any kinetic statistics however. To compute kinetic statistics (or equilibrium statistics for irreversible systems), several alternative approaches have been proposed. Here we will discuss transition path theory, path sampling, and Markov state modeling.

Transition path theory is a means of analyzing rare events by considering quantities which only depend on the current configuration. We begin with any Markov process $X^t$, with stationary distribution $\pi(dx)$ and two disjoint states $A$ and $B$, and define $D = (A \cup B)^{\mathsf{c}}$. In the molecular setting, the sets $A$ and $B$ could correspond to folded and unfolded states of a protein, for example. Define the forward stopping time $T^+(t) = \min\{s > t : X^s \notin D\}$ and the backward stopping time $T^-(t) = \min\{s > t : X^{-s} \notin D\}$. For any infinite trajectory $\{X^t\}_{t=-\infty}^{\infty}$, the reactive ensemble is the set of configurations $X^t$ for which $X^{T^-(t)} \in A$ and $X^{T^+(t)} \in B$.

The basic idea of TPT is to compute expectations over the reactive path ensemble by decomposing the reaction into a forward and backward fragment. Thus, the reactive density is

$$\mathbb{P}[X^t = x, X^{T^-(t)} \in A, X^{T^+(t)} \in B] = \mathbb{P}[X^t = x]\,\mathbb{P}_x[X^{T^-(t)} \in A]\,\mathbb{P}_x[X^{T^+(t)} \in B], \quad (1.1)$$

where the probability can be factorized since the future and past are independent by Markovianity. The latter two conditional probabilities are functions of $x$ which define the backward and forward committor probabilities respectively. The reactive density can therefore be written succinctly as

$$\rho_{AB}(x) = \pi(x) q^-(x) q^+(x) \quad (1.2)$$

Therefore, if the forward and backward committor probabilities can be calculated and expectations against the stationary distribution can be calculated, then one can compute expectations against the reactive density.

Another important quantity is the reactive flux from $A$ to $B$. This quantity is the average number of reactive trajectories from $A$ to $B$ per unit time. To write down a formula for this which will prove useful later, consider a set $S$ such that $A \subset S$ and $B \cap S = \varnothing$. Then the

reactive flux is given by:

$$R_{AB} = \lim_{dt\to 0^+} \frac{1}{dt} \int \pi(x) q_-(x)\, \mathbb{E}_x[(\mathbb{1}_S(X^0)\mathbb{1}_{S^c}(X^{dt}) - \mathbb{1}_{S^c}(X^0)\mathbb{1}_S(X^{dt}))q^+(X^{dt})]dx \quad (1.3)$$

This formula counts the net number of trajectories that cross the boundary of $S$ and weights the starting point by the probability of last coming from $A$, $\pi q^-$, and the end by the probability of proceeding to $B$ next, $q^+(X^{dt})$. Finally, another important object is the reactive current. We define this current to be the mean velocity of reactive trajectories at each point. Written out using a central difference, this is:

$$J_{AB}(x) = \lim_{dt\to 0^+} \frac{1}{2dt}\, \mathbb{E}[\mathbb{1}_A(X^{T^-(t)})\mathbb{1}_B(X^{T^+(t+dt)})(X^{t+dt} - X^t)(\delta(X^t - x)) + \delta(X^{t+dt} - x))] \quad (1.4)$$

$$= \lim_{dt\to 0^+} \frac{1}{2dt}\, \mathbb{E}[q^-(X^t)(X^{t+dt} - X^t)q^+(X^{t+dt})(\delta(X^t - x)) + \delta(X^{t+dt} - x))] \quad (1.5)$$

Transition path theory (TPT) is an extremely useful theoretical framework, but its utility has thus far been limited by the difficulty of computing the ingredients in the above reactive flux and current formulae. In the last decades, several methods have been developed to compute the committor probabilities and the stationary distribution needed for TPT analysis. We will introduce the two main ones that we build upon in this work, splitting methods and Markov state models.

Splitting methods are a common way to compute the stationary distributions and backward committors that TPT needs. Such methods work by branching and pruning a collection of simultaneously evolving trajectories to promote progress in a small number of order parameters (or collective variables, CVs) [10, 11, 12, 13, 14, 15]. A well-designed splitting and killing method will ensure that adequate sampling is maintained along each free energy barrier. An important example is the cloning method [16, 17]. In this method, copies of the system are split and killed according to the value of a history-dependent observable $A$,

commonly the nonequilibrium work or entropy production. The splitting and killing is done in such at way that the large deviation rate function may be computed. The large deviation theory quantities may then be used to study a variety of phenomena such as active matter, chemical reaction networks, or driven systems, to name a few applications. [18, 19]. While our contributions in this work may be used to refine cloning and other related algorithms, we will focus mainly on computing stationary distributions.

One of the most important examples of splitting and killing methods is the weighted ensemble (WE) of Huber and Kim [20]. The weighted ensemble approach proceeds by alternating between evolution and splitting/killing steps. In the evolution step, trajectories are evolved until they exit a certain pre-defined region of a CV space, known as a bin or stratum. The splitting and killing step then duplicates and kills the endpoints of the evolved trajectories in such a way that a fixed number of new points are sampled in each bin. The weights of each bin are then updated in such a way that unbiased kinetic statistics may be computed. This approach is often successful at reducing variance in estimation of very general kinetic statistics relative to brute force simulation, and has been shown to be asymptotically unbiased [13]. A drawback, however, is that the removal of initialization bias in WE and other splitting and killing methods can be extremely slow. This is a drawback we will address in Chapters 5 and 6.

A third approach to the rare event problem is Markov State Modelling (MSM). The basic idea of the MSM approach is to coarse-grain the system into a small number, on the order of 100-1000 microstates. Trajectory data is then used to estimate the transition matrix for these coarse-grained states. The transition probabilities which need to be estimated are of the form

$$P_{ij} = \mathbb{E}_{\mu}[\mathbb{1}_j(X^{\tau})\mathbb{1}_i(X^0)], \tag{1.6}$$

where $\mathbb{1}_i$ is indicator function that is 1 if $X$ is in the microstate $i$. The coarse-grained transition matrix is small enough to be handled with standard numerical linear algebra

techniques. The MSM approach has been developed from extensive literature from the last decade that shows that the eigenvalues and eigenvectors of the transition operator for the full dynamics can be approximated from the eigenvalues and vectors of the coarse-grained matrix, subject to a Markov assumption on the dynamics of the coarse-grained states [21, 22, 23, 24, 25, 26, 27, 28, 29, 30, 31, 32, 33, 34]. In particular, when the distribution $\mu$ is the stationary distribution for the dynamics, the MSM approach can compute certain unbiased kinetic statistics. It is often impossible to estimate the correlation functions (1.6) with $\mu$ chosen as that stationary distribution, however.

MSMs and other related spectral estimation methods aim to characterize the slowest dynamical features of the system (e.g., transitions between metastable states) as eigenvectors corresponding to the largest eigenvalues of the transition operator. When the goal is to study a particular event of interest, the indirect relationship between the eigenvectors of the transition operator and the specific event of interest is a weakness of the spectral estimation approach. Indeed, for many complex systems the true slowest dynamical features of the system are too slow to be of any physical interest.

The MSM approach has recently been shown to have a natural modification that allows a similar construction to approximately solve for the TPT quantities introduced above [35, 36]. This extension to the MSM approach allows one to approximate all of the necessary ingredients of the TPT reactive current and rate formulae. The main contribution of this work will be to introduce a number of methods to solve for the main ingredients in the formula (1.5), namely the stationary distribution, backward committor, and forward committor. Our approach in Chapters 2, 3, and 4 will be to use short, unbiased trajectories as data to solve the Feynman-Kac equation. In Chapter 2, we will introduce the Feynman-Kac equation and present a linear method for solving it. This method will refine and clarify the previously introduced DGA method [35]. We will then illustrate the linear method on the trp-cage miniprotein. In Chapters 3 and 4, we will introduce two machine-learning

based approximations to the Feynman-Kac equation. The first will minimize the squared residual of the Feynman-Kac equation, and the second is an inexact subspace iteration. Our subspace iteration will be illustrated on the $AIB_9$ peptide left-to-right handed helix transition. In Chapter 5, we will introduce a new short-trajectory accelerated variant of the weighted ensemble sampler which we will use to compute the stationary distribution $\pi$, as well as the backward committor. The main novel contribution of Chapter 5 is that we will show how any short-trajectory based *approximation* to the stationary distribution may be inserted into the WE algorithm to yield an unbiased sample of the stationary distribution without approximation. Finally, in Chapter 6 we will apply this accelerated WE sampler to a protein folding problem.

# CHAPTER 2

# LINEAR METHODS AND APPLICATION TO THE TRP CAGE MINIPROTEIN

## 2.1 Introduction

In the original work introducing the DGA method, [35], Thiede et. al. compared diffusion map [37] and indicator basis sets for predicting mean first-passage times and committors for the Müller-Brown model [2] and the folding of the protein Fip35 using six long equilibrium trajectories from D. E. Shaw Research [38]. Because there were only a few folding events within those trajectories, it was difficult to assess the performance of the method. One goal of the present study is to generate a protein folding data set that enables robust application of the approach and to compare different basis sets and estimators systematically. To this end, we study the trp-cage miniprotein, a 20-residue fast-folding artificial sequence (asn-leu-tyr-ile-glu-trp-leu-lys-asp-gly-gly-pro-ser-ser-gly-arg-pro-pro-pro-ser) that has been studied extensively both experimentally and computationally [39, 1, 40, 41, 42, 43, 44]. In solution at 298 K, the protein folds on a 4 $\mu$s timescale and unfolds on a 12 $\mu$s timescale [39], which makes these processes difficult but not impossible to simulate directly. In particular, D. E. Shaw Research produced a 208 $\mu$s equilibrium simulation of the K8A mutant of trp-cage using the Anton supercomputer with the CHARMM 22* force field [42]. Although, like the Fip35 data, this trajectory contains relatively few folding events, it has been the subject of previous MSM [43] and variational approach for Markov processes (VAMP) studies [44]. These earlier studies serve as valuable points of comparison and enable us to identify CVs that provide good control over sampling. Though DGA does not depend directly on any choice of CV, its performance is strongly affected by the quality of the data set of sampled trajectories. We use our chosen set of CVs together with enhanced sampling methods to generate a new data set comprised of many short trajectories that are distributed evenly

throughout the CV space.

In this chapter, we reformulate DGA in terms of the transition operator of the underlying Markov process. This has two primary advantages relative to our previous formulation in terms of the generator of the process [35]. First, it clarifies the role of lag time in DGA estimates, showing that correctly constructed estimators should have no dependence on lag time in the infinite-basis, infinite-sampling limit. Second, the formulation in terms of the transition operator leads directly to estimators that correctly account for boundary conditions by stopping underlying trajectories appropriately. Using our improved DGA estimators we introduce new estimators for TPT reaction rates and reactive currents. To make computation of the reactive current tractable and the result readily interpretable, we introduce a projection formula for the reactive current onto a CV space which allows us to assign relative weights to transition paths in arbitrary CV spaces. We also introduce a new procedure for constructing a basis set from arbitrary molecular features (here, primarily pairwise distances between $C_\alpha$ atoms, though we also explore CVs with delay embedding) and compare it with two basis sets that are used widely in the MSM literature: indicator functions on molecular features and indicator functions on time-lagged independent component analysis (TICA) coordinates [45, 46, 47]. We show that our DGA estimators with selected basis sets can robustly yield remarkably good agreement with published results for committors and pathways, even though the total simulation time of our trp-cage data set is only 30 $\mu$s, with a maximum trajectory length of 30 ns. The projection of the reactive currents on CVs facilitates both visualization and quantification of information about pathways, enabling immediate identification of the defining properties of transition states. This makes our approach an efficient one for exploring mechanisms.

## 2.2 Long time phenomena from short trajectory data

In this section, we introduce key dynamical statistics and explain how they can be defined in terms of an evolution operator (Section 2.2.1). An emphasis on forms that lead directly to practical and accurate numerical estimators causes several departures from the standard presentation of this material. We present our approach for solving the operator equations numerically by Galerkin (basis expansion) approximation [35] (Section 2.2.2), and distinguish forward-in-time statistics (Section 2.2.2) from backward-in-time statistics involving the adjoint of the evolution operator (Section 2.2.2); this is followed by a discussion of basis sets (Section 2.2.2) and an approach for constructing an approximately Markovian process when the molecular representation does not adequately capture the dynamics (delay embedding, Section 2.2.2). Finally, guided by TPT, we combine the dynamical statistics estimated by DGA to yield approximations of reaction rates and currents. (Section 2.2.3).

### *2.2.1 The transition operator and Feynman-Kac representation*

The dynamics of a Markov process $X(t)$ can be encoded in its associated transition operator, $\mathcal{T}^t$, which specifies the evolution of the expectation of a function $f$ over some interval of time $t \geq 0$:

$$\mathcal{T}^t f(x) = \mathbb{E}\left[f(X^t) \,|\, X^0 = x\right]. \tag{2.1}$$

The time index $t$ can be continuous or discrete. The transition operator (also known as the Koopman operator), and in particular its eigenvectors and eigenvalues, are the key quantities in well-established methods for discovering slowly decorrelating features of a Markov process [48]. The transition operator is also central to the DGA approach [35]. However, in DGA, instead of estimating the spectrum of the transition operator, the goal is to solve linear equations representing certain conditional expectations.

In ref. 35, we presented DGA in terms of the generator which, for a continuous time

process is defined by the limit:

$$\mathcal{L}f(x) = \lim_{t\to 0} \frac{\mathcal{T}^t f(x) - f(x)}{t}. \tag{2.2}$$

For a discrete time process the limit is removed and $t$ in (2.2) is replaced by the unit of a single time step. A presentation in terms of the generator has the advantage that it results in very concise equations for quantities of interest. For example, consider the (forward) committor, $q_+(x)$, which is the probability of entering a product state $B$ before a reactant state $A$ starting from $x \notin A \cup B$:

$$q_+(x) = \mathbb{P}[X^{T^+(0)} \in B | X^0 = x], \tag{2.3}$$

where $T^+(0) = \min\{t \geq 0 : X^t \in A \cup B\}$ is the time of first entrance into $A \cup B$. For $x \in A$, $q_+(x) = 0$, and, for $x \in B$, $q_+(x) = 1$. The committor satisfies the Feynman-Kac relation

$$\mathcal{L}q_+(x) = 0 \quad \text{for} \quad x \notin A \cup B, \qquad q_+(x) = \mathbb{1}_B(x) = \begin{cases} 1, & x \in B \\ 0, & x \notin B \end{cases} \quad \text{for} \quad x \in A \cup B \tag{2.4}$$

(see Eqs. (18) and (19) of ref. 35).

In this article we choose to work directly with the transition operator instead of the generator because it facilitates the implementation of numerical formulas. It also greatly simplifies our description of TPT and clarifies the relationship between DGA and the well-established VAC approach to approximating spectral properties of the transition operator (see ref. 48). In the case of the committor, we integrate (2.4) until a chosen time $\tau$ to obtain the equivalent form of the Feynman-Kac relation,

$$\mathcal{S}^\tau q_+(x) - q_+(x) = 0 \quad \text{for} \quad x \notin A \cup B, \qquad q_+(x) = \mathbb{1}_B(x) \quad \text{for} \quad x \in A \cup B. \tag{2.5}$$

In this expression we have introduced the notation $\mathcal{S}^t$ for the transition operator of the stopped process $X^{t\wedge T^+(0)}$, i.e.,

$$\mathcal{S}^t f(x) = \mathbb{E}\left[ f(X^{t\wedge T^+(0)}) \,|\, X^0 = x \right]. \tag{2.6}$$

Here and below $t \wedge T^+(0) = \min\{t, T^+(0)\}$, indicating that the evolution process does not proceed beyond escape.

For a more general domain $D$ and $T^+(s) = \min\{t \geq s : X^t \notin D\}$, the conditional expectation

$$u(x) = \mathbb{E}\left[ \Psi(X^{T^+(0)}) - \int_0^{T^+(0)} \Gamma(X^t) dt \,\middle|\, X^0 = x \right] \quad \text{for} \quad x \in D \tag{2.7}$$

solves the equation

$$\mathcal{S}^\tau u(x) - u(x) = \int_0^\tau \mathcal{S}^t \Gamma(x) dt \quad \text{for} \quad x \in D, \qquad u(x) = \Psi(x) \quad \text{for} \quad x \notin D. \tag{2.8}$$

To obtain (2.5) for the committor, choose $D = (A \cup B)^{\mathrm{c}}$, $\Psi = \mathbb{1}_B$, and $\Gamma = 0$. In (2.7) and (4.5) we assume for simplicity that $\Gamma(x) = 0$ for $x \notin D$. For a discrete-time process the time integral in these expressions should be interpreted as a sum.

Crucially, (4.5) holds for any choice of $\tau \geq 0$ including relatively small values. For very large values of $\tau$, (4.5) converges to (2.7). However, in most cases of interest, the escape time $T^+(0)$ is very large, making estimation of $u$ in (2.7) by direct simulation of sample trajectories of $X^t$ prohibitively expensive. In the context of DGA, the significance of (4.5) is that it expresses $u$ in terms of an expectation over short trajectories. The catch is that (4.5) must be "inverted" to solve for $u$.

### *2.2.2 Dynamical Galerkin Approximation (DGA)*

We now describe a Galerkin approach to approximating conditional expectations from short trajectory data. We first introduce a "guess" function $\psi$ that satisfies the boundary conditions (i.e., $\psi(x) = \Psi(x)$ for $x \notin D$). Our approximation has the form

$$u(x) \approx \psi(x) + \sum_{j=1}^{n} \phi_j(x) v_j, \tag{2.9}$$

where $\{\phi_j(x)\}$ is a set of $n$ basis functions satisfying $\phi_j(x) = 0$ for $x \notin D$, and $v$ is a vector of $n$ coefficients.

## Forward-in-time predictions

We begin by approximating predictions of quantities forward-in-time as in (2.7) by expanding the solution $u$ of (4.5) at a particular user chosen value of $\tau$ called the lag time. While the solution $u$ itself is independent of $\tau$ in (4.5), the quality of our approximation of $u$ with a finite basis may depend on the choice of lag time (even in the absence of sampling error). A similar phenomenon has recently been explained in detail in the context of the VAC algorithm [48]. Substituting (2.9) into (4.5), multiplying by $\phi_i$ and integrating over the distribution of sampled points $\mu$ to form the inner product $\langle f, g \rangle_\mu = \int f(x) g(x) \mu(dx)$, we obtain the linear system of equations:

$$(C^\tau - C^0) v = r^\tau, \tag{2.10}$$

with matrices $C^s \in \mathbb{R}^{n \times n}$ for $s = 0, \tau$,

$$C^s_{ij} = \langle \phi_i, \mathcal{S}^s \, \phi_j \rangle_\mu \tag{2.11}$$

and vector $r^\tau \in \mathbb{R}^n$,

$$r_i^\tau = \left\langle \phi_i, \psi(x) - \mathcal{S}^\tau \psi(x) + \int_0^\tau \mathcal{S}^t \Gamma(x) dt \right\rangle_\mu . \tag{2.12}$$

Given (4.39) and (2.12), (2.10) can be readily solved for $v$ by standard methods of linear algebra.

In models that represent molecules with high fidelity, (4.39) and (2.12) cannot be evaluated directly because a closed form of $\mathcal{S}^\tau$ is not known. DGA overcomes this issue by approximating the action of the transition operator using short molecular dynamics trajectories: if $X^0$ is a sample drawn from $\mu$ and $\{X^t\}_{t=0}^\tau$ is a trajectory segment of length $\tau$ starting from $X^0$, then we can estimate $C_{ij}^s$ (for $s = 0, \tau$) and $r_i^\tau$ as

$$C_{ij}^s \approx \frac{1}{M} \sum_{m=1}^{M} \phi_i(X_m^0) \phi_j(X_m^{s \wedge T_m^+(0)}) \tag{2.13}$$

$$r_i^\tau \approx \frac{1}{M} \sum_{m=1}^{M} \phi_i(X_m^0) \left( \psi(X_m^0) - \psi(X_m^{s \wedge T_m^+(0)}) + \Delta \sum_{p=0}^{(\tau \wedge T_m^+) - 1} \Gamma(X_m^{p\Delta}) \right) \tag{2.14}$$

where $m$ indexes trajectory segments, $\Delta$ is the sampling interval. To avoid overhead, it is advantageous to generate trajectories much longer than $\tau$ (but still much shorter than typical values of $T^+$) and use a rolling window to generate short trajectories of length $\tau$. We further note that in practice configurations are not saved at every molecular dynamics step. This limits the resolution of both the lag time and the stopping time, which we take to be the time of the first saved configuration outside the domain $D$.

## Adjoints, the steady state, and backward-in-time predictions

To compute many important quantities we need not only to solve equations involving the transition operator but also equations involving its adjoint $(\mathcal{T}^t)^\dagger_\mu$ in the $\mu$-weighted inner product, which by definition satisfies

$$\int f(x)\mathcal{T}^t g(x)\,\mu(dx) = \int g(x)(\mathcal{T}^t)^\dagger_\mu f(x)\,\mu(dx). \tag{2.15}$$

One such equation is for the change of measure $w = d\pi/d\mu$, which can be used to reweight from the sampling distribution $\mu$ to the stationary distribution $\pi$:

$$\int f(x)\pi(dx) = \int f(x)w(x)\mu(dx), \tag{2.16}$$

assuming $\mu$ and $w$ are normalized such that $\int w(x)\mu(dx) = 1$. Owing to the time translational invariance of averages over the stationary distribution $\pi$, (2.15), and (2.16), the change of measure satisfies the equation

$$(\mathcal{T}^\tau)^\dagger_\mu w(x) - w(x) = 0. \tag{2.17}$$

(2.17) can be solved analogously to (4.5), but, in this case, there are no boundary conditions. The introduction of a basis leads to a linear system of equations of the form

$$(\bar{C}^\tau - \bar{C}^0)^\top v = 0, \tag{2.18}$$

with $\bar{C}^s$ (for $s = 0, \tau$) differing from $C^s$ only in the choice of basis (which is no longer restricted to $D$) and the use of $\mathcal{T}^s$ in place of $\mathcal{S}^s$; $\top$ denotes the transpose. We note that by including $\phi_1(x) = 1$ in the basis we can guarantee that the equation for $v$ has a solution.

Given an approximate $w$, (2.16) can be computed as

$$\int f(x)\pi(dx) \approx \sum_{j=1}^{M} f(X_j^0)w(X_j^0), \tag{2.19}$$

with the weights normalized such that

$$\sum_{j=1}^{M} w(X_j^0) = 1. \tag{2.20}$$

That the change of measure can be estimated from short nonequilibrium trajectory data was previously observed in ref. 29.

Another important quantity expressible in terms of an equation involving an adjoint of the transition operator is the backwards committor

$$q_-(x) = \mathbb{P}\left[X^{T^-(0)} \in A \,|\, X^0 = x\right], \quad T^-(s) = \max\{t \le s :\, X^t \in A \cup B\}, \tag{2.21}$$

for $x \notin A \cup B$, where $X^t$, $t \le 0$ is the steady-state backward-in-time process governed by the transition operator

$$\mathcal{T}^{-t} f(x) = (\mathcal{T}^t)^\dagger_\pi f(x) = \frac{1}{w(x)} (\mathcal{T}^t)^\dagger_\mu [fw](x) \tag{2.22}$$

(the last equality can be verified using (2.15)). The backward committor is the probability that a trajectory currently at position $x$ last came from the reactant state $A$ rather than the product state $B$. It satisfies the Feynman-Kac relation

$$\mathcal{S}^{-\tau} q_-(x) - q_-(x) = 0 \;\text{ for }\; x \notin A \cup B, \qquad q_-(x) = \mathbb{1}_A(x) \;\text{ for }\; x \in A \cup B. \tag{2.23}$$

Consistent with our definition of $\mathcal{S}^t$ above, $\mathcal{S}^{-t}$ is the transition operator for the steady-state backward-in-time process stopped upon first entrance in $A \cup B$.

To expand and approximate $q_-$ according to the DGA recipe described above, we need to estimate $\mu$-weighted inner products involving $\mathcal{S}^{-t}$. To that end we note that, as long as $g = 0$ on $A \cup B$,

$$\left\langle g, \mathcal{S}^{-t} f \right\rangle_\mu = \int \mathbb{E}\left[ f(X^{T^-(t)}) \frac{g(X^t)}{w(X^t)} \,\middle|\, X^0 = x \right] w(x)\mu(dx) \tag{2.24}$$

We provide a derivation of (2.24) in Appendix 2.6.1. Just as for the forward committor, we expect that use of a sampling measure $\mu$ with high resolution in transition regions will lead to higher approximation accuracy (i.e., better ability of a finite basis to capture the dynamics). However, in our experience the factor of $w^{-1}(X^t)$ in (2.24) leads to significant sampling errors for larger values of $t$. For our backward committor calculation we therefore weight inner products by $\pi$, using the formula

$$\left\langle g, \mathcal{S}^{-t} f \right\rangle_\pi = \int \mathbb{E}\left[ f(X^{T^-(t)}) g(X^t) \,\middle|\, X^0 = x \right] w(x)\mu(dx). \tag{2.25}$$

(2.25) allows inner products involving $\mathcal{S}^{-t}$ to be computed using forward trajectories of $X$ initiated according to $\mu$, i.e., exactly the same ingredients required to make forward-in-time predictions by DGA.

Following our procedure for forward quantities outlined in Section 2.2.2, given a guess function $\psi$ satisfying $\psi = 1$ on $A$ and $\psi = 0$ on $B$ and basis functions $\phi_j$ that are zero on $A \cup B$, we can build an approximation

$$q_-(x) \approx \psi(x) + \sum_{j=1}^{n} \phi_j(x) v_j \tag{2.26}$$

by solving

$$(C^{-\tau} - C^0)v = r^{-\tau} \tag{2.27}$$

with

$$C_{ij}^{-\tau} = \left\langle \phi_i, \mathcal{S}^{-\tau}\, \phi_j \right\rangle_\pi = \int \mathbb{E}\left[ \phi_j(X^{T^-(\tau)}) \phi_i(X^\tau) \,\middle|\, X^0 = x \right] w(x)\mu(dx) \tag{2.28}$$

and

$$\begin{aligned} r_i^{-\tau} &= \langle \phi_i,\, \psi \rangle_\pi - \left\langle \phi_i,\, \mathcal{S}^{-\tau}\, \psi \right\rangle_\pi \\ &= \int \mathbb{E}\left[ \left( \psi(X^\tau) - \psi(X^{T^-(\tau)}) \right) \phi_i(X^\tau) \,\middle|\, X^0 = x \right] w(x)\mu(dx) \end{aligned} \tag{2.29}$$

where the second equality in each display follows from (2.25).

Along with the forward committor $q_+$ and the stationary change of measure $w$, the backward committor is a key ingredient of TPT. In Section 2.2.3 we describe how DGA estimates of these quantities can be combined with TPT to reveal key properties of steady-state transition paths from the reactant state $A$ to the product state $B$. However, before that, we complete our presentation of DGA with a discussion of molecular representations and basis sets, with emphasis on those that we employ in the present study to analyze trp-cage miniprotein unfolding and folding.

## Basis functions

A key determinant of the performance of DGA is the choice of basis set. Constructing a basis set that respects the boundary conditions of the problem and captures the dynamics with relatively few functions requires care. Here we discuss how we generated the basis sets that we compare later in our numerical experiments, and explain why we chose them over alternatives.

In addition to the choice of functions, there is also a choice of molecular representa-

tion (i.e., the features that serve as inputs to the functions). Although molecular dynamics trajectories are generally recorded as sequences of Cartesian coordinates, the inputs to the basis functions are generally internal coordinates. This removes the effects of trivial translations and rotations, and it can improve the statistics. The internal coordinates that we use are pairwise distances between all $C_\alpha$ atoms, except those pairs which are less than three sequence positions apart; for trp-cage, there are 153 such distances. In other words, the process $X(t)$ to which we apply DGA (and TPT) is the length 153 vector of pairwise distance values. In our tests we found that including additional features, such as backbone dihedral angles, did not improve performance. We assume that the reactant state $A$ and product state $B$ of interest can be characterized in terms of these variables. We construct basis functions of these variables that satisfy the homogeneous boundary condition on the domain $D = (A \cup B)^{\mathrm{c}}$.

In this work, we compare three choices of basis set: indicator functions on the pairwise distances, indicator functions constructed on the top 10 TICA coordinates[45, 46, 47] computed from the pairwise distances at a lag time of 0.5 ns, and smooth functions of pairwise distances that satisfy the boundary conditions. We refer to these henceforth as the distance indicator, TICA indicator, and modified distance basis sets. We constructed the distance indicator and TICA indicator basis sets and their guess functions as follows:

1. For the distance indicator basis set, we constructed 200 indicator functions by mini-batch $k$-means clustering as implemented in PYEMMA on the values of the 153 pairwise distances. For the TICA indicator basis set, the clustering was performed on the top 10 TICA coordinates constructed on the pairwise distances.

2. We retained all resulting indicator functions with non-zero regions fully contained in $(A \cup B)^c$ as the basis set. We split any indicator functions with non-zero regions overlapping with $A$ or $B$, and we redefined them to be non-zero only in the portions in $(A \cup B)^c$. For the change of measure calculations, boundary conditions are not present,

so we used all indicator functions unmodified.

3. For the forward committor calculation we took the the guess function to be $\psi(x) = \mathbb{1}_B(x)$. For the backward committor calculation we took the guess function to be $\psi(x) = \mathbb{1}_A(x)$.

With an indicator basis, the DGA and MSM estimator (with appropriate state definitions) of the forward committor $q_+$ and change of measure $w$ become similar [35]. We note however that the DGA (as formulated here) and MSM approaches diverge both in DGA's use of stopped trajectories and in the way $q_+$ and $w$ (and $q_-$) are used to estimate TPT quantities as described in Section 2.2.3.

We constructed the distance basis set and its guess function as follows:

1. We computed $d_A$ and $d_B$ as the distance in feature space (i.e. in 153-dimensional Euclidean space) to the sampled points in states $A$ and $B$, respectively.

2. We set $h(x) = d_A d_B/(d_A + d_B)^2$, which obeys the homogeneous boundary conditions by construction.

3. We computed basis functions obeying the boundary conditions by multiplying each coordinate of the pairwise distance vector $x$ by $h(x)$: $\phi_i(x) = x_i\, h(x)$. For the change of measure calculation, we use $\phi_i = x_i$ and add the constant function into the set of chosen features.

4. To remove any linear dependencies introduced by enforcing the boundary conditions, and to ensure numerical stability, we orthogonalized the basis set $\phi_i$ with respect to the sampling measure (up to sampling error) using a singular value decomposition.

5. For the forward committor calculation we took the guess function to be $\psi(x) = d_B^2/(d_A + d_B)^2$. For the backward committor calculation we took the guess function to be $\psi(x) = d_A^2/(d_A + d_B)^2$.

Although here we use the backbone pairwise distances, we note that this construction procedure could be used to generate basis sets obeying the homogeneous boundary conditions for a choice of variables other than the pairwise distances such as dihedral angles, radial basis functions, or soft indicator functions.

The indicator and TICA basis sets are the most widely used in the MSM literature. Various alternatives have been proposed specifically in the context of spectral estimation [49, 50, 51, 52]. In our previous work [35], we considered a basis set based on diffusion maps [37]. Due to the size of our trp-cage data set ($\sim 10^6$ datapoints), the $O(N^3)$ scaling of the matrix diagonalization associated with the diffusion map proved prohibitively computationally costly without subsampling and out of sample extension.

## Delay Embedding

Application of DGA as described so far assumes that the underlying process $X^t$ is Markovian; the conditional expectations that DGA seeks to approximate are not fully defined if $X^t$ is not Markovian. Yet, in the previous section we described an approach to building a basis set for DGA consisting of functions of only a subset of the full collection of variables (selected pairwise distances). Though the dynamics of this subset are not strictly Markovian, in Section 2.4 we show that, at least in the specific context of the trp-cage system, the remaining degrees of freedom relax sufficiently fast that DGA yields accurate results.

However, in some circumstances, one may only have access to a small number of variables that are insufficient to specify the dynamics. This situation is typical when the data are from an experiment. In this case, we can construct a more expressive representation of the system from time-lagged images, i.e., if $X^t$ is not itself Markovian we can instead apply DGA to the augmented process $\bar{X}^t = (X^{t-M\delta}, X^{t-(M-1)\delta}, \ldots, X^t)$ [35]. For large enough $M$ one can

expect $\bar{X}^t$ to be nearly Markovian. State space augmentation was also used in the history-augmented MSM (haMSM) approach of ref. 53 to obtain accurate MFPT estimates at all lag times. Our approach differs in that we construction our basis on the delay embedded space, whereas in the haMSM approach the transition probabilities are conditioned on visiting multiple clusters in sequence. In principle, one can explicitly include memory as defined by the Mori-Zwanzig formalism [54], though our delay-embedding approach is computationally more straightforward because it does not require choosing a form for the memory kernel and then estimating it from data, both of which are quite challenging [55, 56].

In Section 2.4.5, we show that delay embedding can significantly improve DGA estimates when a small number of CVs is used to characterize molecular configurations. Writing the values of the CVs at time $t$ as the vector $X^t$, we construct the delay embedded process $\bar{X}^t$. We then construct a basis set following the recipe in Section 2.2.2 for the modified distance basis, but replacing $X$ with $\bar{X}$. We then extend other functions $f$ of the CV space to the delay-embedded space by $f(\bar{X}^t) = f(X^{t-\lfloor M/2 \rfloor \delta})$. This allows us to extend the states $A$ and $B$ (which can both be defined in terms of the CVs) as well as the functions $a$ and $b$ in (2.7). We then apply DGA as outlined above directly on the delay-embedded space.

### *2.2.3 Reaction rates and currents*

Estimates of rates from simulations are frequently of interest because they can be compared directly with experimental measurements, and they can provide indirect information about mechanisms. TPT in principle provides not just rate estimates but reactive currents or fluxes, which provide direct information about mechanisms. However, previous calculations of reactive current have been limited to toy models and depictions of the reactive flux between metastable states can been difficult to interpret. Working within the TPT framework and building upon DGA approximations of $w$, $q_+$, and $q_-$, in this section we introduce robust estimates of the reaction rate and of an easily interpretable projection of the reactive current

onto CVs (as opposed to over the network of metastable states).

There are various expressions for the rate in TPT. One approach is based on the rate at which trajectories transition from $A$ to $B$, $R_{AB}$. If $U$ is any set for which $A \subset U$ and $B \subset U^{\mathrm{c}}$ then, for a continuous time process,

$$\begin{aligned} R_{AB} &= \lim_{t\to 0} \frac{1}{t} \int \Big( \mathbb{1}_U(x) \mathcal{T}^t [\mathbb{1}_{U^{\mathrm{c}}}\, q_+](x) - \mathbb{1}_{U^{\mathrm{c}}}(x) \mathcal{T}^t [\mathbb{1}_U\, q_+](x) \Big)\, q_-(x) \pi(dx) \\ &= \lim_{t\to 0} \frac{1}{t} \int \Big( \mathbb{1}_U(x) \mathcal{T}^t q_+(x) - \mathcal{T}^t [\mathbb{1}_U\, q_+](x) \Big)\, q_-(x) \pi(dx), \end{aligned} \tag{2.30}$$

where the second line is obtained by noting that $\mathbb{1}_{U^{\mathrm{c}}}(x) = 1 - \mathbb{1}_U(x)$. Here and below, for a discrete time process the limit is removed and $t$ is replaced by the unit of a single time step.

Expression (2.30) simply counts trajectories with forward crossings of the surface dividing $U$ and $U^{\mathrm{c}}$, weighted by their probabilities that they start in $A$ and end in $B$. Consequently, when using this formula to estimate rates from data, only those trajectories that cross the surface dividing $U$ and $U^{\mathrm{c}}$ contribute. Because these trajectories are generally a small fraction of the data, this results in relatively large variances in estimates. We can obtain considerably better estimates by considering the isocommittor surfaces: $U(z) = \{x : q_+(x) \leq z,\ x \in D\}$ for $z \in (0,1)$, and noting that $R_{AB}$ is independent of $z$. Integrating (2.30) with respect to $z$[57], then exchanging integrals over $z$ with applications of $\mathcal{T}^t$ and noting that $\int_0^1 \mathbb{1}_{[0,z]}\big(q_+(x)\big)\, dz = 1 - q_+(x)$, we find that

$$\begin{aligned} R_{AB} &= \lim_{t\to 0} \frac{1}{t} \int \int_0^1 \Big\{ \mathbb{1}_{[0,z]}\big(q_+(x)\big) \mathcal{T}^t q_+(x) - \mathcal{T}^t \Big[ q_+\, \mathbb{1}_{[0,z]}(q_+) \Big](x) \Big\} dz\, q_-(x) \pi(dx) \\ &= \lim_{t\to 0} \frac{1}{t} \int \Big\{ \big[1 - q_+(x)\big] \mathcal{T}^t q_+(x) - \mathcal{T}^t \big[ q_+(1 - q_+) \big](x) \Big\} q_-(x) \pi(dx) \\ &= \lim_{t\to 0} \frac{1}{t} \int \Big( \mathcal{T}^t q_+^2(x) - q_+(x) \mathcal{T}^t q_+(x) \Big)\, q_-(x) \pi(dx), \end{aligned} \tag{2.31}$$

where we have made use of the fact that the integral of the Heaviside function (which enters through the indicator functions) is the ramp function. This expression for $R_{AB}$ immediately

suggests the estimator:

$$R_{AB} \approx \frac{1}{t} \sum_i q_+(X_i^{t \wedge T_i^+}) \left[ q_+(X_i^{t \wedge T_i^+}) - q_+(X_i^0) \right] q_-(X_i^0) w(X_i^0) \tag{2.32}$$

for some small choice of $t$. Note the use of stopped trajectories in (2.32). For very small values of $t$ the inclusion of the stopping time $T^+$ has no impact. However, in our numerical experiments we find that use of stopped trajectories improves the accuracy of (2.32) and, in particular, (2.36) below, for most choices of $t$. Given an estimate of $R_{AB}$, the rate constant is

$$k_{AB} = \frac{R_{AB}}{\sum_i q_-(X_i^0) w(X_i^0)}. \tag{2.33}$$

The denominator in 2.33 is the mean of the backward committor, which is the fraction of time the system spends having last visited state A.

We can also use simulations to understand how reactive trajectories flow through a CV space. One way to do this is to partition the space into discrete states and then estimate the reactive fluxes between pairs of states [58]. However, the resulting directed graph can be complicated and difficult to interpret. When the sample paths are continuous the reactive flux between neighboring values in CV space is can be summarized as a single vector field in CV space. If $\theta$ is a vector-valued CV and $ds$ is a bin in CV space of volume $|ds|$, the reactive current at point $s$ is

$$\begin{aligned} J_{AB}^{\theta}(s) = \lim_{t,\, |ds| \to 0} \frac{1}{2t\, |ds|} \int & \left( \mathcal{T}^t[\theta q_+](x) - \theta(x) \mathcal{T}^t q_+(x) \right) \mathbb{1}_{\{\theta \in ds\}}(x) q_-(x) \pi(dx) \\ & + \left( \mathcal{T}^t[\mathbb{1}_{\{\theta \in ds\}}\, \theta q_+](x) - \theta(x) \mathcal{T}^t[\mathbb{1}_{\{\theta \in ds\}}\, q_+](x) \right) q_-(x) \pi(dx) \end{aligned} \tag{2.34}$$

In appendix 2.6.2, we show that $J_{AB}^{\theta}(s) = \int J_{AB} \cdot \nabla \theta(x) \delta(\theta(x) - s) \pi(dx)$, and we established

that the projected reactive current satisfies

$$\int_{\partial C^\theta} J^\theta_{AB}(s) \cdot n_{C^\theta} d\sigma_{C^\theta} = \int_{\partial C} J_{AB} \cdot n_C d\sigma_C, \tag{2.35}$$

where $C^\theta$ is any region of CV space such that its inverse image (under the CV mapping) in the full configuration space, $C$, contains $A$ and does not intersect $B$. To estimate $J^\theta_{AB}$ from trajectory data we have the following estimator:

$$\begin{aligned} J^\theta_{AB}(s) \approx \frac{1}{2t|ds|} \sum_{i=1}^{M} q_+(X_i^{t\wedge T_i^+(0)}) \left(\theta(X_i^{t\wedge T_i^+(0)}) - \theta(X_i^0)\right) \mathbb{1}_{\theta\in ds}(X_i^0) q_-(X_i^0) w(X_i^0) \\ + \frac{1}{2t|ds|} \sum_{i=1}^{M} q_+(X_i^t) \left(\theta(X_i^t) - \theta(X_i^{T^-(t)})\right) \mathbb{1}_{\theta\in ds}(X_i^t) q_-(X_i^{T^-(t)}) w(X_i^0) d \end{aligned} \tag{2.36}$$

Note that the lag time $t$ in (2.32) and (2.36) need not be the same as the lag time $\tau$ used to estimate the committors $q_+$ and $q_-$. Even with perfect sampling and a perfect basis, estimates of TPT quantities will depend on $t$, in contrast to $\tau$. Several considerations are involved in the choice of $t$. For larger values of $t$ (2.32) and (2.36) incur significant bias due to poor approximation of the $t \to 0$ limit in (2.31) and (2.34). On the other hand, for small values of $t$, we found that (2.32) and (2.36) suffer large statistical errors. A full analysis of error sources is beyond the scope of this work, and in practice we choose a lag time that gives reasonable results for the change of measure and reasonable smoothness in the vector field.

## 2.3 Simulation methods and choices

In this section, we specify the computational procedure to generate and analyze the data set for the unfolding and folding of trp-cage. We describe preparing the system and its underlying dynamics (Section 2.3.1), choosing collective variables based on their ability to

distinguish metastable states (Section 2.3.2), generating and validating the data set of short trajectories (2.3.3), and defining the unfolded and folded states (Section 2.3.4).

### *2.3.1 System setup*

Unless otherwise noted, all molecular dynamics simulations were performed with GROMACS 5.1.4 [59] and PLUMED 2.3 [60, 61, 62] using the CHARMM36m force field [63, 64, 65] in the NVT ensemble at 300 K using the Langevin thermostat with a temperature coupling constant of 10 $ps^{-1}$ applied to all atoms, and a time step of 2 fs. Bonds to hydrogen atoms were constrained using the LINCS algorithm [66]. Electrostatic interactions were computed using particle-mesh Ewald summation with a cutoff of 1.2 nm. Lennard-Jones interactions were switched off from 1.0 to 1.2 nm using the default GROMACS switching function.

The system was prepared from an NMR structure of trp-cage (PDB code 1L2Y [67]). The protein was solvated in a 50 Å cubic box with the TIP3P water model [68] using CHARMM-GUI 3.0 [69, 70]. 10 $K^+$ and 11 $Cl^-$ ions were added, bringing the system to charge neutrality and 150 mM KCl. The energy of the system was minimized until the maximum force was below 1000 kJ/mol nm. The system was then equilibrated for 1 ns in the NVT ensemble with position restraints (using a 1 fs timestep), 10 ns in the NPT ensemble with harmonic restraints on non-hydrogen atom positions (force constant 400 kj/mol $nm^2$ for backbone atoms and 40 kj/mol $nm^2$ for side chain atoms.) and a Parrinello-Rahman barostat with a pressure coupling constant of 5 $ps^{-1}$, 5 ns in the NPT ensemble without position restraints, and then 10 ns in the NVT ensemble without position restraints. The cubic box length was determined from the restraint-free NPT equilibration run to be 4.48 nm and fixed at that value after that run.

### *2.3.2 Choice of CVs*

The performance of DGA rests on having a data set with good sampling of all states that contribute to the reaction mechanism. As mentioned in the Introduction, the available physically weighted molecular dynamics data for trp-cage [42] contain few unfolding and folding transitions. We thus sought to use enhanced sampling methods to generate a data set with improved representation outside the stable states. To this end, we evaluated CVs for their ability to control sampling and resolve the unfolded and folded states.

Based on previous studies [40, 44], we considered five CVs:

1. The radius of gyration of the $C_\alpha$ atoms ($R_g$);
2. The root mean squared deviation (RMSD) of all $C_\alpha$ atoms from their positions in an equilibrated structure ($\text{RMSD}_{\text{full}}$);
3. The RMSD of the $C_\alpha$ atoms of residues 2 to 9, which make up the $\alpha$-helix in the native state ($\text{RMSD}_{\text{hx}}$);
4. The RMSD of the $C_\alpha$ atoms of residues 11 to 15, which make up the 3-10 helix in the native state ($\text{RMSD}_{3-10}$);
5. The end-to-end distance ($d$).

$R_g$, $\text{RMSD}_{\text{full}}$, and $\text{RMSD}_{\text{hx}}$ were used in ref. 40, and $\text{RMSD}_{3-10}$ was used in ref. 44 (there defined only to residue 14), where they found that it was able to resolve several metastable states identified by spectral clustering.

To explore how these collective variables change as trp-cage unfolds, we ran a series of Adiabatic Bias Molecular Dynamics (ABMD) [71] simulations to drive unfolding from the equilibrated native structure. ABMD uses a ratchet-and-pawl-like bias to trap spontaneous fluctuations that move the system forward in selected CVs. By applying ABMD with different combinations of the CVs above, we found that $\text{RMSD}_{\text{full}}$ and $\text{RMSD}_{3-10}$ yielded

reasonable control of the system and enabled exploration of all metastable states characterized in previous studies.

### *2.3.3 Generation of the DGA data set*

To initialize a data set of short trajectories for DGA, we defined a grid of 64 points in the space of $\mathrm{RMSD}_{\mathrm{full}}$ and $\mathrm{RMSD}_{3-10}$ (Figure 2.1). We then used 64 independent ABMD simulations to steer the system to each of these points from the final structure from the equilibration simulations described in Section 2.3.1. We ran each ABMD simulation for 1 ns, saving the structure every 5 ps; the force constants were 1.25 kJ/(mol Å$^2$) and 1.0 kJ/(mol Å$^2$) for $\mathrm{RMSD}_{\mathrm{full}}$ and $\mathrm{RMSD}_{3-10}$, respectively. From the set of all recorded structures, we chose the 64 structures closest to the targets and equilibrated each for 1 ns with a harmonic restraint with the same force constants as in the AMBD simulations. From each of the resulting structures, we then launched 14 free simulations (with different random number generator seeds) of length 30 ns each, saving structures every 5 ps.

From this data set, we computed all possible two-dimensional potentials of mean force (PMFs) involving the CVs listed in Section 2.3.2. We compared these PMFs with corresponding ones from replica exchange umbrella sampling (REUS). Based on the DGA PMFs, we used the RMSD of the $\alpha$-helix ($\mathrm{RMSD}_{\mathrm{hx}}$), and the RMSD of the 3-10 helix ($\mathrm{RMSD}_{3-10}$), and the end-to-end distance ($d$) to control the sampling. REUS window centers were placed on a uniform $8 \times 8 \times 8$ grid of these three CVs, with $\mathrm{RMSD}_{\mathrm{hx}}$ ranging from 0.3 to 2.8 Å, $\mathrm{RMSD}_{3-10}$ ranging 0.3 to 3.3 Å, and $d$ ranging from 6 to 38 Å. This grid fully covered the relevant areas of CV space identified by previous simulations. The force constants for the harmonic potentials for each window were 29.2 kJ/(mol $\cdot$ Å$^2$) for $\mathrm{RMSD}_{\mathrm{hx}}$, 20.3 kJ/(mol $\cdot$ Å$^2$) for $\mathrm{RMSD}_{3-10}$, and 0.178 kJ/(mol $\cdot$ Å$^2$) for $d$, following ref. 72. To initialize each window, structures were taken from the DGA database that were closest to each window center. The built-in replica exchange functionality of GROMACS was used to create a

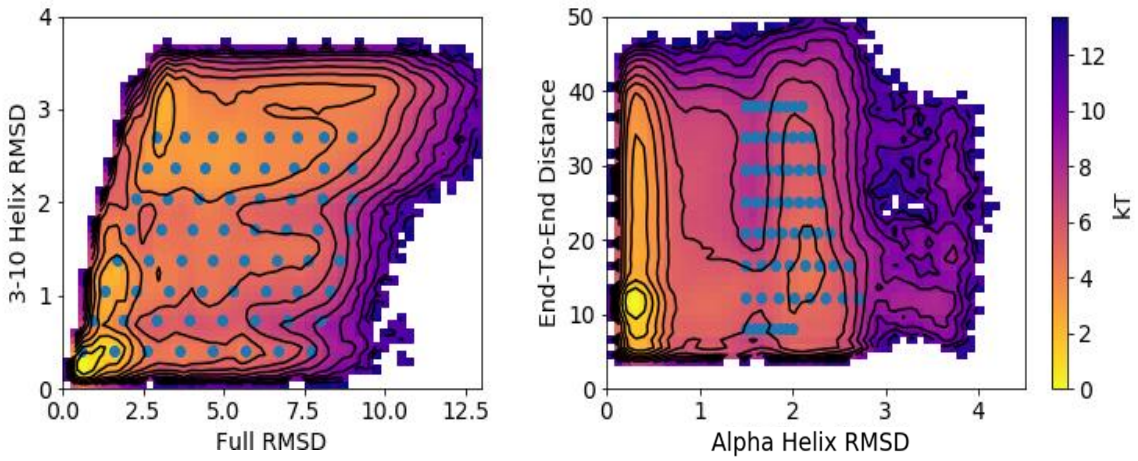


Figure 2.1: Initialization points for the data set of short trajectories. ABMD targets (symbols) are overlaid on DGA PMFs (color scale and contours, spaced every 1 $k_BT$) for the CVs used for steering. (left) The initial 64 ABMD targets were based on $\mathrm{RMSD_{full}}$ and $\mathrm{RMSD_{3\text{-}10}}$; 14 free simulations of length 30 ns were launched from each of the structures resulting from these ABMD simulations. (right) 64 ABMD targets in $\mathrm{RMSD_{hx}}$ and end-to-end distance added to ensure adequate sampling of the unfolded state; 2 free simulations of length 30 ns were launched from each of the structures resulting from these ABMD simulations.

three-dimensional replica exchange procedure, where structures from nearby windows were periodically exchanged [73]. Every window was first simulated for 100 ps, with swaps attempted between adjacent windows in $d$ space (i.e., window centers with the same $\mathrm{RMSD_{hx}}$ and $\mathrm{RMSD_{3-10}}$ values, but neighboring $d$ values) every 10 ps. This was repeated for a total of three 100 ps iterations, with the second and third iterations proposing swaps between neighboring windows in $\mathrm{RMSD_{hx}}$ and $\mathrm{RMSD_{3-10}}$, respectively. This 300-ps procedure was repeated until a total simulation time of 10 ns was reached for each window, with structures saved every 10 ps. Following this protocol, structures were exchanged across all of the three-dimensional grid, with exchange probabilities in the range 10-60%. The PMF was constructed by using the Eigenvector Method for Umbrella Sampling (EMUS) [74] extended

to REUS [75]. The REUS simulations were run until the asymptotic variance of the PMF dropped below 0.1 $(k_B T)^2$ (Figure 2.10).

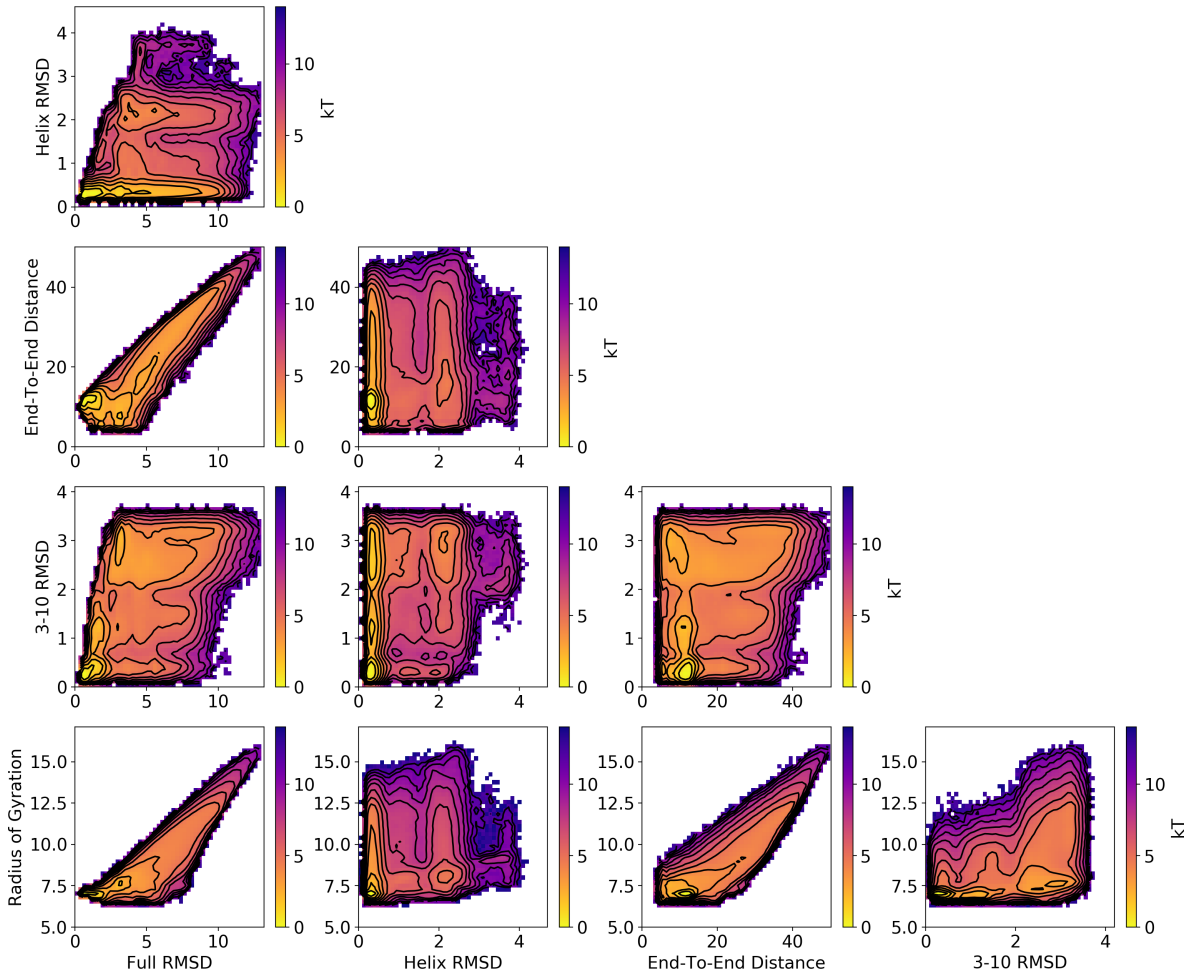


Figure 2.2: PMFs for the indicated CVs. Results shown are computed by DGA with the modified distance basis set and a lag time of 0.5 ns. We use a $50 \times 50$ grid to compute each PMF. Similar results are obtained with other basis sets and REUS; see Figures 2.13, 2.14, and 2.15.

The REUS PMFs suggested that the initial DGA data set did not adequately sample configurations with $\mathrm{RMSD}_{\mathrm{hx}} > 1.5$ (Figure 2.12, note the lack of sampling toward the upper right areas of the plots compared with those in Figure 2.2). In this case several of the basins are missing, and the RMSD over all bins is $> 1.3\, k_B T$. Therefore, we selected 64 more points from a grid with $\mathrm{RMSD}_{\mathrm{hx}} > 1.5$ and a range of end-to-end distances from our short

trajectory data set. From each of these points, we released two new free molecular dynamics simulations of length 30 ns (Figure 2.1B). With these additional trajectories, we obtained good agreement between DGA and REUS PMFs. Adding the extra sampling improved the PMFs involving $\mathrm{RMSD}_{\mathrm{hx}}$ the most, but other PMFs were also noticeably improved. The data set used for all further DGA calculations thus contains a total of 1024 trajectories, each of length 30 ns, with structures saved every 5 ps.

### *2.3.4 State definitions*

We found that PMFs projected onto only global measures of unfolding ($\mathrm{RMSD}_{\mathrm{full}}$, $R_g$, and $d$) did not have clearly identifiable unfolded basins (Figures 2.1 and 2.2). By contrast, the PMF on the CVs tracking secondary structure ($\mathrm{RMSD}_{\mathrm{hx}}$ and $\mathrm{RMSD}_{3-10}$) had clearly identifiable unfolded and folded basins, as well as several intermediates. Based on this analysis, we took the unfolded state to be

$$\frac{|\mathrm{RMSD}_{\mathrm{hx}} - 2.15\ |^3}{0.008\ ^3} + \frac{|\mathrm{RMSD}_{310} - 2.8\ |^3}{0.125\ ^3} < 1. \tag{2.37}$$

The folded state is

$$\frac{(\mathrm{RMSD}_{\mathrm{hx}} - 0.3\ )^2}{0.0289\ ^2} + \frac{(\mathrm{RMSD}_{310} - 0.3\ )^2}{0.04\ ^2} < 1 \text{ and } d < 17\ . \tag{2.38}$$

We included the end-to-end distance constraint on the folded state to exclude structures which are extended but have the secondary structure intact.

Heterogeneous structures contribute to the unfolded state, making it challenging to define, and there is no guarantee that the choices above are optimal in any sense. Because we expect unfolding and folding to be among the slowest motions of the system, an alternative would be to define the states in terms of the slowest mode of the system identified by a dimensionality-reduction algorithm. However, data-driven state definitions are often difficult to interpret

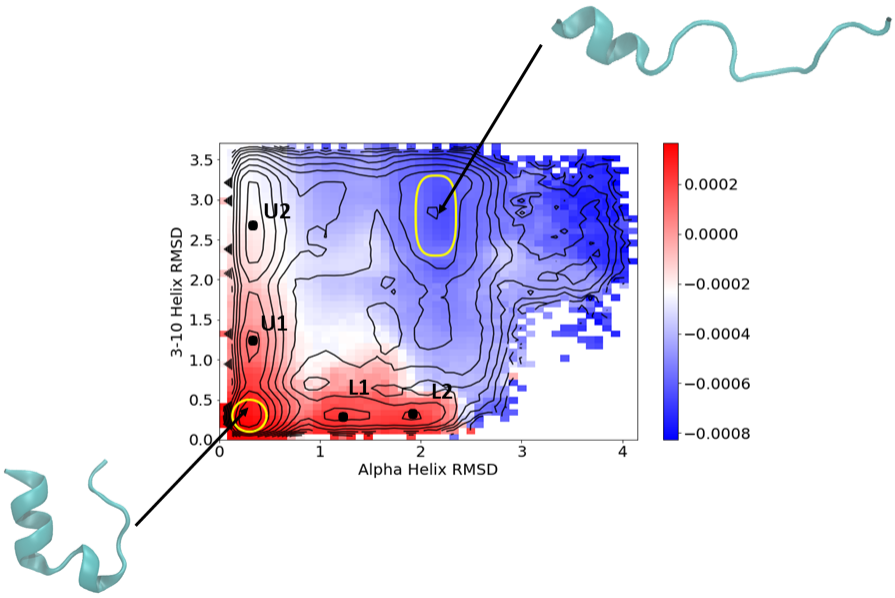


Figure 2.3: Top nontrivial TICA eigenvector averaged on the $\text{RMSD}_{\text{hx}}$ and $\text{RMSD}_{\text{3-10}}$ CVs with physical weighting. The unfolded and folded states are indicated in yellow with representative structures. Intermediate states in Table 2.1 are marked and labeled.

physically, despite their theoretical justifications. Furthermore, data-driven state definitions can be difficult to incorporate into sampling algorithms. We thus use physical CVs for path sampling, stratification, and state definitions, and we then check for consistency with a data-driven state choice.

Figure 2.3 shows that the slowest mode of the system identified by TICA applied to the DGA data set correlates with the PMF and switches between low and high values in going between the unfolded and folded states. Here and going forward, all functions we project onto CVs are conditional averages of the form $\int f(x)\delta(\theta(x) - s)\pi(dx)/\int \delta(\theta(x) - s)\pi(dx)$. We estimate these by binning our CV space into bins, and for each bin $ds$, plotting:

$$\frac{\int f(x)\delta(\theta(x) - s)\pi(dx)}{\int \delta(\theta(x) - s)\pi(dx)} \approx \frac{\sum_i f(X_i^0)w(X_i^0)\,\mathbb{1}_{\theta\in ds}(X_i^0)}{\sum_i w(X_i^0)\,\mathbb{1}_{\theta\in ds}(X_i^0)}. \tag{2.39}$$

We furthermore show in Section 2.4.2 that this mode correlates with the committor. We thus feel that $\mathrm{RMSD}_{\mathrm{hx}}$ and $\mathrm{RMSD}_{3-10}$ enable the clearest two-dimensional projection of the reaction and present most of our results in terms of these CVs. In addition to the unfolded and folded states, we define four intermediate states U1, U2, L1, and L2 shown on Figure 2.3. In the next section, we apply our DGA and TPT formalism to show that trp-cage can fold along an upper path through intermediates U1 and U2, or a lower path through L1 and L2.

## 2.4 Trp-cage analysis

In this section, we evaluate how the three basis sets described in Section 2.2.2 (indicator functions of pairwise distances, indicator functions of TICA coordinates, and pairwise distances modified to satisfy the boundary conditions) impact the performance of DGA for estimating PMFs, rates, committors, and reactive currents for the unfolding and folding of the trp-cage miniprotein. Where possible, we compare our results with references obtained by independent means.

### *2.4.1 Comparison of PMFs*

Figure 2.2 shows PMFs computed on each pair of the physically motivated CVs with DGA with the modified distance basis set. The corresponding PMFs from REUS are shown in Figure 2.11; difference maps comparing the results obtained with the two methods and three basis sets are shown in Figures 2.13, 2.14, and 2.15. All of the main basins identified by REUS are present in the DGA PMFs, and there is good quantitative agreement between REUS and DGA, with RMSDs of $< 1\ k_B T$ for all three basis sets (that said, of these, the distance indicator basis set results in the largest deviations). Consistent with their agreement with the REUS PMF, the three DGA PMFs are in agreement with each other. We did observe that REUS tends to give slightly flatter PMFs than DGA with all three basis

sets. In principle, there are two sources of error in the DGA PMFs: (i) approximation error from representing the true change of measure with a basis expansion and (ii) estimation (sampling) error. Analysis of error in DGA will be the subject of future work. Error in US is discussed in refs. 74, 75, and 76.

Table 2.1: CV values for metastable states.

| State | $\text{RMSD}_{\text{full}}$/Å | $\text{RMSD}_{\text{hx}}$/Å | $d$/Å | $\text{RMSD}_{\text{3-10}}$/Å | $R_g$/ Å |
|---|---|---|---|---|---|
| Folded | 1.1 | 0.30 | 11.1 | 0.30 | 7.0 |
| Unfolded | 5.8 | 2.1 | 20.2 | 2.8 | 9.2 |
| U1 | 2.4 | 0.34 | 13.1 | 1.2 | 7.3 |
| U2 | 5.2 | 0.34 | 19.3 | 2.8 | 8.8 |
| L1 | 2.2 | 1.2 | 9.5 | 0.30 | 7.2 |
| L2 | 2.6 | 1.9 | 14.5 | 0.30 | 7.3 |

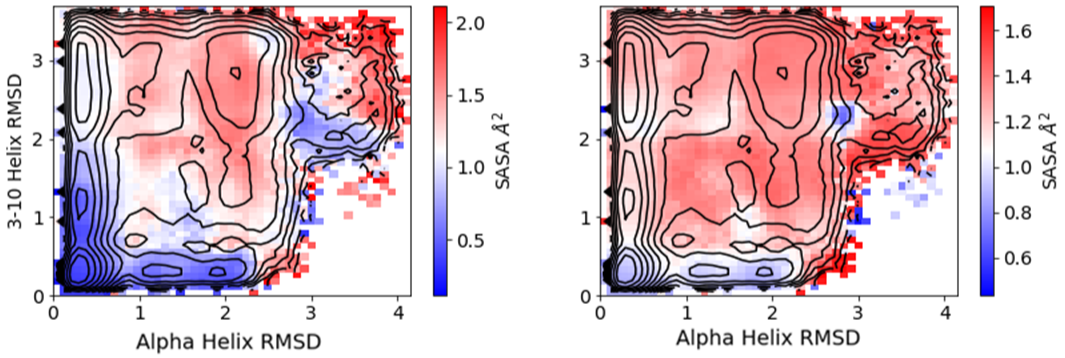


Figure 2.4: Equilibrium average solvent accessible surface area (SASA) projected onto the $\text{RMSD}_{\text{hx}}$ and $\text{RMSD}_{\text{3-10}}$ CVs for (left) trp-6 and (right) proline-12.

We found that the projection onto the $\text{RMSD}_{\text{hx}}$ and $\text{RMSD}_{\text{3-10}}$ coordinates was best able to separate the pathways and states of interest, so we now focus on this projection. Figure 2.3

indicates the folded (lower left) and unfolded (upper right) basins, as well four intermediates. The intermediates define two pathways, which we label upper (with intermediates U1 and U2) and lower (with intermediates L1 and L2). Table 2.1 gives the five CV values for each of the six states.

To understand the characteristics of the intermediate states, we turn to Figure 2.4, which shows the solvent-accessible surface area (SASA) of trp-6 on the left, and pro-12 on the right. We find that the U2 intermediate state is characterized by partial solvation of the hydrophobic core, measured by the SASA of trp-6, and nearly full detachment of pro-12. Furthermore, the U2 state is significantly more extended than the lower pathway intermediates as measured both by $R_g$ and end-to-end distance. In addition to being more compact, with near-native $R_g$ values, L1 and L2 have near-native trp-6 and pro-12 SASA values, suggesting the hydrophobic core is fully formed. These intermediate states can be mapped to those previously reported in the literature. Bolhuis and Jurazek [1] identified three folding intermediates. Our U1 and U2 intermediates roughly map onto their $P_d$ and I intermediates, and our L1 and L2 intermediates roughly map onto their L intermediate. U1 and U2 also correspond to states $S_7$ and $S_0$ identified by Sidky *et al.* [44].

### *2.4.2 Comparison of committors*

We next calculated both forward and backward committors using DGA with the three basis sets and lag times ranging from 0.5 ns to 12 ns (Figure 2.5 and Figure 2.16). As they should, the backward committors mirror the forward committors, so we focus our discussion on the latter. The timescale of trp-cage folding is on the order of 5 $\mu$s from both experiment [39] and simulation[40], thus both our trajectory lengths (30 ns) and lag times are several orders of magnitude shorter than the motions of interest, providing an appropriate setting in which we expect DGA to show benefits.

In contrast to the PMFs, we found the committors to be sensitive to the choice of basis

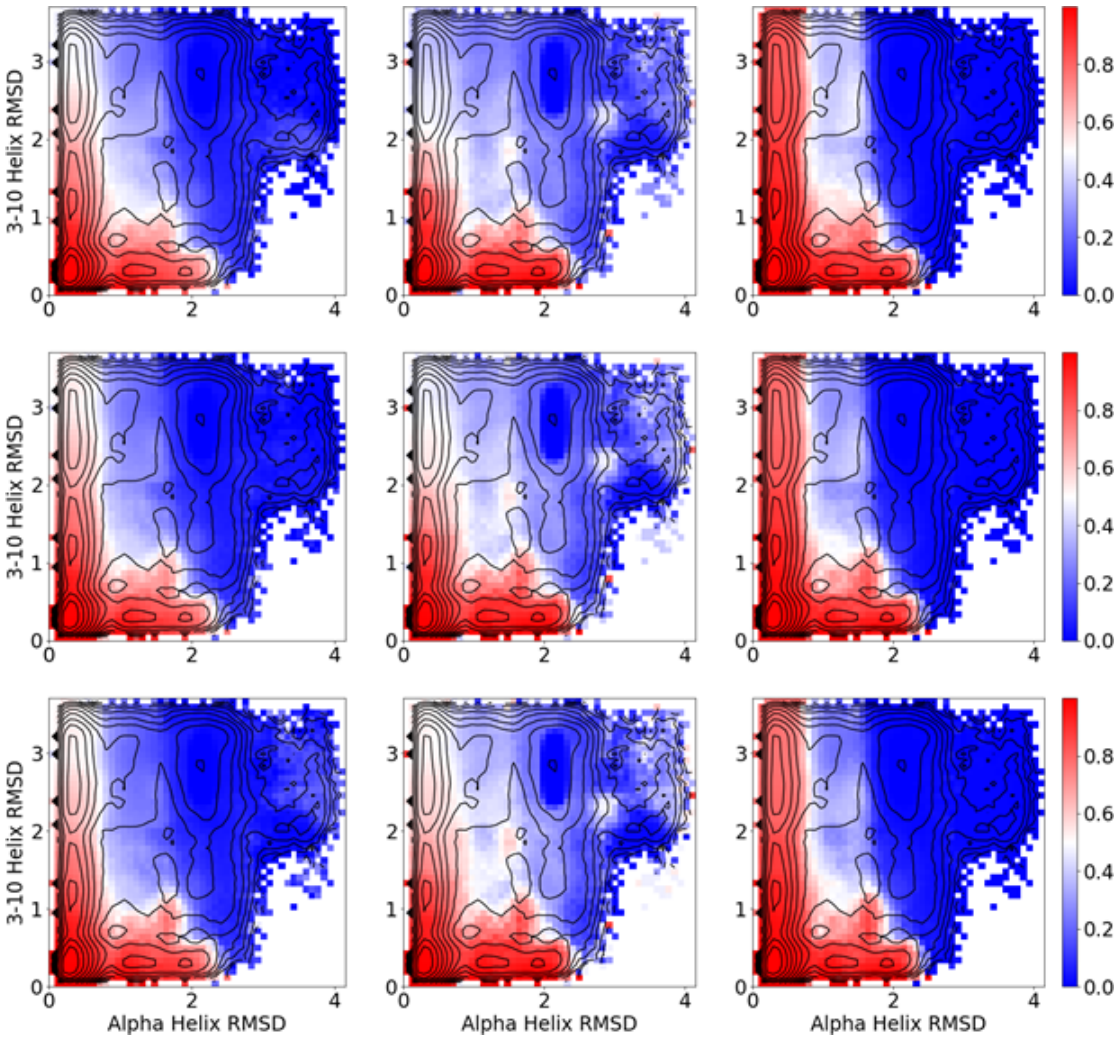


Figure 2.5: DGA forward committors. Left, middle, and right columns are computed with the modified distance, distance indicator, and TICA indicator basis sets, respectively. Top, middle, and bottom rows are computed with lag times of 0.5, 2.5, and 7.5 ns, respectively.

set (and associated guess function). The modified distance basis set, in addition to being substantially faster to construct as it avoids slow and unstable high-dimensional clustering, is less prone to discontinuities at the boundary than the distance indicator function basis set. The TICA indicator function basis set performs similarly to the modified distance basis set and has the advantage over the distance indicator basis set that clustering on the lower-dimensional subspace is significantly faster and more stable.

For a given basis set, we found relatively little variation in the committors across lag times.

This is in contrast to variational approach for conformational dynamics (VAC) algorithm, where the results can strongly depend on the lag time [48] (although this can be mitigated by using multiple lag times[77]). We postpone a full investigation of DGA's error properties, and in particular its dependence on the choice of lag time, to future work.

Because we expect unfolding and folding to be among the slowest motions of the system, we can validate the DGA committors by comparing them with the slowest mode of the system identified by TICA. Comparing Figures 2.3 and 2.5 shows that the largest TICA eigenvector (estimated with a lag time of 0.5 ns) correlates almost perfectly with the estimated committors obtained with the modified distance basis set, when projected onto $\mathrm{RMSD}_{\mathrm{hx}}$ and $\mathrm{RMSD}_{3-10}$. The agreement between these two independent calculations furthermore suggests that the physically motivated CVs capture the behavior detected by the data-driven method. In this projection, we see that the transition states fall where the SASA of trp-6 (Figure 2.4) changes rapidly.

As an additional validation, we used DGA with the modified distance basis set and a lag time of 0.5 ns (Figure 2.6) to compute committors on the CVs used by Juraszek and Bolhuis [1]. When projected onto RMSD and $\mathrm{RMSD}_{\mathrm{hx}}$, the positions of the transition states in Figure 4 of ref. 1 fall in areas estimated to have $q_+ = 0.5$ (white in Figure 2.6). The traditional shooting approach employed in ref. 1 is quite computationally costly and provides information about only a limited number of structures. Our ability to capture the transition states thus makes clear the benefit of DGA. We discuss DGA's ability to provide mechanistic information further in the next section.

### *2.4.3 Reactive currents*

We computed reactive currents for the three basis sets using the estimator in (2.36) and the committors from the the previous section (Figure 2.7). For this calculation, we use the shortest lag time of 0.5 ns for both the committor and reactive current, though in principle

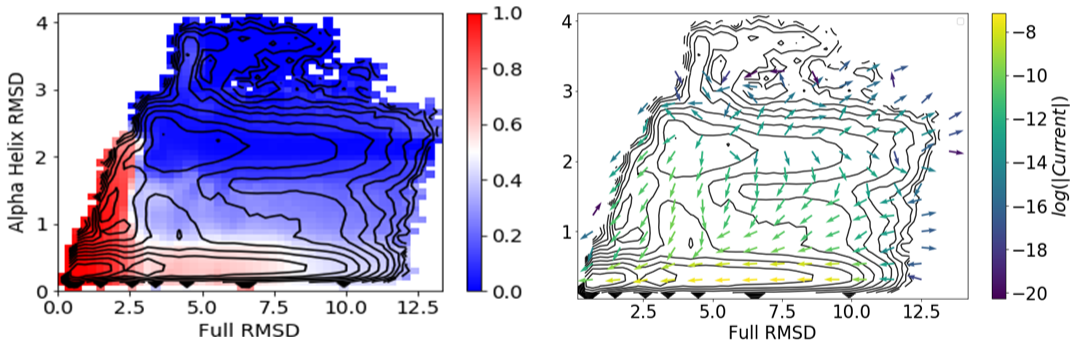


Figure 2.6: Forward committor (left) and reactive current (right) projected onto the $\mathrm{RMSD_{hx}}$ and full RMSD CVs used in ref. 1. Results shown are computed with the modified distance basis set and a lag time of 0.5 ns.

they could be chosen separately. As previously, we primarily present our results projected onto $\mathrm{RMSD_{hx}}$ and $\mathrm{RMSD_{3-10}}$. Overall the results for the three basis sets are similar, though the distance indicator basis set exhibits greater noise around ($\mathrm{RMSD_{hx}}$, $\mathrm{RMSD_{3-10}}$) = (1.3 Å, 1.3 Å), consistent with the plateau in the committor in this region.

The currents, which provide information directly about dynamics, confirm the presence of two paths for the folding process: an upper path with formation of the $\alpha$-helix prior to formation of the 3-10 helix, and a lower path with the order of these events transposed. The upper path proceeds through intermediates U1 and U2, with folding beginning with formation of the $\alpha$-helix and partial desolvation of trp-6, followed by full formation of the 3-10 helix. The lower path proceeds through L1 and L2, with folding beginning with collapse into the L2 intermediate with no $\alpha$-helix, but the hydrophobic core fully formed, followed by formation of the $\alpha$-helix. Both of these paths correspond to troughs in the PMFs on these

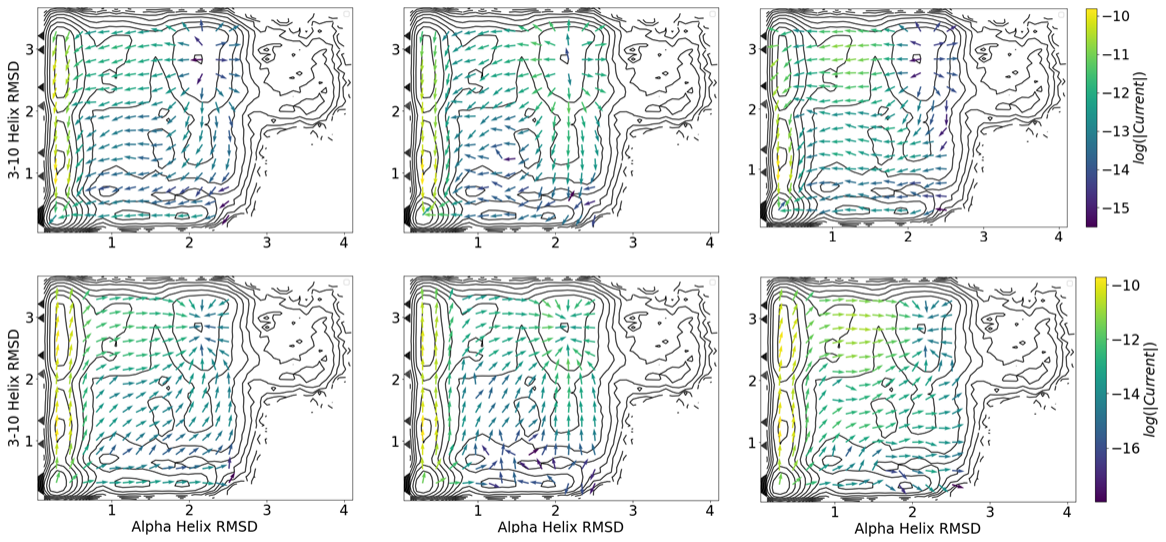


Figure 2.7: Folding (top) and unfolding (bottom) reactive current projected onto the $\mathrm{RMSD}_{\mathrm{hx}}$ and $\mathrm{RMSD}_{\text{3-10}}$ CVs using (2.36) with the three choices of basis set. Left, middle, and right columns are computed with the modified distance, distance indicator, and TICA indicator basis sets, respectively. All computations use a lag time of 0.5 ns.

CVs.

Previous studies have found multiple pathways resembling the ones we find here. Kim et al. [78] used diffusion maps to identify two pathways: one with tertiary contacts forming first, followed by $\alpha$-helix formation, and another with the order transposed. Jurazek and Bolhius came to similar conclusions using transition path sampling [1].

An advantage of the reactive current is that we can use it to assign weights to the two paths. By computing the relative flux crossing $\mathrm{RMSD}_{\text{3-10}} = 1.8$ Å with either $\mathrm{RMSD}_{\mathrm{hx}} < 1.4$ Å (upper pathway) or $\mathrm{RMSD}_{\mathrm{hx}} > 1.4$ Å (lower pathway), we conclude that 88% of the reactive paths proceed by first forming the $\alpha$-helix, and then the 3-10 helix and hydrophobic core (i.e., the upper pathway). Although we are not aware of a previous estimate of the reactive current for this system, we can compare these numbers to the frequencies with

which transition path sampling sampled the pathways in ref. 1. There, Juraszek and Bolhuis observed the pathway in which tertiary contacts form first (i.e., the lower pathway) 80% of the time. The difference may be due to different CV and state definitions (Jurazek and Bolhuis [1] used 5 CVs in their state definitions , whereas we consider only $\mathrm{RMSD}_{3-10}$ and $\mathrm{RMSD}_{\mathrm{hx}}$) or force field and setup differences.

### *2.4.4 Rates*

Finally, we computed rates using the estimator in (2.30). We present our results as inverse rates (unfolding and folding times) to make comparisons to lag times and trajectory lengths clear. As mentioned previously, these times are expected to be on the order of microseconds. In particular, Juraszek and Bolhuis used transition interface sampling to estimate inverse unfolding and folding rates of 1.2 $\mu$s and 0.4 $\mu$s [40], though as noted previously those results are for a different model.

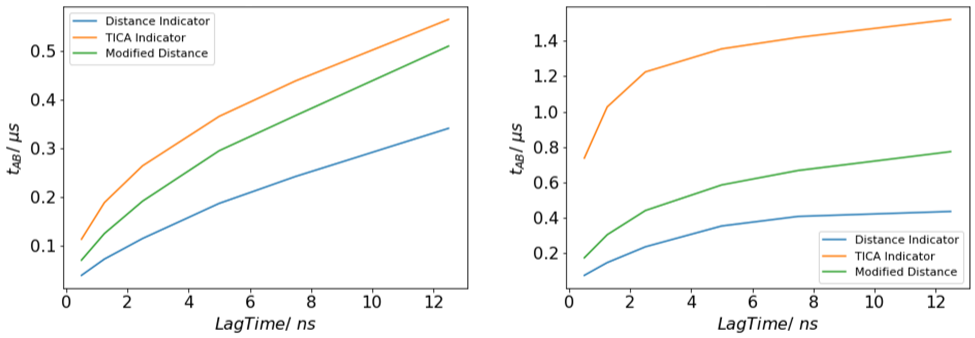


Figure 2.8: Inverse rates estimated for folding (left) and unfolding (right).

All three basis sets gave rate estimates that were within an order of magnitude of those numbers. However, the results for the distance indicator basis were markedly faster. Furthermore, in all three cases, the inverse rate exhibited significant dependence on lag time. We do not show lag times >12 ns since they suffer from pronounced statistical error due to the limitations of our short-trajectory data set. Our analysis of the trajectory of the K8A mutant suggests the need for a lag time of at least 100 ns (consistent with ref. 44), though as discussed in the Introduction, those data do not contain a sufficient number of unfolding and folding events to obtain accurate rate estimates. Juxtaposed with the lack of sensitivity to lag time for the committor and reactive current, these observations suggest that DGA's strength is in its ability to give statistical insight into mechanisms with relatively little data, but that rates may be more efficiently computed by methods that directly sample relevant statistics such as stratification schemes[14].

### *2.4.5 Demonstration of delay embedding*

As described in Section 2.2.2, delay embedding can be used to construct an approximately Markovian process when the feature space does not fully capture the dynamics. To illustrate this idea using our trp-cage data set, we restrict the feature space to the five physical CVs and apply DGA with the modified distance basis set on either the feature space itself or the delay-embedded feature space. Figure 2.9 shows the reactive currents and committors resulting from DGA on these two spaces. We find that the committor and current constructed from the delay embedded representation largely agree with the DGA result constructed on the 153 pairwise distances. Without delay embedding, we find several qualitative disagreements, in particular the U2 state has a committor value close to zero, and the reactive current does not resolve the two pathways since many of the arrows point directly towards the folded state.

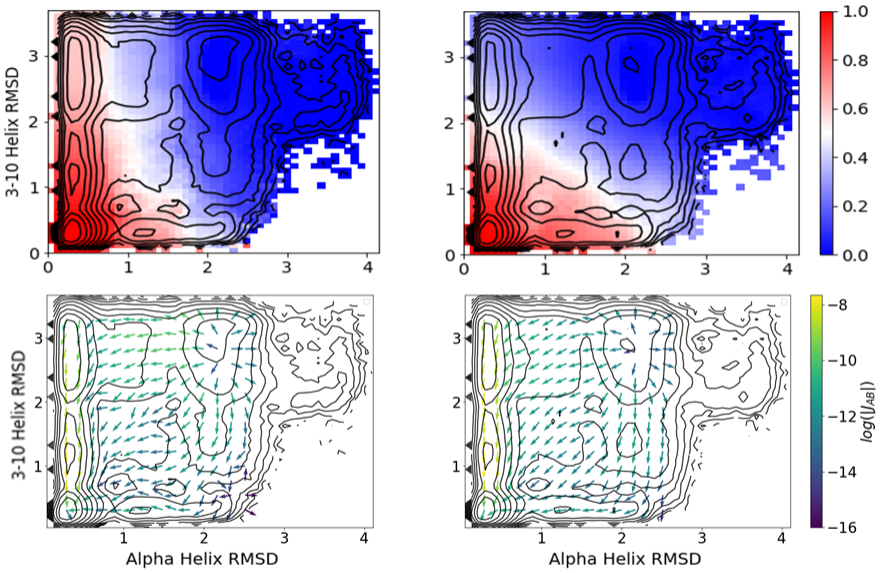


Figure 2.9: Comparison of DGA estimates for the forward committor (top) and reactive current for folding (bottom) with the modified distance basis set on a feature space restricted to the five physical CVs (right) and a delay-embedded feature space (left). The delay-embedded results are obtained with a delay of $\delta = 0.125$ ns, $N = 40$ images, and a DGA lag time of 0.5 ns.

## 2.5 Conclusions

In this paper, we have cast the dynamical Galerkin approximation (DGA) [35] for computing chemical kinetic statistics from short trajectories in terms of the stopped transition operator. This formulation can be immediately translated into expressions that can be applied to simulation data. It also clarifies the role of the lag time, showing that estimates of conditional expectations computed by DGA are exact in the infinite basis and data limit, independent of the choice of lag time.

To evaluate DGA's performance, we generated and carefully validated a data set of short trajectories for the unfolding and folding of the trp-cage miniprotein, a well-characterized system. We used umbrella sampling to validate our short trajectory data set by comparing

the resulting PMFs. Quantitative agreement between the PMFs was observed, suggesting that our short trajectory data set had sufficient sampling to compute dynamical statistics. The PMF calculations furthermore enabled us to rapidly assess different combinations of CVs for their abilities to separate metastable states. The $\alpha$-helix RMSD and 3-10 helix RMSD in particular allowed us to resolve intermediates to a greater degree than found in previous studies.

We next applied DGA to compute forward and backward committors between the unfolded and folded states. We evaluated a number of competing estimators for the backward committor and found that one based on forward trajectories weighted by the stationary distribution gave the best results. The committors by themselves are not able to identify reaction pathways or transition states, but they can be combined according to transition path theory to extract this information. Specifically, we introduce a new estimator for the TPT rate, and a projection formula and corresponding estimator for the reactive current in a CV space. Our projected reactive current allows us to easily resolve and visualize the pathways that the system takes in arbitrary CV spaces, and even lets us assign relative weights to these pathways. Acquiring this kind of mechanistic information has previously been possible only through transition path sampling and related methods; such methods do not as readily allow exploration of CVs and state definitions because the sampling is linked directly to them.

We introduced a simple procedure that takes an arbitrary set of molecular features and adapts them to produce a basis set that satisfies the homogeneous boundary conditions. Using pairwise distances as the molecular features, we compared the performance of such a basis set with indicator functions on the molecular features and indicator functions on TICA coordinates. Other basis constructions such as diffusion maps and radial basis functions are possible, and we expect that the best choice will be system dependent. We applied our DGA and TPT formalism to our data set, and identified intermediate states and pathways which

have been previously reported in the literature, providing further validation of our methods. We found that the estimates of the TPT rate, while on the same order of magnitude as previous estimates, nevertheless show significant dependence on lag time. Finally, we showed that delay embedding can be an effective strategy for constructing a molecular representation with approximately Markovian dynamics from a low-dimensional feature space.

Our results suggest several interesting directions for future investigation. We have seen that in our trp-cage application the choice of lag time has only a modest effect on DGA estimates of conditional expectations, while TPT quantities, in particular the rate, depend sensitively on lag time. Recently, we showed that integrating over lag times for VAC improves the robustness of that method [77]. It will be interesting to see if an analogous strategy can improve rate estimates from DGA. An in depth mathematical study of DGA's error and its dependence on lag time along the lines of our previous analysis of VAC [48] is also in order. By showing how DGA's results depend on the sampling measure, such an analysis could lead to a practical scheme for targeting sampling to selected regions, just as our analysis of US [8, 79] did [75]. This will be particularly important for systems that are not amenable to the strategy that we took in the present study of using REUS for identifying regions of CV space that require more sampling.

Though DGA has performed well in our tests so far, looking ahead to larger and more complex systems, it may become necessary to move away from a Galerkin approach and toward more flexible representations of the kinetic functions we seek to approximate. This would be consistent with a trend toward using neural networks to represent eigenfunctions in spectral estimation [30, 44, 77]. Indeed, some of the first estimates of committors from data used neural networks [80, 81]. Introducing this higher level of representational flexibility while maintaining the reliability we observe in our trp-cage application of DGA will be a challenge.

## 2.6 Appendix

### *2.6.1 Backward-in-time inner products*

In this appendix we provide an elementary derivation of (2.24) which is key to our estimates of inner products involving the stopped backward-in-time transition operator $\mathcal{S}^t$. For the purposes of this derivation we assume that $X^t$ is a discrete time process (so that $t$ is a non-negative integer) with probability density $p(x^1|x^0)$ for transition from $x^0$ to $x^1$ and stationary density $\pi$. The steady state backward-in-time process $X^{-t}$ then has transition density

$$q(x^0|x^1) = \frac{p(x^1|x^0)\pi(x^0)}{\pi(x^1)}. \tag{2.40}$$

From this expression we immediately find that

$$\begin{aligned}\pi(x^t)q(x^{t-1}|x^t)q(x^{t-2}|x^{t-1})\cdots q(x^0|x^1)& \\ = \pi(x^0)p(x^t|x^{t-1})p(x^{t-1}|x^{t-2})\cdots p(x^1|x^0)&\end{aligned} \tag{2.41}$$

relating the steady state backward-in-time path density to the steady state forward-in-time path density. Therefore, for any path function $F(x^0, x^1, \ldots, x^t)$ and any density $\mu$ (equivalent to $\pi$) we find (recalling that here $w = \pi/\mu$) that

$$\begin{aligned}&\int \mathbb{E}\left[F(X^0, X^{-1}, \ldots, X^{-t})\,|\,X^0 = x\right]\mu(x)dx \\ &\quad= \int \frac{F(x^0, \ldots, x^{-t})}{w(x^0)}\pi(x^0)q(x^{-1}|x^0)\cdots q(x^{-t}|x^{-t+1})dx^0\cdots dx^{-t} \\ &\quad= \int \frac{F(x^t, \ldots, x^0)}{w(x^t)}\pi(x^t)q(x^{t-1}|x^t)\cdots q(x^0|x^1)dx^t\cdots dx^0 \\ &\quad= \int \mathbb{E}\left[\frac{F(X^t, X^{t-1}, \ldots, X^0)}{w(X^t)}\,\middle|\,X^0 = x\right]w(x)\mu(x)dx.\end{aligned} \tag{2.42}$$

We will use (2.42) to find an expression for

$$\langle g, \mathcal{S}^{-t} f \rangle = \int \mathbb{E}\left[ g(x) f(X^{-(T^-(0)\wedge t)}) \,|\, X^0 = x \right] \mu(x) dx \tag{2.43}$$

in terms of the forward-in-time process. If we choose

$$F(X^0, X^{-1}, \ldots, X^{-t}) = g(X^0) f(X^{-(T^-(0)\wedge t)}), \tag{2.44}$$

then

$$\langle g, \mathcal{S}^{-t} f \rangle = \int \mathbb{E}\left[ F(X^0, X^{-1}, \ldots, X^{-t}) \,|\, X^0 = x \right] \mu(x) dx. \tag{2.45}$$

In terms of the forward process

$$F(X^t, X^{t-1}, \ldots, X^0) = f(X^{T^-(t)}) g(X^t), \tag{2.46}$$

where we remind the reader that:

$$T^-(t) = \max\{s \leq t : \, X^s \in A \cup B\}$$

with $T^-(t) = 0$ if $X^s \notin A \cup B$ for all $0 \leq s \leq t$.

Applying (2.42) with this choice of $F$ yields (2.24):

$$\left\langle g, \mathcal{S}^{-t} f \right\rangle = \int \mathbb{E}\left[ f(X^{T^-(t)}) \frac{g(X^t)}{w(X^t)} \,\middle|\, X^0 = x \right] w(x) \mu(dx).$$

### *2.6.2 A formula for the reactive current*

It has been shown[57] that for a diffusion with generator

$$\mathcal{L}f(x) = \sum_j b_j(x)\frac{\partial f}{\partial x_j}(x) + \frac{1}{2}\sum_{ij} a_{ij}(x)\frac{\partial^2 f}{\partial x_i \partial x_j}(x) \tag{2.47}$$

the reactive current is the vector field given by

$$\begin{aligned}(J_{AB})_i &= q_+(x)q_-(x)J_i + \\ &\quad \pi(x)q_-(x)\sum_j a_{ij}(x)\frac{\partial q_+}{\partial x_j}(x) - \pi(x)q_+(x)\sum_j a_{ij}(x)\frac{\partial q_-}{\partial x_j}(x),\end{aligned} \tag{2.48}$$

where $J$ is the equilibrium current:

$$J_i = \pi(x)b_i(x) - \sum_j \frac{\partial[\pi a_{ij}]}{\partial x_j}(x). \tag{2.49}$$

To project the current onto a CV space of interest, we take the dot product with $\nabla\theta$ for any smooth CV $\theta$ and, using the identity

$$J_i \cdot \nabla f(x) = \frac{\pi(x)}{2}\left(\mathcal{L}f(x) - \mathcal{L}_\pi^\dagger f(x)\right), \tag{2.50}$$

which follows from direct manipulations, we can write

$$J_{AB} \cdot \nabla\theta(x) = \frac{\pi(x)}{2}\left(q_-(x)\mathcal{L}[q_+\theta](x) - q_+(x)\mathcal{L}_\pi^\dagger[q_-\theta](x)\right) \tag{2.51}$$

for $x \in (A \cup B)^{\mathrm{c}}$. This formula is not useful computationally since it still contains a backward-in-time generator. To compute statistics from data, we need to formulate their estimators as expectations against the stationary distribution since this (1) permits the use of the adjoint relation to clear away backward transition operators and (2) is consistent with our

reweighting scheme. To this end, we define the projected reactive current as

$$J^{\theta}_{AB}(s) = \int J_{AB}(x)\cdot\nabla\theta(x)\delta(\theta(x)-s)dx = \lim_{|ds|\to 0}\frac{1}{|ds|}\int_{\{\theta(x)\in ds\}} J_{AB}(x)\cdot\nabla\theta(x)dx, \quad (2.52)$$

where $ds \in (A\cup B)^{\mathrm{c}}$ is an infinitesimal region of CV space with $s\in ds$, and $\{x:\theta(x)\in ds\}$ does not intersect $A\cup B$. Using (2.51) and the fact that $\mathcal{L}q_+ = 0$ and $\mathcal{L}^{\dagger}_{\pi}q_- = 0$ on $(A\cup B)^{\mathrm{c}}$, we have

$$\begin{aligned} J^{\theta}_{AB}(s) &= \lim_{|ds|\to 0}\frac{1}{|ds|}\int \mathbb{1}_{\{\theta(x)\in ds\}}\frac{\pi(x)}{2}\Big(q_-(x)\mathcal{L}[q_+\theta](x) - q_+(x)\mathcal{L}^{\dagger}_{\pi}[q_-\theta](x)\Big)\,dx \\ &= \lim_{|ds|\to 0}\frac{1}{|ds|}\int \mathbb{1}_{\{\theta(x)\in ds\}}\frac{\pi(x)}{2}\Big(q_-(x)\mathcal{L}[q_+\theta](x) - q_-(x)\mathcal{L}q_+(x)\theta(x) \\ &\qquad\qquad - q_+(x)\mathcal{L}^{\dagger}_{\pi}[q_-\theta](x) + q_+(x)\mathcal{L}^{\dagger}_{\pi}q_-(x)\theta(x)\Big)dx. \end{aligned} \quad (2.53)$$

Writing this expression in terms of the transition operator and canceling terms, we find that

$$\begin{aligned} J^{\theta}_{AB}(s) &= \lim_{t,|ds|\to 0}\frac{1}{2t\,|ds|}\int \mathbb{1}_{\{\theta(x)\in ds\}}\,\pi(x)\Big(q_-(x)\mathcal{T}^t[q_+\theta](x) - q_-(x)\mathcal{T}^t q_+(x)\theta(x) \\ &\qquad - q_+(x)(\mathcal{T}^t)^{\dagger}_{\pi}[q_-\theta](x) + q_+(x)(\mathcal{T}^t)^{\dagger}_{\pi}q_-(x)\theta(x)\Big)dx \\ &= \lim_{t,|ds|\to 0}\frac{1}{2t\,|ds|}\int \pi(x)q_-(x)\Big(\,\mathbb{1}_{\{\theta(x)\in ds\}}\,\Big(\mathcal{T}^t[q_+\theta](x) - \mathcal{T}^t q_+(x)\theta(x)\Big) \\ &\qquad + \Big(\mathcal{T}^t[q_+\theta\;\mathbb{1}_{\{\theta\in ds\}}](x) - \mathcal{T}^t[q_+\,\mathbb{1}_{\{\theta\in ds\}}](x)\theta(x)\Big)\,\Big)dx, \end{aligned} \quad (2.54)$$

where the second equality follows from the definition of the adjoint $(\mathcal{T}^t)^{\dagger}_{\pi}$.

Expression (2.54) for $J^{\theta}_{AB}(s)$ can be directly translated into an estimator for computing

from short-trajectory data:

$$J^{\theta}_{AB}(s) \approx \frac{1}{2t|ds|}\sum_{i=1}^{M} q_+(X_i^t)\left(\theta(X_i^t)-\theta(X_i^0)\right) \times q_-(X_i^0)\,\mathbb{1}_{\theta\in ds}(X_i^0)w(X_i^0) + \frac{1}{2t|ds|}\sum_{i=1}^{M} q_+(X_i^t)\left(\theta(X_i^t)-\theta(X_i^0)\right) \times q_-(X_i^0)\,\mathbb{1}_{\theta\in ds}(X_i^0)w(X_i^0). \tag{2.55}$$

Finally, without affecting the $t \to 0$ limit, we can stop our trajectories when they exit or enter $A \cup B$, yielding the estimator

$$J^{\theta}_{AB}(s) \approx \frac{1}{2t|ds|}\sum_{i=1}^{M} q_+(X_i^{t\wedge T^+(0)})\left(\theta(X_i^{t\wedge T^+(0)})-\theta(X_i^0)\right) \times q_-(X_i^0)\,\mathbb{1}_{\theta\in ds}(X_i^0)w(X_i^0) + \frac{1}{2t|ds|}\sum_{i=1}^{M} q_+(X_i^t)\left(\theta(X_i^t)-\theta(X_i^{T^-(0)})\right) \times q_-(X_i^{T^-(0)})\,\mathbb{1}_{\theta\in ds}(X_i^t)w(X_i^0) \tag{2.56}$$

which, in our experience, outperformed (2.55) for larger values of $t$. Note that we could have canceled additional terms in (2.54) to yield a more concise estimator. However, we found that the estimator (2.56) gave less noisy results.

### *2.6.3 Reactive current on a CV space*

We now establish that our projected reactive current gives the flux over surfaces in CV space. We assume that our CVs are smooth and that, for some subset $C^{\theta}$ of CV space with smooth boundary, the set $C = \{x : \theta(x) \in C^{\theta}\}$ contains $A$ and does not intersect $B$. We

will establish that for such a subset,

$$\int_{\partial C^\theta} J^\theta_{AB}(s) \cdot n_{C^\theta} d\sigma_{C^\theta} = \int_{\partial C} J_{AB} \cdot n_C d\sigma_C. \tag{2.57}$$

Here $n_{C^\theta}$ is the outward pointing normal vector to the boundary $\partial C^\theta$ of $C^\theta$, $n_C$ is the normal vector to the boundary $\partial C$ of $C$, $\sigma_{C^\theta}$ is the surface measure on $\partial C^\theta$ and, $\sigma_C$ is the surface measure on $\partial C$. The significance of (2.57) is that it shows that our definition of $J^\theta_{AB}$ preserves reactive flux across surfaces in the CV space so that statistics of reactive paths could, in principle, be computed directly from $J^\theta_{AB}$.

Let $f_\delta$ be a smooth function on CV space that is equal to 1 on $C^\theta$ and equal to 0 for $x$ a distance of more than $\delta$ from $C^\theta$. Applying the divergence theorem and integrating by parts we find that

$$\begin{aligned}
\int_{\partial C^\theta} J^\theta_{AB}(s) \cdot n_{C^\theta} d\sigma_{C^\theta} &= \int_{C^\theta} \mathrm{div} J^\theta_{AB}(s) ds \\
&= \lim_{\delta \to 0} \int f_\delta(s) \, \mathrm{div} J^\theta_{AB}(s) ds \\
&= -\lim_{\delta \to 0} \int J^\theta_{AB}(s) \cdot \nabla f_\delta(s) ds.
\end{aligned} \tag{2.58}$$

Inserting our definition of $J^\theta_{AB}$ we find that

$$\begin{aligned}
\int_{\partial C^\theta} J^\theta_{AB}(s) \cdot n_{C^\theta} d\sigma_{C^\theta} &= -\lim_{\delta \to 0} \sum_j \int \int J_{AB}(x) \cdot \nabla \theta_j(x) \delta(\theta(x) - s) \frac{\partial f_\delta(s)}{\partial s_j} dx ds \\
&= -\lim_{\delta \to 0} \sum_j \int J_{AB}(x) \cdot \nabla \theta_j(x) \frac{\partial f_\delta}{\partial s_j}(\theta(x)) dx.
\end{aligned} \tag{2.59}$$

Using the chain rule the last expression can be rewritten as

$$\int_{\partial C^\theta} J^\theta_{AB}(s) \cdot n_{C^\theta} d\sigma_{C^\theta} = -\lim_{\delta \to 0} \int J_{AB}(x) \cdot \nabla f_\delta(\theta(x)) dx. \tag{2.60}$$

Integrating by parts, taking the $\delta \to 0$ limit, and applying the divergence theorem again yields (2.57).

### *2.6.4 Linear Perturbation Theory*

It has been shown that the largest source of error in most Markov state modelling procedures is sampling error [82]. Since we expect DGA to behave similarly, we therefore would like to develop tools to assess the magnitude of sampling error and to guide the allocation of computational resources so as to reduce error as efficiently as possible. In this section and the next, we state an asymptotic error formula for the DGA method.

Suppose we have a linear system $Ax = b$, and an approximate system $\hat{A}\hat{x} = \hat{b}$. We want to estimate the error vector $\hat{x} - x$. Our basic tool is the first-order perturbation estimate to the difference between the original system and the estimated system:

$$\hat{x} - x = A^{-1}(\hat{b} - \hat{A}x) \tag{2.61}$$

For an in depth explanation, see Ref. 83, section 3. The estimation error in the DGA forecast function is then:

$$\begin{aligned}
\mathbb{E}\left[\int (u(x) - \hat{u}(x))^2 \mu(dx)\right] &= \mathbb{E}\left[\int \left(\sum_j \phi_j(x)(\hat{v}_j - v_j)\right)^2 \mu(dx)\right] && (2.62)\\
&= \mathbb{E}\left[\int \left(\sum_{jk} \phi_j(x)(A^{-1}(\hat{b} - \hat{A}v))_j\right)^2 \mu(dx)\right] && (2.63)\\
&= \int \sum_{jk} \phi_j(x)\phi_k(x)\sigma_{jk}\mu(dx) && (2.64)\\
&= \sum_{jk} C^0_{jk}\sigma_{jk}, && (2.65)
\end{aligned}$$

where

$$\sigma_{jk} = \mathbb{E}[(A^{-1}(\hat{b} - \hat{A}v))_j (A^{-1}(\hat{b} - \hat{A}v)_k)] = \mathrm{Cov}[A^{-1}(\hat{b} - \hat{A}v)]. \tag{2.66}$$

To estimate the covariance matrix $\Sigma$ from data, we use the DGA estimates of $A^{-1}$ and $v$ in 2.66, Substituting the DGA definitions of the matrix $\hat{A}$ and vector $\hat{b}$, we have

$$\hat{b} - \hat{A}v = \frac{1}{N} \sum_j \Phi(X_j^0) \left( u(X_j^0) - u(X_j^{\tau \wedge T^+}) + \Delta \sum_{t=0}^{(\tau \wedge T^+)-1} \Gamma(X_j^t) \right) \tag{2.67}$$

Define the residual

$$R_j = u(X_j^0) - u(X_j^{\tau \wedge T^+}) + \Delta \sum_{t=0}^{(\tau \wedge T^+)-1} \Gamma(X_j^t) \tag{2.68}$$

Therefore, we have the estimator:

$$\begin{aligned} \hat{\sigma}_{jk} = \frac{1}{N^2} \sum_i [\hat{A}^{-1}\Phi(X_i^0)]_j [\hat{A}^{-1}\Phi(X_i^0)]_k R_i^2 \\ - \left( \frac{1}{N} \sum_i [\hat{A}^{-1}\Phi(X_i^0)]_j R_i \right) \left( \frac{1}{N} \sum_i [\hat{A}^{-1}\Phi(X_i^0)]_k R_i \right). \end{aligned} \tag{2.69}$$

Global estimation error estimates can then be obtained from (2.65).

### *2.6.5 Optimal Sampling Distributions*

To compute the optimal sampling distribution, we first write down an ansatz for the sampling distribution:

$$\mu(dx) = \sum_\ell \frac{w_\ell \mathbb{1}_{S_\ell}(x) \mu_\ell(dx)}{\int \mathbb{1}_{S_\ell}(x) \mu_\ell(dx)} \tag{2.70}$$

with $\sum_\ell w_\ell = 1$. The sets $S_\ell$ are disjoint sets such that $\bigcup_\ell S_\ell = D \cup A \cup B$. In practice, we often choose $S_\ell$ to be a set of bins on a low-dimensional CV space which are capable

of resolving the highest free-energy barriers. The distributions $\mu_\ell$ are arbitrary but fixed sampling distributions on each of the sets $S_\ell$. We then write, for example, the matrix element estimator as

$$C^\tau_{jk} = \sum_\ell \frac{w_\ell}{N_\ell} \sum_i \phi_j(X^0_i)\phi_k(X^{\tau\wedge T^+}_i)\mathbb{1}_{S_\ell}(X^0_i), \tag{2.71}$$

where $N_\ell = \sum_i \mathbb{1}_{S_\ell}(X^0_i)$ is the number of samples in bin $S_\ell$. With this estimator of the inner products, our sampling measure does not depend on the number of samples in each bin in expectation, and so our goal will be to minimize the estimation error over all choices of the $N_\ell$ subject to the constraint $\sum_l N_\ell = N$. We do this by substituting the sampling distribution ansatz (2.70) into the covariance matrix estimator (2.66) to obtain:

$$\begin{aligned}\sigma_{jk} = \sum_\ell \frac{w_\ell^2}{N_\ell} \Bigg\{ &\frac{1}{N_\ell}\sum_i [\hat{A}^{-1}\Phi(X^0_i)]_j [\hat{A}^{-1}\Phi(X^0_i)]_k R_i^2 \mathbb{1}_{S_\ell}(X^0_i) \\ &- \left(\frac{1}{N_\ell}\sum_i [\hat{A}^{-1}\Phi(X^0_i)]_j R_i \mathbb{1}_{S_\ell}(X^0_i)\right)\left(\sum_i [\hat{A}^{-1}\Phi(X^0_i)]_k R_i\right)\mathbb{1}_{S_\ell}(X^0_i)\Bigg\}\end{aligned} \tag{2.72}$$

Defining the factor in curly brackets to be $V_{jk\ell}$, we see that the global error is then given by

$$\text{error} = \sum_\ell \frac{w_\ell^2}{N_\ell} \sum_{jk} C^0_{jk} V_{jk\ell} \tag{2.73}$$

Since asymptotically $V_{jk\ell}$ does not depend on $N_\ell$, the minimizer of the global estimation error over $N_\ell$ subject to the constraint $\sum_\ell N_\ell = N$ is then:

$$N^*_\ell = N \frac{\sum_{jk} C^0_{jk} V_{jk\ell}}{\sum_{jk\ell} C^0_{jk} V_{jk\ell}} \tag{2.74}$$

### *2.6.6 Supplemental figures*

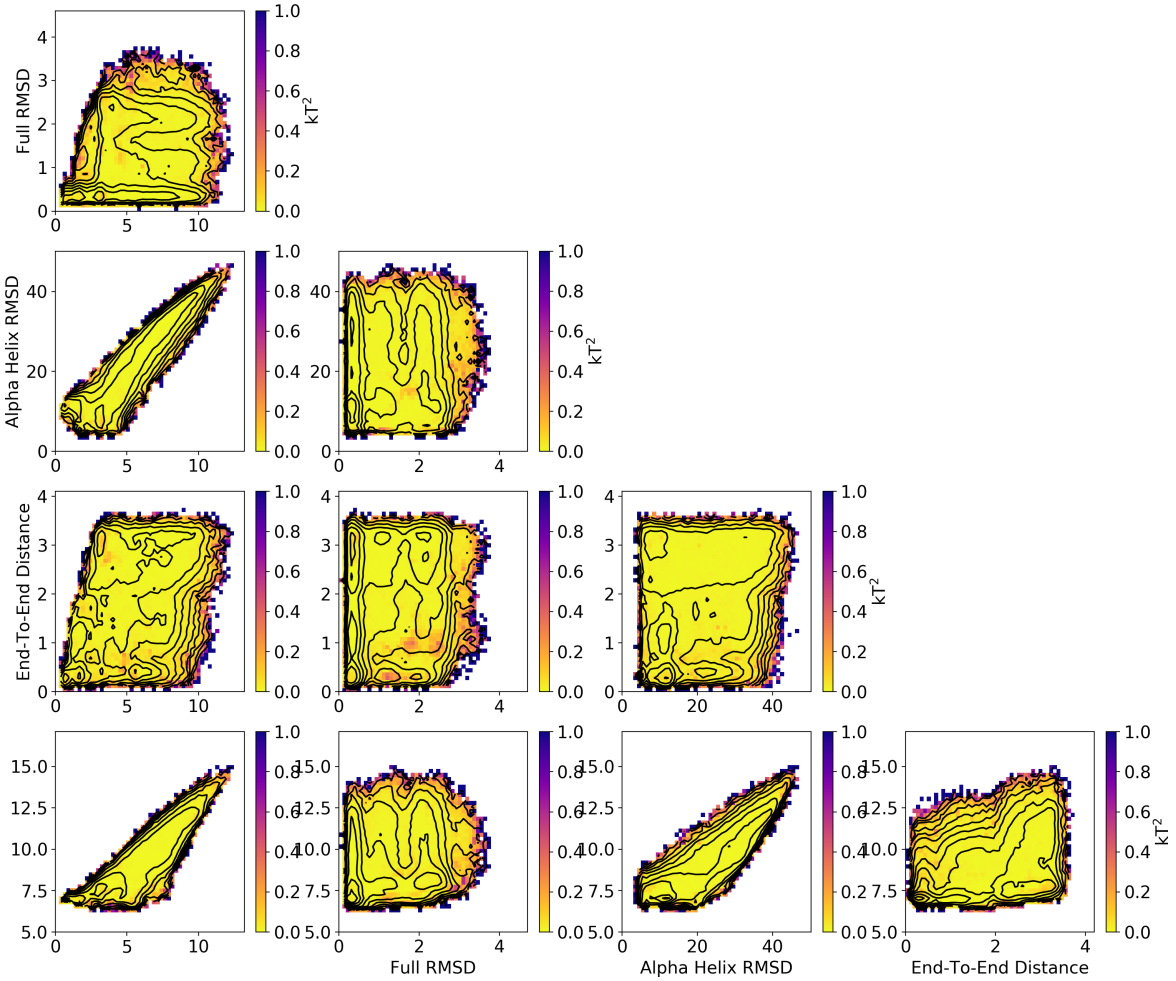


Figure 2.10: EMUS asymptotic variance for REUS PMFs.

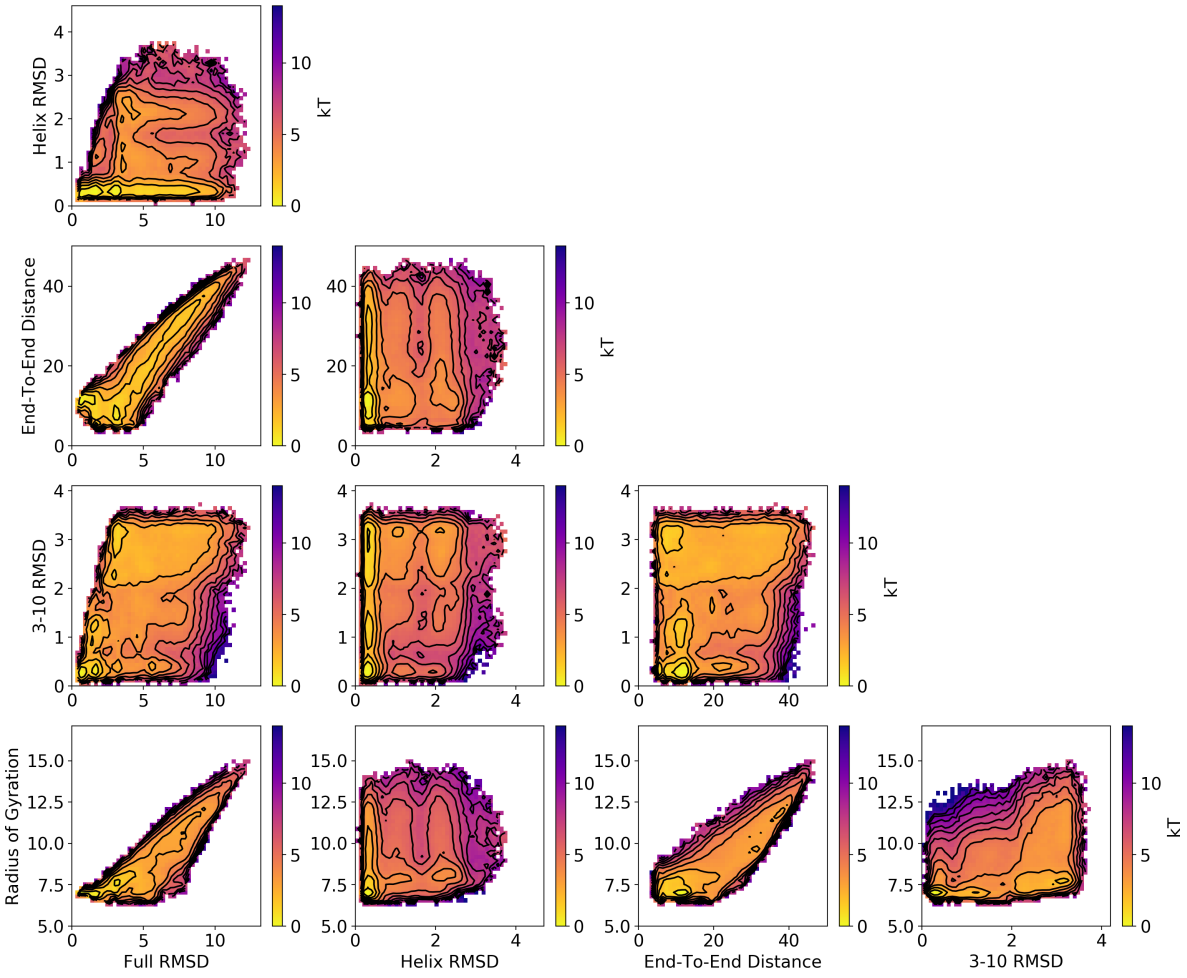


Figure 2.11: REUS PMFs.

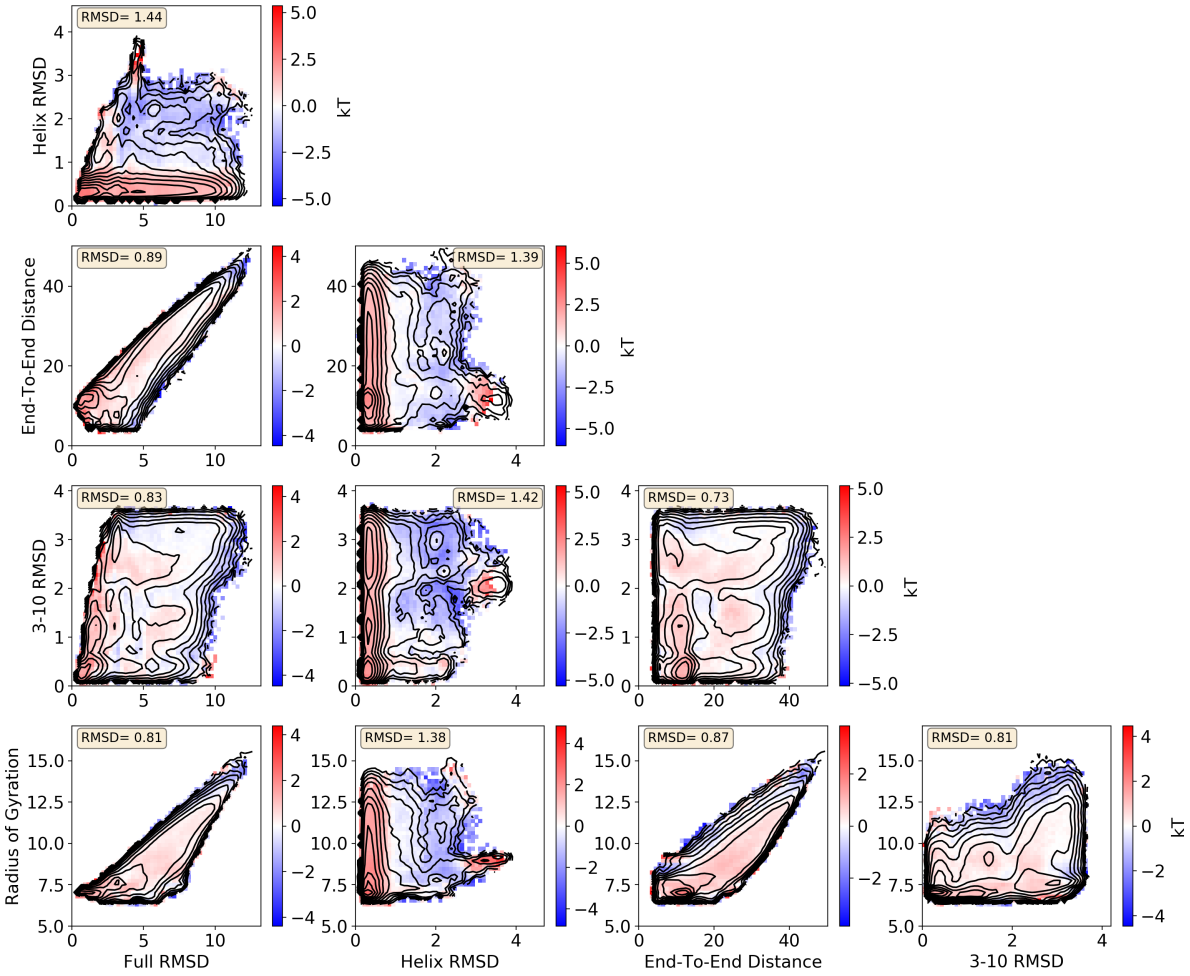


Figure 2.12: Difference between DGA with the modified distance basis set without the $\alpha$-helix resampling and REUS.

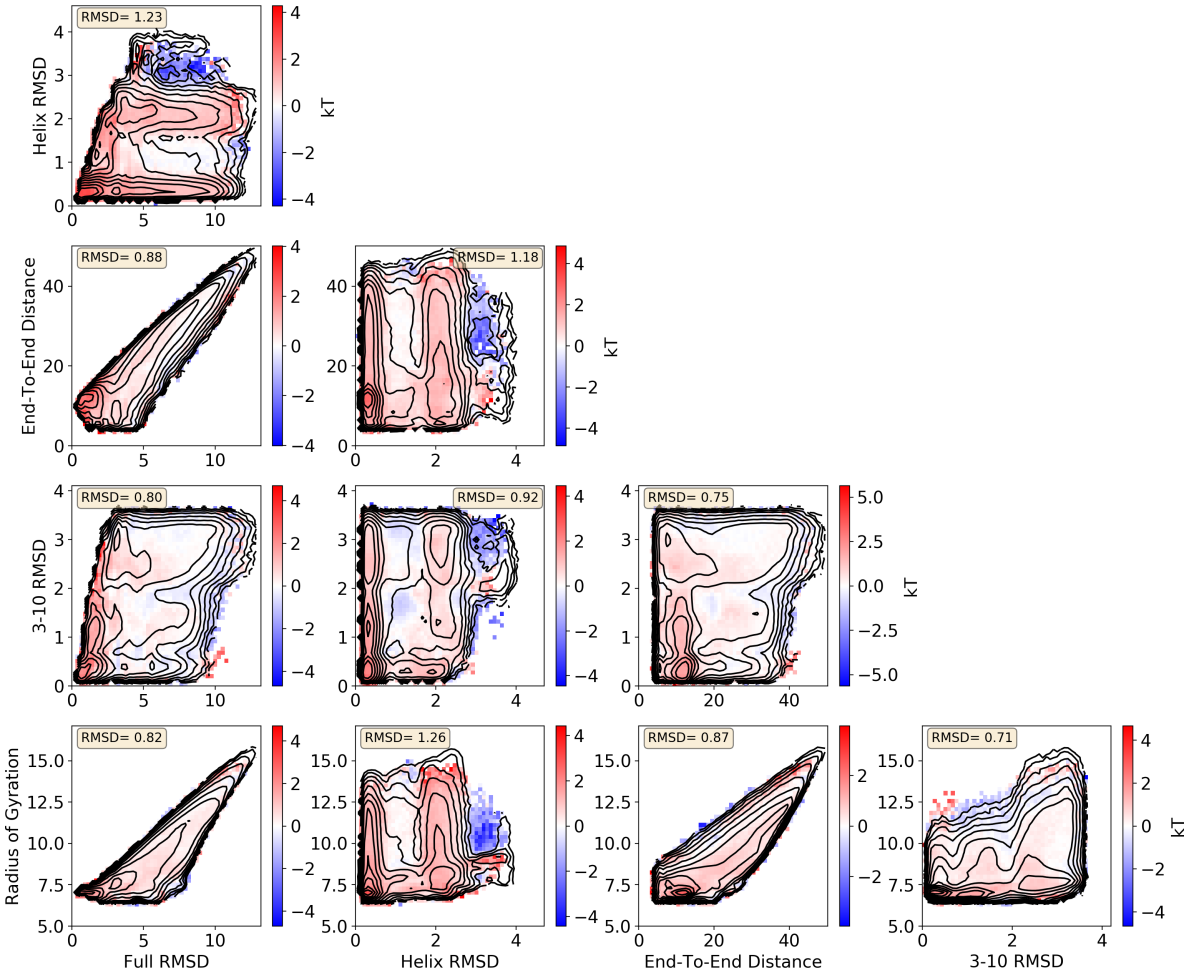


Figure 2.13: Difference between the PMF from DGA with the distance indicator basis set and the PMF from REUS.

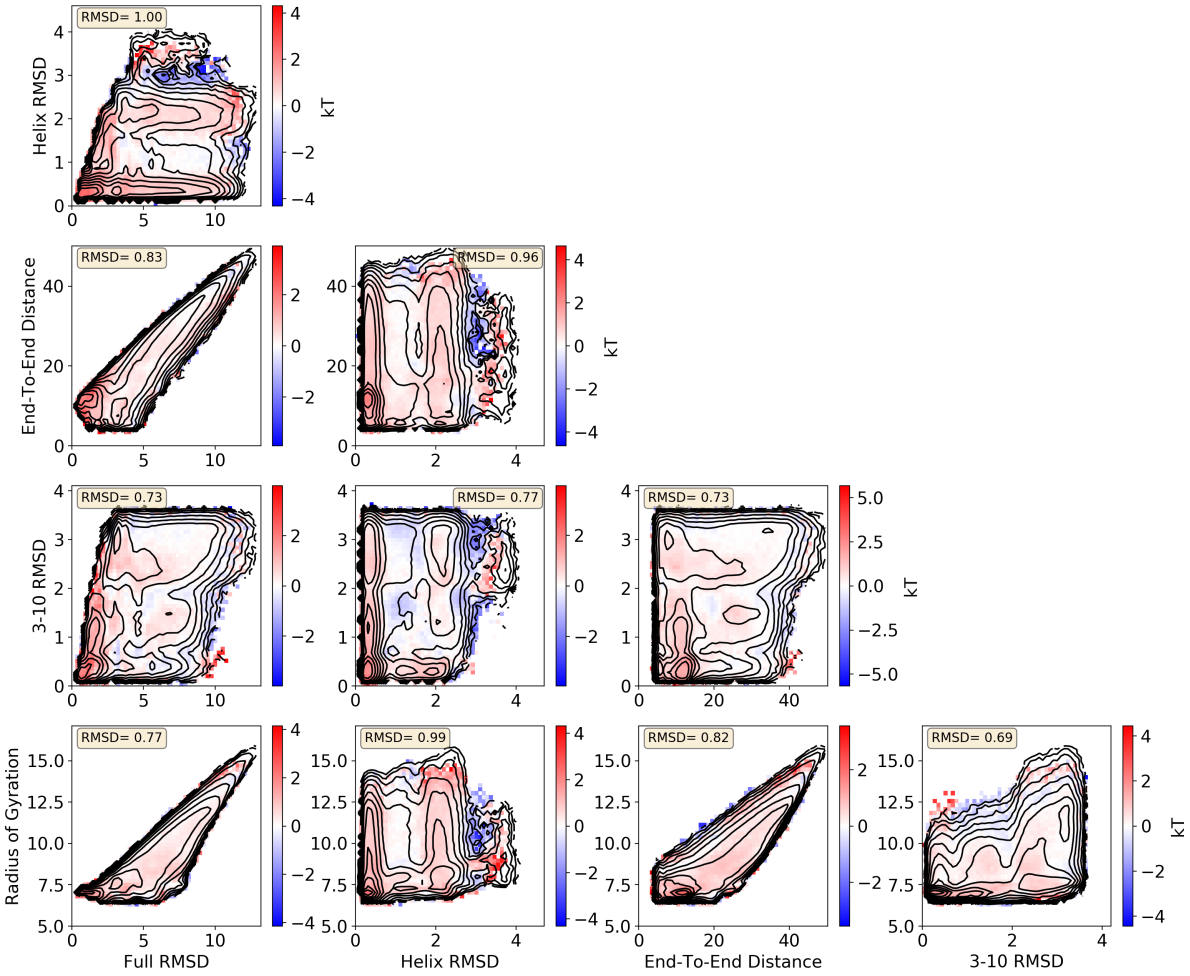


Figure 2.14: Difference between the PMF from DGA with the TICA indicator basis set and the PMF from REUS.

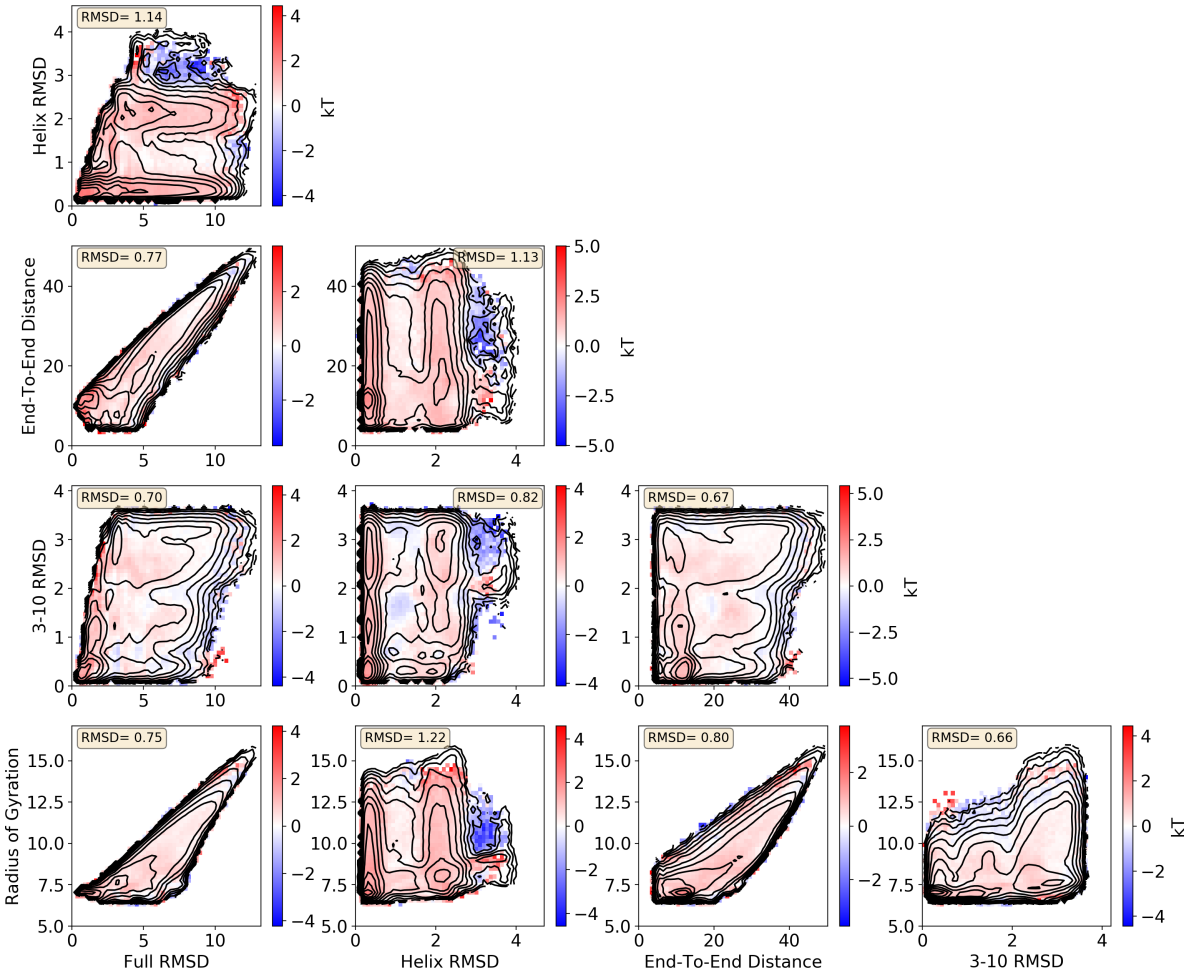


Figure 2.15: Difference between the PMF from DGA with the modified distance basis set and the PMF from REUS.

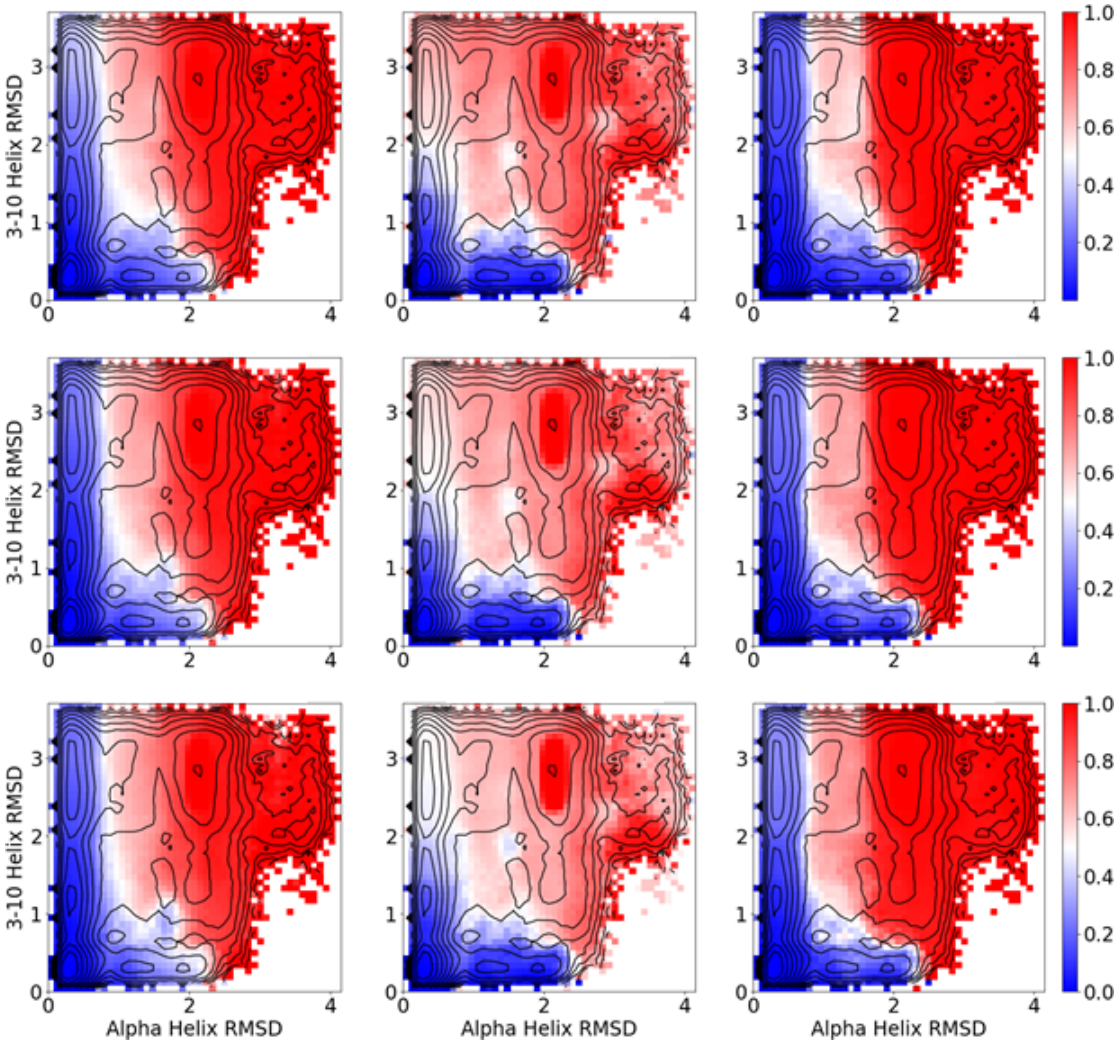


Figure 2.16: DGA backward committors. Left, middle, and right columns are computed with the modified distance, distance indicator, and TICA indicator basis sets, respectively. Top, middle, and bottom rows are computed with lag times of 0.5, 2.5, and 7.5 ns, respectively.

# CHAPTER 3

# PREDICTING WITH MACHINE LEARNING

## 3.1 Introduction

In many complex dynamical systems, behaviors of strong interest occur infrequently compared to the system's fastest timescale phenomena. For example, most climate-related destruction is due to extreme weather events (e.g., hurricanes, heat waves, flooding) [84, 85, 86, 87, 88]. More broadly, fluid turbulence in both natural and engineered systems produces intermittent, damaging extreme events [89]. In the molecular sciences, chemical reactions and molecular rearrangements occur on timescales many orders of magnitude longer than the timescale of individual bond vibrations [90, 91]. In the biomedical sciences, it may take many mutations before a virulent strain of a pathogen emerges [92], or many heart beats before a cardiac arrhythmia becomes life-threatening [93, 94].

Among the most common computational tasks related to these rare events is prediction—assessing the likelihood and extent of an event (i.e., the risk and cost in the case of a deleterious event)—before it occurs. When the event is not too rare, it can often be predicted with sufficient accuracy by direct forward-in-time integration of a computer model as is frequently done, for example, in weather prediction. However, when the event is very rare, direct forward-in-time integration becomes prohibitively expensive because many simulated model trajectories are required to observe even one instance of the event, leave alone compute statistics. The computational cost increases further when the goal is to gain an understanding of how the rare event develops, which requires predictions generated from many initial conditions.

One common approach to this problem is to construct a "coarse-grained" model, in which some details of the system are treated implicitly [95, 96, 97]. One example is a Markov State Model (MSM), in which one groups the states of the full system into discrete sets

and then evolves the system between these sets according to transition probabilities that are estimated from trajectories of the full system [98, 99, 100, 25]. A variety of machine learning approaches that instead yield continuous coarse-grained representations of systems have come to be known as "equation discovery" [101, 102, 103, 104, 105, 106]. When an accurate coarse-grained model can be constructed, it can be simulated extensively to make predictions with statistical confidence. However, building an accurate coarse-grained model can be challenging, in particular, because it is often not clear a priori which features must be included. The construction of coarse-grained models thus remains a subject of intense inquiry.

Here, we pursue an alternative approach: directly estimating conditional expectations of a Markov process as a function of initial condition. We term these conditional expectations "prediction functions." Prediction functions can be used to reveal how a rare event develops in remarkable detail. For example, the committor (also known as the splitting probability)—the probability that a process proceeds to a set of target states before a competing set of states—can be used to define the transition state ensemble of a molecular rearrangement [107, 108], as well as the pathways that lead between the reactant and product states [109, 110]. Prediction functions can also provide important information for decision making. For example, the committor can be used by energy, transportation, and financial sectors to measure risk due to extreme weather and allocate resources accordingly [111]. Committor estimation is a growing research focus in meteorology [112, 113, 114, 115]. In real-time settings, the lead time—the expected time until onset of the event given that it occurs—is also essential to know [116, 117, 115].

Prediction functions satisfy Feynman-Kac equations, linear equations of the operator that describes the evolution of expectations of functions of a process, the transition operator (also known as the Koopman operator [24]) and its infinitesimal generator [118, Chapter 3]. Feynman-Kac equations cannot be solved by conventional discretization approaches because

they involve a high-dimensional independent variable (the state of the underlying process). Moreover, the form of the transition operator is generally not known. Nonetheless, we showed recently that Feynman-Kac equations can be solved approximately by a basis expansion in which inner products of basis functions are estimated from a data set of short trajectories [109, 3].

While this approach has been successfully applied to such diverse processes as protein folding [109], molecular dissociation [110], and sudden stratospheric warming [116], it relies on identifying an effective basis set. One choice is to use a basis of indicator functions for discrete sets, in which case the approach reduces to construction of an MSM (but with appropriate boundary conditions for the prediction function). However, just as it can be challenging to group states into sets that satisfy the Markov assumption in construction of an MSM [25, 44], the choice of basis set is not always straightforward.

Here, we address this issue through a neural network ansatz for prediction functions. Our work builds on recent studies, which showed that a neural network ansatz can be used to solve for the committor if one assumes particular, explicit forms for the dynamical operator [119, 120, 121, 122] (and see [123] for a closely related approach using tensor network approximation). Similar neural-network techniques have been devised to solve a wide variety of partial differential equations [124, 125, 126, 127, 128, 129]. Because we work directly with a data set of short trajectories, our approach is free of restrictive assumptions about the dynamics (e.g., microscopic reversibility) and does not require explicit knowledge of a model generating the data, opening the door to treating high-fidelity models, and even experimental and observational data [115], without simplifying assumptions.

In Section 3.2, we review prediction functions and the Feynman-Kac equation that we need to solve to estimate them. In Section 3.3 we introduce our neural network approach to solving Feynman-Kac equations using a data set of paired trajectories. In Section 3.4, we compare with Galerkin methods and explore the role of the lag time and the distribution of

trajectory initial conditions on performance. In Section 3.5 we introduce an adaptive sampling method that enriches the data set based on the current neural network approximation. Finally, in Section 3.6 we apply our algorithm to estimating the probability of onset and the lead time of a sudden stratospheric warming event.

## 3.2 Prediction functions and their Feynman-Kac equations

We consider events defined by a set of target states $B$; often, there is also a competing set of states $A$. For example, if we want to estimate the probability that a moderate storm develops into an intense hurricane before dissipating, we would take $B$ to include all weather states consistent with an intense hurricane and $A$ to include all quiescent states. The initial moderate storm would be a state in the domain $D = (A \cup B)^c$. Mathematically, we select states in $B$ with the indicator function

$$\mathbb{1}_B(x) = \begin{cases} 1, & x \in B \\ 0, & x \notin B, \end{cases} \tag{3.1}$$

where $x$ denotes a particular state of the system. We define analogous indicator functions for other sets.

We assume the dynamics of the system can be described by a Markov process $X^t$. In the example above, $X^0 = x$ is a moderate storm state, and the probability that it develops into an intense hurricane before the weather returns to a quiescent state is the committor:

$$q(x) = \mathbb{P}_x[X^T \in B] = \mathbb{E}_x[\mathbb{1}_B(X^T)], \tag{3.2}$$

where the subscript indicates the initial condition, and $T = \inf\{t > 0 : X^t \in A \cup B\}$ is the stopping time, i.e., when the process leaves the domain $D = (A \cup B)^c$.

Continuing the example above, we may also want to compute the lead time, i.e., the

average time until a moderate storm develops into an intense hurricane, given that the intense hurricane occurs ($B$ occurs before $A$). The lead time tells us how much time we have to prepare for the worst case; by definition, it is shorter than the average time until an intense hurricane develops, which can be misleadingly large if the storm has a high probability of dissipating ($A$ occurs before $B$). Mathematically, the lead time is

$$m_{AB}(x) = \frac{\mathbb{E}_x[T\mathbb{1}_B(X^T)]}{\mathbb{E}_x[\mathbb{1}_B(X^T)]}. \tag{3.3}$$

When the event of interest is rare, computing $q(x)$ or $m_{AB}(x)$ by direct forward-in-time simulation is difficult. It involves repeatedly simulating $X^t$ starting in a selected initial condition $x$ and running until either $A$ or $B$ is reached (which defines the stopping time $T$), and then assembling a sample average. This approach has significant drawbacks: first, when the time $T$ is very large, generation of a single sample trajectory may be prohibitively computationally expensive, and second, when $q(x)$ is small, many sample trajectories will be required to observe a single trajectory reaching $B$. For example, starting from a typical weather state, the expected time to the next extreme event may be years, and the probability that it occurs on a much shorter time scale may be very small.

In this paper we estimate prediction functions by solving operator equations for them approximately. In the case of the committor, the operator equation takes the form

$$(\mathcal{S}^\tau - \mathcal{I})[q](x) = 0 \text{ with } q(x) = 0 \text{ for } x \in A \text{ and } q(x) = 1 \text{ for } x \in B, \tag{3.4}$$

where $\tau$ is a time interval known as the lag time and $\mathcal{I}$ is the identity operator. Here we focus on finite $\tau$; the case of infinitesimal $\tau$ is discussed in Section 3.4.4. Above, the stopped transition operator $\mathcal{S}^s$ encodes the full dynamics of the system when it is in $D$; it is defined

by its action on an arbitrary test function $f$:

$$\mathcal{S}^\tau[f](x) = \mathbb{E}_x\left[f(X^{\tau\wedge T^+(0)})\right], \tag{3.5}$$

where $\tau \wedge T^+(0) = \min\{\tau, T^+(0)\}$. Physically, (3.4) reflects the fact that the average probability that $B$ occurs before $A$ after time $\tau$ over all trajectories emanating from $X^0 = x$ is the same as the probability that $B$ occurs before $A$ starting from $x$. Similarly, the lead time satisfies

$$(\mathcal{S}^\tau - \mathcal{I})[m_{AB}q](x) = -\mathbb{E}_x\left[\int_0^{\tau\wedge T^+(0)} q(X^s)ds\right] \text{ with } m_{AB} = 0 \text{ for } x \in A\cup B. \tag{3.6}$$

In this case, the right hand side accumulates the time until reaching $A\cup B$, weighted by the likelihood of reaching $B$ before $A$.

Eqs. (3.4) and (3.6) are examples of Feynman-Kac equations [118, Chapter 3], which can take more general forms, such as

$$(\mathcal{S}^\tau - \mathcal{I})[u](x) = -\mathbb{E}_x\left[\int_0^{\tau\wedge T^+(0)} h(X^s)ds\right] \text{ with } u(x) = g(x) \text{ for } x \notin D \tag{3.7}$$

which is solved by the prediction function.

$$u(x) = \mathbb{E}_x\left[g(X^{T^+(0)}) + \int_0^{T^+(0)} h(X^s)ds\right] \tag{3.8}$$

We recover (3.4) by setting $h(x) = 0$ and $g(x) = \mathbb{1}_B(x)$ and (3.6) by setting $h(x) = q(x)$ and $g(x) = 0$; the latter case yields $[m_{AB}q](x)$, and we must solve separately for $q(x)$ and divide by it to obtain $m_{AB}(x)$. Crucially, (3.7) exactly characterizes $u$ for any choice of $\tau > 0$. In particular, $\tau$ can be chosen much shorter than typical values of $T$.

On its own, (3.7) brings us no closer to a practically viable approximation of the prediction function. The independent variable $x$ is typically high-dimensional, rendering useless any

standard discretization approach to solving (3.7) for $u$. Instead, the current state-of-the-art approach involves expansion of $u$ in a problem-dependent basis [109, 110, 3]. In the next section, we explore a potentially more flexible and automated approach to solving (3.7).

## 3.3 Solving Feynman-Kac equations with neural networks

The goal of the present study is to solve (3.7) by approximating $u$ by a neural network $u_\theta$ with a vector of parameters $\theta$. Specifically, we seek $\theta = \theta^*$ that minimizes a mean square difference between the left and right hand sides of the Feynman-Kac equation and boundary condition in (3.7):

$$\theta^* = \arg\min_\theta [C_{\text{FKE}} + \lambda C_{\text{BC}}] \tag{3.9}$$

with

$$C_{\text{FKE}} = \left\| \left( (\mathcal{S}^\tau - \mathcal{I}) u_\theta + \mathbb{E}_x \left[ \int_0^{\tau \wedge T^+(0)} h(X^s) ds \right] \right) \mathbb{1}_D \right\|_\mu^2 \text{ and } C_{\text{BC}} = \left\| (u_\theta - g) \mathbb{1}_{D^c} \right\|_\mu^2 . \tag{3.10}$$

The norm that we use is the $\mu$-weighted $L^2$ norm $\|f\|_\mu^2 = \int |f(x)|^2 \mu(dx)$, where $\mu$ is the sampling distribution. Importantly, unlike the many existing estimators [119, 120, 121, 122, 123, 130, 131, 132, 133, 134], our data need not be generated from (or re-weighted according to) the invariant distribution of $X_t$ (which may not exist), a feature that we exploit in Section 3.4.5. In (3.10), $C_{\text{FKE}}$ and $C_{\text{BC}}$ are both zero when $u_\theta$ equals the desired prediction function. The parameter $\lambda$ controls the strength of the first norm, which enforces the Feynman-Kac equation, relative to the second norm, which enforces the boundary condition. Smaller values enforce the boundary conditions more strictly but can compromise the satisfaction of the Feynman-Kac equation. For our numerical tests below, we tuned $\lambda$ by trial and error to the smallest value that still enforced the boundary conditions to the desired precision.

The gradient of $C_{\text{FKE}}$ includes the integral of a product of two terms of the form $\mathcal{S}^\tau v$

with $v = u_\theta$ and $v = \partial_\theta u_\theta$, the gradient of $u_\theta$ with respect to the parameters $\theta$. While we cannot hope to evaluate $\mathcal{S}^\tau v$ exactly for any non-trivial $v$, as long as we can evaluate $v$ we have access to the random variable $v(X^{\tau \wedge T^+(0)})$ whose expectation is $\mathcal{S}^\tau v$. With only one sample of $X^{\tau \wedge T^+(0)}$ for each sample of $X^0$, we would not be able to build an unbiased estimate of the product of two terms of the form $\mathcal{S}^\tau v$. One approach, common in reinforcement learning applications, is to simply drop the term involving this product from the gradient [135]. However, given at least two independent samples of $X^{\tau \wedge T^+(0)}$ for each sample of $X^0$, we can construct an unbiased estimator of the full gradient of $C_{\text{FKE}}$ that converges to the exact gradient of $C_{\text{FKE}}$ in the limit of many samples of $X^0$ (even when the number of independent samples of $X 6\tau \wedge T^+(0)$ for each sample of $X^0$ does not increase). Below we outline a procedure that constructs an unbiased estimate of the gradient of $C_{\text{FKE}}$ given a data set of samples of $X^0$, together with $\ell \geq 2$ samples of $X^{\tau \wedge T^+(0)}$ for each sample of $X^0$ (in tests of $2 \leq \ell \leq 10$, we found the results to be insensitive to the choice of $\ell$, and we use $\ell = 2$ throughout).

Our procedure is as follows.

1. Select a set of $n$ initial conditions $X_i^0$ from the sampling distribution $\mu$.

2. From each $X_i^0$, launch $\ell$ independent unbiased simulations to generate trajectories $\{(X_i^0, X_{i,j}^\Delta, ..., X_{i,j}^{S\Delta})\}_{j=1}^\ell$. Here we assume that $\tau = S\Delta$.

3. For trajectory $j = 1, 2, \ldots, \ell$ with initial condition $X_i^0$, determine the index of its stopping time as $k_{i,j} = \min\{s' : X_{i,j}^{s'\Delta} \in (A \cup B) \text{ or } s' = S\}$.

4. Given the data set of grouped trajectories, approximate the first norm in (3.9) as

$$\begin{aligned}\bar{C}_{\mathrm{FKE}} = \frac{1}{n}\sum_{i=1}^{n}\left(\frac{1}{|S_i|}\sum_{j\in S_i} u_\theta(X_{i,j}^{k_{i,j}\Delta}) - u_\theta(X_i^0) + \Delta\sum_{s=0}^{k_{i,j}} h(X_{i,j}^{s\Delta})\right) \\ \times\left(\frac{1}{|S_i'|}\sum_{j'\in S_i'} u_\theta(X_{i,j'}^{k_{i,j'}\Delta}) - u_\theta(X_i^0) + \Delta\sum_{s=0}^{k_{i,j'}} h(X_{i,j'}^{s\Delta})\right)\mathbb{1}_D(X_i^0)\end{aligned} \tag{3.11}$$

   and the second norm in (3.9) as

$$\bar{C}_{\mathrm{BC}} = \frac{1}{n}\sum_{i=1}^{n}\left(u_\theta(X_i^0) - g(X_i^0)\right)^2 \mathbb{1}_{D^c}(X_i^0). \tag{3.12}$$

   where $S_i$ and $S_i'$ are randomly chosen index sets such that $S_i \cap S_i' = \emptyset$.

5. Compute the total approximate loss function as

$$\bar{C} = \bar{C}_{\mathrm{FKE}} + \lambda\bar{C}_{\mathrm{BC}}, \tag{3.13}$$

   which converges to the loss in (3.9) as $n$ increases.

6. Adjust the parameters to minimize (3.13).

7. Check termination criteria and stop if met (discussed further below).

8. If adaptively sampling, apply the procedure in Section 3.5 and set $n$ to the total number of initial conditions.

9. Go to step 3.

In principle, the loss can be minimized over any sufficiently flexible ansatz $u_\theta$. In this work, $u_\theta$ is a fully connected feed-forward neural network, and we determine the optimal parameters via the Adam algorithm [136]. In the present study, we stop training (step 7) when the

average loss for an epoch is less than zero. It is possible for $\bar{C}_{\mathrm{FKE}}$ to become negative because the two parenthetical factors in (3.11) are evaluated using independent samples of $X^{\tau\wedge T^+}(0)$. While this could be avoided by choosing $S_i = S_i'$, the result would be a biased estimator of $C_{\mathrm{FKE}}$. In the limit of large $n$, $\bar{C}_{\mathrm{FKE}}$ converges to $C_{\mathrm{FKE}}$, which must be non-negative. When using any sample approximation of $C_{\mathrm{FKE}}$, some regularization is required to avoid overfitting. We find early stopping at the first occurrence of a negative value of $\bar{C}_{\mathrm{FKE}}$ to be a natural and effective approach. Further details are given in conjunction with the numerical examples.

## 3.4 Illustration of numerical considerations

In this section, we use a model for which we are able to compute reference results to illustrate the advantages of our approach relative to existing ones. Specifically, we compare our approach with one that employs a basis expansion (a Markov State Model) and one that employs a neural network with an assumed form for the dynamical operator. Finally, we examine common choices for the sampling distribution. We show that an important practical advantage of our approach is the freedom to choose the sampling distribution $\mu$ with which to weight the norm in $C_{\mathrm{FKE}}$.

### *3.4.1 Müller-Brown model*

The system that we study is specified by the Müller-Brown potential [2], which is a sum of four Gaussian functions:

$$V_{\mathrm{MB}}(y,z) = \frac{1}{20}\sum_{i=1}^{4} C_i \exp[a_i(y-z_i)^2 + b_i(y-y_i)(z-z_i) + c_i(z-z_i)^2]. \tag{3.14}$$

For all results shown, we use $C_i = \{-200, -100, -170, 15\}$, $a_i = \{-1, -1, -6.5, 0.7\}$, $b_i = \{0, 0, 11, 0.6\}$, $c_i = \{-10, -10, -6.5, 0.7\}$, $y_i = \{1, -0.27, -0.5, -1\}$, $z_i = \{0, 0.5, 1.5, 1\}$.

The potential is shown in Figure 3.1(left).

We consider the overdamped Langevin dynamics associated with $V_{\mathrm{MB}}$, discretized with the BAOAB algorithm [137]:

$$X^{t+dt} = X^t - \nabla V(X^t)dt + \sqrt{\frac{dt}{2\beta}}(Z^t + Z^{t-dt}) \tag{3.15}$$

where $dt$ is the time step, $\beta$ is the inverse temperature, $Z^t \sim N(0,1)$, $N(0,1)$ is the normal distribution with zero mean and unit standard deviation (i.e., $Z^t$ is Gaussian noise), and $V = V_{\mathrm{MB}}$ with $\beta = 1$ unless otherwise specified. In practice, we use a time step of $dt = 0.001$, saving the configuration every time step, such that $\Delta = 0.001$ (cf. step 2 in Section 3.3). When the parameter $\beta$ is large, $X^t$ makes only very rare transitions between the local minima of $V_{\mathrm{MB}}$.

We define states $A$ and $B$ as

$$\begin{aligned} A &= \{y, z : 6.5(y + 0.5)^2 - 11(y + 0.5)(z - 1.5) + 6.5(z - 1.5)^2 < 0.3\} \\ B &= \{y, z : (y - 0.6)^2 + 5(z - 0.02)^2 < 0.2\}, \end{aligned} \tag{3.16}$$

neighborhoods of two of the three local minima of $V_{\mathrm{MB}}$ (Figure 3.1(left)).

The Müller-Brown model described above is commonly employed as a simple illustration of the features of molecular rearrangements [2, 3, 120, 121, 122]. The presence of local minima in addition to $A$ and $B$, and the fact that, at low noise, the trajectories connecting the minima do not align with the coordinate axes are both features that can be challenging for algorithms that enhance the sampling of transitions between the reactant ($A$) and product ($B$) states. In our tests, we specifically focus on the committor. We compute a reference committor by the finite difference scheme outlined in the appendices of [3, 4] with $\epsilon = 0.0125$. In all tests, we compare the estimated committor to the reference committor computed using the same potential energy function used to generate the data.

Finally, to represent the fact that one of the most challenging aspects of treating complex systems is that the manifold on which the dynamics take place is generally not known, we transform the trajectories and sets $A$ and $B$ to a new set of coordinates. Specifically, we map the two-dimensional system onto a Swiss roll (Figure 3.1(right)):

$$\begin{bmatrix} y \\ z \end{bmatrix} \rightarrow \begin{bmatrix} (c+y)\cos((y+c)f) \\ z \\ (c+y)\sin((y+c)f) \end{bmatrix}, \tag{3.17}$$

where the parameter $f$ controls how tightly the roll is wound, and $c$ is an offset to ensure that the range of $x$ is positive. Unless otherwise specified, we use $f = 3$ and $c = 1.8$. For the remainder of the tests based on the Müller-Brown potential, we use the three-dimensional coordinates as input features for all neural networks and $k$-means clustering. For clarity of visualization, we plot the estimated committors on the original two-dimensional coordinates. The error metric that we use is independent of coordinate system.

Unless otherwise specified, for our experiments with the Müller-Brown model below, we draw 30,000 initial conditions uniformly from the region:

$$\Omega = \{y, z : -1.5 < y < 1.0, -0.5 < z < 2.5, V(y, z) < 100\}. \tag{3.18}$$

Two independent trajectories of length $\tau$ (to be specified below) are then generated from each initial condition using (5.32).

### *3.4.2 Neural network details*

For all the numerical experiments involving the Müller-Brown potential, we use fully connected feed-forward neural networks with three inputs, three hidden layers, each consisting of 30 sigmoid activation functions, and an output layer with a single sigmoid activation

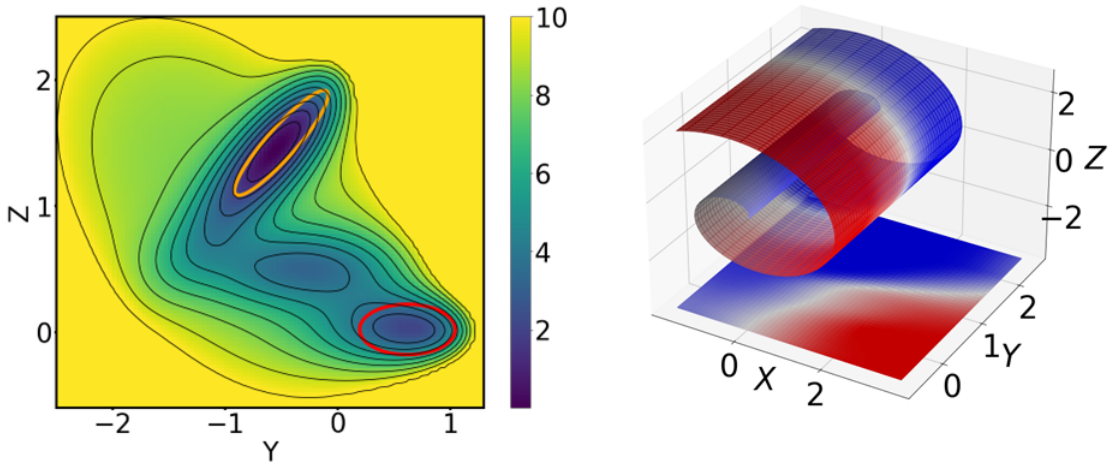


Figure 3.1: The system used for the numerical experiments in Section 3.4. (left) Müller-Brown potential [2]. Sets $A$ and $B$ are marked by the orange and red ellipses, respectively, and contours are spaced at intervals of 1 in the units of (5.31). (right) Reference committor for the Müller-Brown dynamics mapped to the Swiss roll, and below on the two-dimensional surface. We compute the reference from a finite difference scheme [3] in two dimensions and then map it to the Swiss roll using (3.17).

function. In all trials with fixed data sets, we trained for a maximum of 3000 epochs with a learning rate of 0.0005 and a batch size of 1500. Each epoch proceeds by drawing a permutation of the data set, then one step of Adam is performed using mini-batches of size 1500 (that is, 1500 pairs of trajectories) such that each trajectory pair is used exactly once per epoch (that is, the number of Adam steps is the data set size divided by the mini-batch size). The boundary term is computed with the same mini-batch as $\bar{C}_{\mathrm{FKE}}$; we use $\lambda = 1$ to weight the terms in the loss function. We also explored deeper networks with ReLU activation functions, and they performed comparably and generally required shorter training times (results not shown); we focus on the shallower networks with sigmoid activation functions because they allow a direct comparison with loss functions involving explicit derivatives of $u_\theta$ in Section 3.4.4.

### *3.4.3 Galerkin methods*

As discussed in the Introduction, one of our main motivations in introducing an approach based on neural networks is that it can be difficult to identify basis functions for linear (e.g., Finite Element or other Galerkin) methods for solving Feynman-Kac equations. To illustrate this issue explicitly, we compare estimates for the committor from our approach with those obtained from dynamical Galerkin approximation [3, 109] using a basis of indicator functions, which can be considered a MSM [3]. We do so as a function of the parameter $f$ in (3.17) and generate data sets with $0 \leq f \leq 10$.

To construct an MSM, we clustered the configurations in each data set by the $k$-means algorithm (with $k$ as specified below) applied to the three-dimensional coordinates of the model. The indicator functions of the set of points closest to each cluster centroid form a basis for a Galerkin approximation of the committor function. A transition matrix $\mathbf{T}$ was constructed by counting transitions of the stopped process between clusters among our trajectory data set with a lag time of $\tau = 150\Delta$. Here we use the convention that the row and column indices are zero for $A$ and their maximum values for $B$. The committor is then computed from $\mathbf{T}\mathbf{q}_+ = \mathbf{q}_+$ with the last component of the solution vector set to 1. The neural network and its training were as described in Section 3.4.2.

Figure 3.2 shows the results. We see that as the roll is wound tighter (higher $f$), the MSM estimates, constructed with a constant 300 clusters, decrease in accuracy, while the network estimates remain consistently good. In the right panel, we vary the number of clusters and report the number required to reach a root mean squared error threshold of 0.045. This threshold is chosen because it results in numbers of clusters in a range that is typical in MSM studies [109, 116]. We increase the number of trajectories in proportion to the number of MSM clusters to ensure that each cluster is sampled a consistent amount. In this test, we see that large numbers of MSM clusters, and hence large amounts of data, are needed. Intuitively, the MSM encounters problems when a single cluster spans adjacent

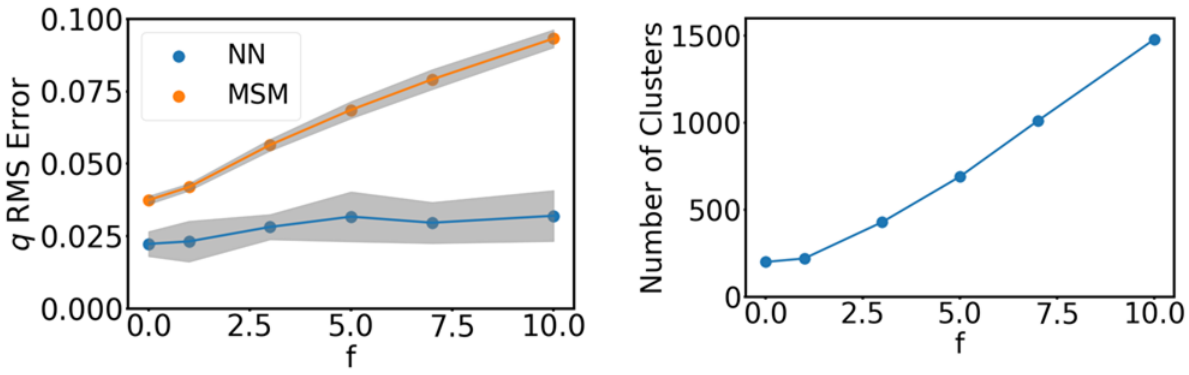


Figure 3.2: Comparison with Galerkin methods. (left) For an MSM estimate with $k = 300$ clusters, the root mean square (RMS) error in the committor for the Müller-Brown model as the Swiss roll is wound tighter (higher $f$ in (3.17)). Shading shows the standard deviation in the error from training on ten independent data sets. (right) Number of MSM clusters needed to achieve an RMS error in the committor of less than 0.045.

layers. Therefore, it is necessary to vary the size of the clusters with the distance between layers of the Swiss roll, which is a linear function of $1/f$. Consistent with this idea, in Figure 3.2 we find an approximately linear dependence of the number of clusters needed to achieve a certain error threshold.

We note that in practice MSMs are often constructed on coordinates obtained from a method for dimensionality reduction and/or manifold learning. With such pre-processing, linear methods can clearly be successful. However, kernel-based methods for dimensionality reduction (e.g., diffusion maps [37] or kernel time-lagged independent component analysis [138, 139]) scale poorly with the size of the data set. A neural network (e.g., an autoencoder [140, 141]) can be used for dimensionality reduction, but the approach presented here is simpler in that we go directly from model coordinates to prediction function estimates.

### *3.4.4 Lag time*

As discussed in the Introduction, neural networks have been applied to estimating high-dimensional committors assuming a partial-differential form for the dynamical operator [119,

120, 121]. This form arises in the limit that one considers an infinitesimal lag time. In this case, one can write (3.7) as

$$\mathcal{L}[u](x) = h(x) \text{ for } x \in D \text{ and } u(x) = g(x) \text{ for } x \notin D, \tag{3.19}$$

where $\mathcal{L}$ is the is the infinitesimal generator:

$$\mathcal{L}[f](x) = \lim_{dt \to 0} \frac{\mathbb{E}_x \left[ f(X^{dt}) \right] - f(x)}{dt}. \tag{3.20}$$

For a diffusion process, $\mathcal{L}$ takes the form

$$\mathcal{L}f(x) = \sum_{i=1}^{\kappa} b_i(x) \frac{\partial f}{\partial x_i} + \frac{1}{2} \sum_{i,j=1}^{\kappa} (\sigma\sigma^{\mathrm{T}})_{ij}(x) \frac{\partial^2 f}{\partial x_i \partial x_j}, \tag{3.21}$$

where $b \in \mathbb{R}^{\kappa}$ and $\sigma \in \mathbb{R}^{\kappa \times \kappa}$ are the drift and diffusion coefficients that determine the evolution of $X_t$. In the limit of small $dt$, the dynamics in (5.32) correspond to a generator with $b = -\nabla V$ and $\sigma = \sqrt{2/\beta}$. In this case, the loss function becomes

$$\bar{C}_{\mathrm{FKE}} = \sum_{i=1}^{n} \left( \sum_{j=1}^{\kappa} b_j(X_i^0) \frac{\partial u_\theta(X_i^0)}{\partial x_j} + \frac{1}{2} \sum_{j,l=1}^{\kappa} (\sigma\sigma^{\mathrm{T}})_{jl}(X_i^0) \frac{\partial^2 u_\theta(X_i^0)}{\partial x_j \partial x_l} - h(X_i^0) \right)^2 \mathbb{1}_D(X_i^0) \tag{3.22}$$

with an appropriate boundary condition term.

The loss function in (3.22) differs from the one used in many recent articles on the subject of committor estimation with neural-network (or recently tensor-network) approximations [119, 120, 121, 122, 123]. Those papers focus specifically on the case of reversible overdamped diffusive dynamics. In this case the committor can be found by minimizing a sample approximation of the loss function $\|\nabla q\|_\pi^2$ (for constant, isotropic diffusion coefficient) where $\pi$ is the invariant distribution of the dynamics [134]. Relatedly, despite a resemblance to

(3.10), the estimator $\left\| \left( q_+(X^\tau) - q_+(X^0) \right) \right\|_\pi^2$ that appears in [130, 131, 132, 133] is, in fact, a small $\tau$ approximation of $\|\nabla q\|_\pi^2$.

We stress that (3.22) is only appropriate for diffusion processes and requires working with the full set of variables in which the dynamics are formulated. Importantly, one generally analyzes only functions of a subset of the variables (termed collective variables or order parameters) [109, 110, 115, 3]. For example, in a molecular simulation of a solute in solvent, one may include only the dihedral angles of the solute. In a weather simulation, one may focus on the wind speed and geopotential height at particular altitudes. When working with observational data, one only has access to the features that were measured. Even when the tracked variables can be described by an accurate coarse-grained model, that model is not known explicitly and is difficult to identify from data. These considerations make minimization of any loss function explicitly involving (3.21) impossible for many practical applications.

Nonetheless, the loss function in (3.22) is appealing because it involves only a single time point, so no trajectories need to be generated if explicit forms for the drift and diffusion coefficients are known. While this would appear to be an advantage, we show in this section that, even when the dynamics can be reasonably described by (3.21), it can be preferable to work with finite lag times.

To make this point, we consider dynamics governed by the Müller-Brown potential with a small oscillating perturbation (Figure 3.3A):

$$V(x) = V_{\mathrm{MB}}(x) + 0.1 \sin(2\pi\omega x) \sin(2\pi\omega y), \tag{3.23}$$

where $\omega$ controls the spatial frequency of the perturbation. Again we represent the data on the Swiss roll as described in Section 3.4.1. As shown in Figure 3.3B, the perturbation is sufficiently small that it makes no qualitative change to the committor.

Given this data set, we train neural networks to minimize the loss function in (3.13),

using either (3.11) or (3.22) for $\bar{C}_{\mathrm{FKE}}$, with $h(x) = 0$ and $g(x) = \mathbb{1}_B(x)$, corresponding to the committor. The network architecture was the same as above: i.e., fully connected feed forward with two inputs, 30 activation functions per hidden layer, and one output. The neural network and its training were as described in Section 3.4.2.

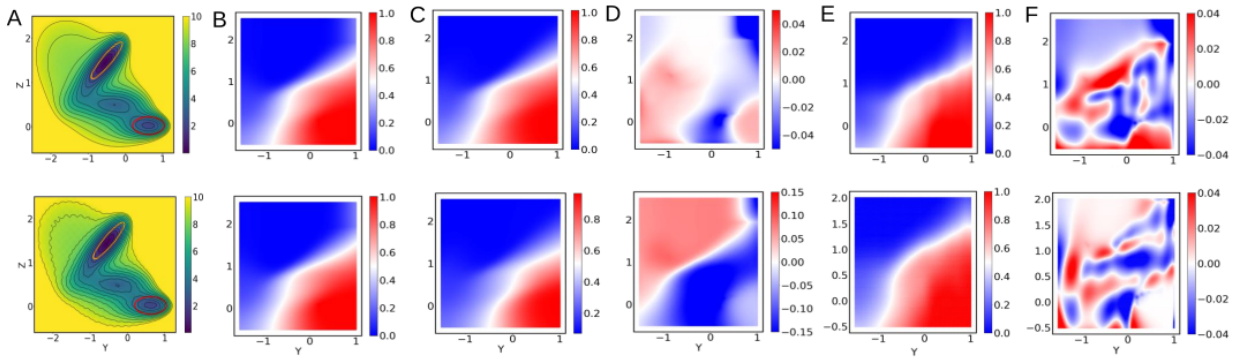


Figure 3.3: Effect of potential roughness on the performance when using a loss function based on the infinitesimal lag time limit. Results shown are obtained with (3.22) and (3.23) with $\omega = 0$ (top row), and $\omega = 10$ (bottom row). (A) The potentials. (B) Reference committors obtained from the finite-difference scheme in [3, 4]. (C) Neural network prediction of the committors. (D) Differences between the references and the predictions. (E,F) Same as columns C and D, except for a lag time of 100 steps. Note the different scales on the difference maps in columns D and F.

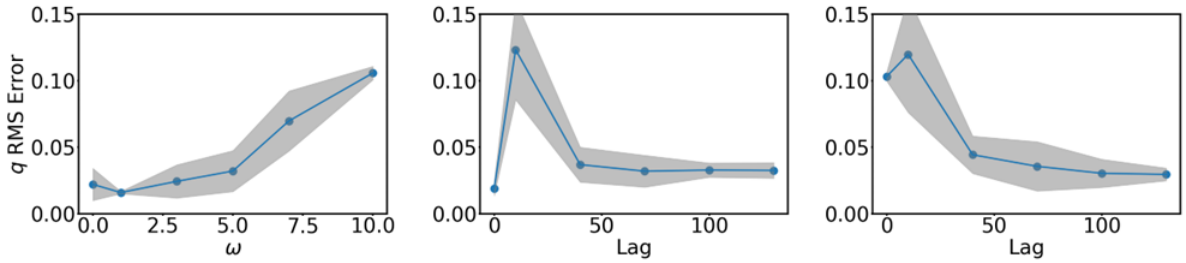


Figure 3.4: Comparison of infinitesimal and finite lag time loss functions. (left) RMS error in the committor for the Müller-Brown dynamics mapped to the Swiss roll obtained with the infinitesimal lag time loss function in (3.22) as the frequency of the sinusoidal perturbation, $\omega$, is increased. The other panels show the error as the lag time is increased with the frequency fixed at $\omega = 0$ (center) or $\omega = 10$ (right). Shading shows the standard deviation in the error from training on ten independent data sets.

Typical results are shown in Figure 3.3C and D, and the error in the committor is quantified in Figure 3.4. As the frequency of the the perturbation increases, the drift becomes large, with rapid sign changes, and the training of the infinitesimal lag time network tends to get stuck at poor estimates of the committor (Figure 3.4(left)). By contrast, finite lag time networks consistently achieve low errors at longer lag times (Figure 3.4(center and right)). This presumably results from averaging over values of the drift. Interestingly, we found that when the potential is smooth (Figure 3.4(center)), slightly lower errors can be obtained using the zero lag time approach. However, in the presence of even such a small amount of roughness that the committor is qualitatively unchanged (Figure 3.3A and B), our finite lag time approach performs better (Figure 3.4(right)). We expect the latter case to be more relevant in many practical applications.

It may be tempting to assume that the zero lag time approach has lower computational cost since there is no need to actually simulate the stochastic differential equation (here, (5.32)). This is not necessarily the case. With the infinitesimal lag time loss function, the drift needs to be evaluated for each data point for every pass over the data set (one epoch). By contrast, the finite lag time loss function introduced here does not require evaluation of the drift once the data set is generated. Therefore, if the number of epochs needed to train the zero lag time network is comparable to the number of time steps used to generate the data set for the two-trajectory method, the finite lag time method will be less computationally costly.

### *3.4.5 Sampling distribution*

In this section, we investigate the role of the choice of sampling distribution. Following generation of a data set as described in Section 3.4.1, we selected initial points from the region specified in (4.25) with weight $\mu(x) \propto \exp(-\beta_s V(x))$ ($\beta_s$ need not be the same as $\beta$) and trained over a range of $\beta_s$ values. To a good approximation when $dt$ is small, the

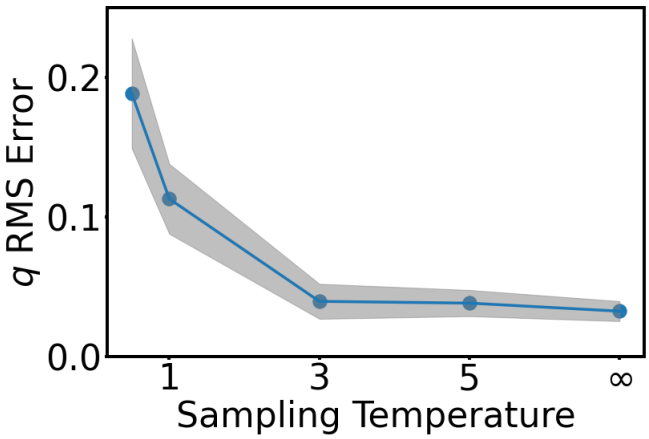


Figure 3.5: RMS error of the committor as the sampling temperature, $1/\beta_s$, is increased. The point at $1/\beta_s = 1$ corresponds to the stationary distribution for the Müller-Brown model (in the small $dt$ limit) at the temperature of the dynamics. Increasing the temperature makes the distribution more uniform. The point labeled $\infty$ is uniform. Shading shows the standard deviation in the error from training on ten independent data sets.

invariant density of (5.32) is $\mu$ with $\beta_s = \beta$. When $\beta_s$ is large, the data set of initial conditions concentrates at the local minima of $V(x)$. As $\beta_s$ becomes small, the distribution approaches uniform. While this parametric form for the sampling distribution is convenient for the tests performed in this section, we emphasize that, unlike many existing schemes [119, 120, 121, 132, 133], our algorithm does not require explicit knowledge of the invariant density.

We trained ten networks on each pair of sampling distribution and lag time following the procedure in Section 3.4.2, and the resulting errors and their standard deviations are plotted as a function of $1/\beta_s$ in Figure 3.5. At low $1/\beta_s$, which concentrates the initial points in the minima, the network is unable to find a good solution at any lag time. As $1/\beta_s$ increases and the distribution becomes more uniform, the solution improves significantly. This suggests that it is important to have the regions between minima well-represented in the data set, which is consistent with previous observations [80, 3, 109, 116, 122]. In high-dimensional examples, sampling the transition regions is not straightforward, and we present a solution

to this problem in the next section.

## 3.5 Adaptive Sampling

As we showed in Section 3.4.5, the choice of sampling distribution is important. In this section, we propose a simple method for adding data as the training proceeds. Since the approach depends on constructing a spatial grid we must first select a low-dimensional (e.g., two-dimensional) set of (possibly non-linear) coordinates $\xi(x)$ which, as noted above, we term collective variables. We then partition the space of possible $\xi$ values into bins of equal volume labeled $S_1, S_2, \ldots, S_m$, and estimate

$$P_\eta = \frac{\int \left( (\mathcal{S}^\tau - \mathcal{I})u_\theta + \mathbb{E}_x \left[ \int_0^{\tau \wedge T^+(0)} h(X^s) ds \right] \right)^2 \mathbb{1}_{S_\eta}(\xi(x)) \mu(x) dx}{\int \mathbb{1}_{S_\eta}(\xi(x)) \mu(x) dx} \tag{3.24}$$

for each bin. The weights $P_\eta$ are then used to select bins, and new initial points are then sampled from the selected bins with uniform probability. The essential idea is that we add data to the regions (bins) that contribute most to $C_{\mathrm{FKE}}$.

When adaptively sampling to learn the committor we approximate (3.24) by

$$\hat{P}_\eta = \frac{\sum_{i=1}^n \left( \frac{1}{|S_i|} \sum_{j \in S_i} (u_\theta(X_{i,j}^{k_{i,j}}) - u_\theta(X_{i,j}^0)) \right) \left( \frac{1}{|S_i'|} \sum_{j \in S_i'} (u_\theta(X_{i,j}^{k_{i,j}}) - u_\theta(X_{i,j}^0)) \right) \mathbb{1}_{S_\eta}(\xi(X_{i,1}^0))}{\sum_{i=1}^n \mathbb{1}_{S_\eta}(\xi(X_{i,1}^0))}. \tag{3.25}$$

We compute (3.25) for each bin, select $N$ bins with probability proportional to $\hat{P}_\eta$ with replacement, and sample a single additional initial point from each selected bin. From each of the $N$ new initial points we generate a trajectory. In practice, we observed that (3.25) can become negative for some bins in the same way that $\bar{C}_{\mathrm{FKE}}$ can become negative. In this case, we set all negative probabilities to zero.

The success of our adaptive sampling approach depends on the choice of $\xi$. In the absence

of other knowledge, a reasonable choice is the current estimate of the committor function itself. We adopt this choice to test our adaptive sampling procedure on the Müller-Brown model. A related adaptive sampling approach using stratified sampling [8, 142] based on a current committor estimate is proposed in [122].

The simulation and Swiss roll parameters, as well as neural network and training parameters are the same as above. We initially train with 10,000 pairs of trajectories drawn uniformly from the region in (4.25) for 1000 epochs. Then we alternate between adding $N = 5000$ new pairs of trajectories and training for 500 epochs, for four cycles. We compare to 30,000 trajectory pairs drawn uniformly from the region in (4.25). Results are presented in Figure 3.6. We find that the adaptive sampling and uniform sampling perform similarly at long lag times, although the adaptive procedure gives more reproducible results as shown by the smaller error bars. At short lag times the average error is lower as well. The adaptive sampling procedure concentrates sampling in the transition region, that is, near $q_+ = 0.5$. In the next section, we illustrate the adaptive sampled distribution on our atmospheric model, and we again see that sampling is effectively directed to the transition region. For low noise diffusions, the transition region becomes narrower, and this is reflected by a sharper peak than in Figure 3.6. In our testing, our adaptive sampling scheme remains effective, although more data are required at lower noise. We find that our method works for barriers $< 10/\beta$.

## 3.6 Predicting an atmospheric transition

As a demanding test of our method, we compute the committor and lead time for a model of sudden stratospheric warming (SSW), aiming to improve upon the benchmarks computed in [116]. Like other models of geophysical flows, the dynamics are irreversible and the stationary distribution is unknown. As a consequence, many competing approaches for computing the committor (e.g., [119, 120, 121, 132, 133]) are not applicable.

Typical winter conditions in the Northern Hemisphere stratosphere support a strong

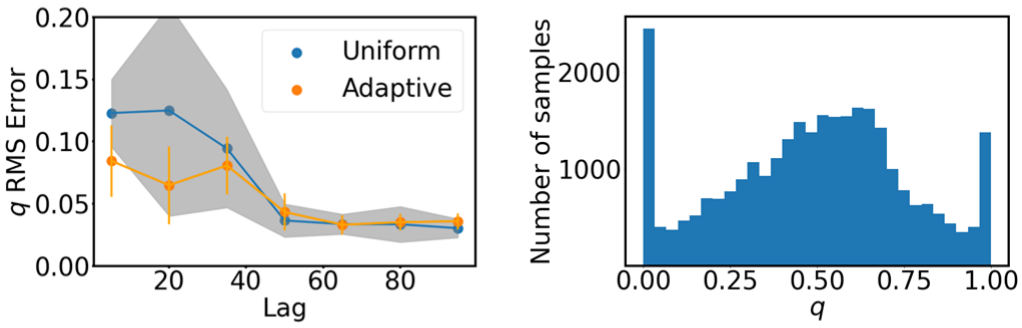


Figure 3.6: Adaptive sampling scheme applied to the Müller-Brown dynamics mapped to the Swiss roll. (left) Comparison of uniform and adaptive sampling. Shading and error bars show the standard deviation in the error from training on ten independent data sets. (right) Histogram of the final data set as a function of the committor from training a neural network with a lag time of 100.

polar vortex, fueled by a large equator-to-pole temperature gradient. Approximately once every two years, planetary waves rising from the troposphere impart a disturbance strong enough to weaken and destabilize the vortex, in some cases splitting it in half. Such events cause stratospheric temperatures to rise by about 50°C over several days, affecting surface weather conditions for up to three months. The polar vortex is dynamically coupled to the midlatitude (tropospheric) jetstream, which sometimes weakens in response to SSW. This can engulf the midlatitudes in Arctic air and alter storm tracks, bringing severe weather conditions to unprepared locations. Predicting SSW events is therefore a prime objective in subseasonal-to-seasonal weather prediction, but their abruptness poses a real challenge. For a review of SSW observations, predictability and modeling, see [143] and references therein.

We consider the Holton-Mass model [144], augmented by time-dependent stochastic forcing as in [116] to represent unresolved processes and excite transitions between the strong and weak vortices. Despite the simplicity of the model relative to state-of-the-art climate models, these transitions capture essential features of SSW such as the rapid upward burst of wave activity mediated by the "preconditioned" vertical structure of zonal-mean flow [145, 146]. We briefly describe the model here, but refer the interested reader to [116, 147, 144, 148] for

additional background and details.

The model domain is the region of the atmosphere north of $30^\circ$ and above the altitude of $z \approx 10$ km (the tropopause). The Holton-Mass model describes stratospheric flow in terms of a wave-mean flow interaction between two physical fields. The mean flow refers to the zonal-mean zonal wind $\overline{u}(y,z,t)$: the horizontal wind velocity component in the east-west (zonal) direction, averaged over a ring of constant latitude (zonal-mean, denoted by the overbar). The spatial coordinate $y$ denotes the north-south (meridional) distance from the latitude line $\phi_0 = 60^\circ$, i.e., $y = a(\phi - \phi_0)$, where $a$ is the Earth's radius and $\phi$ is the latitude. The wave refers to the perturbation streamfunction $\psi'(x,y,z,t)$: the deviation from zonal mean (denoted by a prime symbol) of the geostrophic streamfunction, which is proportional to the potential energy of a given air parcel. Holton and Mass worked with the following ansatz for the interaction:

$$\begin{aligned} \overline{u}(y,z,t) &= U(z,t)\sin(\ell y) \\ \psi'(x,y,z,t) &= \mathrm{Re}\{\Psi(z,t)e^{ikx}\}e^{z/2H}\sin(\ell y) \end{aligned} \tag{3.26}$$

where $k = 2/(a\cos 60^\circ)$ and $\ell = 3/a$ are zonal and meridional wavenumbers, and $H = 7$ km is a scale height. The equations in (3.26) prescribe the horizontal structure entirely, so the model state space consists of $U(z,t)$ and $\Psi(z,t)$, the latter being complex-valued. Insertion of (3.26) into the quasigeostrophic potential voriticity equation yields a system of two coupled PDEs. Following [116, 144, 147, 148], we discretize the PDEs along the $z$ dimension in 27 layers. After enforcing boundary conditions, this results in a 75-dimensional state space:

$$\begin{aligned} X^t = [&\mathrm{Re}\{\Psi(\Delta z,t)\},...,\mathrm{Re}\{\Psi(25\Delta z,t)\}, \\ &\mathrm{Im}\{\Psi(\Delta z,t)\},...,\mathrm{Im}\{\Psi(25\Delta z,t)\}, \\ &U(\Delta z,t),...,U(25\Delta z,t)]. \end{aligned} \tag{3.27}$$

The two states of interest in this model are a strong polar vortex, with large positive

$U(z,t)$ (meaning eastward wind, marked as state $A$ in Figure 3.7), and a weak polar vortex, with a weak wind profile in which $U(z,t)$ sometimes dips negative (marked as state $B$ in Figure 3.7). Specifically, we define $A$ and $B$ as spheres centered on the model's two stable fixed points $(\Psi_{\mathbf{a}}, U_{\mathbf{a}})$ and $(\Psi_{\mathbf{b}}, U_{\mathbf{b}})$ in the 75-dimensional state space. The two spheres have radii of 8 and 20 respectively, with distances measured in the non-dimensionalized state space specified in [116]. In physical units, these correspond to the ellipsoids

$$A = \left\{ \Psi, U : \frac{\|\Psi - \Psi_{\mathbf{a}}\|^2}{\left(7.2 \times 10^5 \text{ m}^2/\text{s}\right)^2} + \frac{\|U - U_{\mathbf{a}}\|^2}{\left(2.9 \text{ m/s}\right)^2} \leq 8^2 \right\} \tag{3.28}$$

$$B = \left\{ \Psi, U : \frac{\|\Psi - \Psi_{\mathbf{b}}\|^2}{\left(7.2 \times 10^5 \text{ m}^2/\text{s}\right)^2} + \frac{\|U - U_{\mathbf{b}}\|^2}{\left(2.9 \text{ m/s}\right)^2} \leq 20^2 \right\} \tag{3.29}$$

where $\|\cdot\|$ is the complex vector 2-norm.

Figure 3.7 illustrates the key features of this model relevant to the prediction problems we consider here. We see that the average time to reach $B$ starting from $A$ is over 1000 days, which is substantially longer than the longest lag times we consider here ($\leq$ 10 days). We can also see that the transition paths do not proceed through the saddlepoint of the effective free energy (i.e., the negative logarithm of the stationary density, marked by the contours), indicating that dynamical, non-diffusive, irreversible dynamics are important. Specifically, the transition path can be roughly divided into two stages: a "preconditioning" phase, in which the vortex gradually weakens, followed by an upward burst of wave activity that rips the vortex apart. Most of the committor's increase happens during the preconditioning phase, which siphons enstrophy (that is, squared vorticity, a measure of vortex strength that is conserved in the absence of dissipation) away from the mean flow and into the wave activity. The wave eventually dissipates, but only after its magnitude $|\Psi|$ bypasses the saddlepoint (Figure 3.7). See [117, 149] for further discussion.

To generate an initial data set, we sampled 30,000 points uniformly in $U$(30 km) and $|\Psi|$(30 km) from a long (50,000 days) trajectory and ran two ten-day trajectories from each

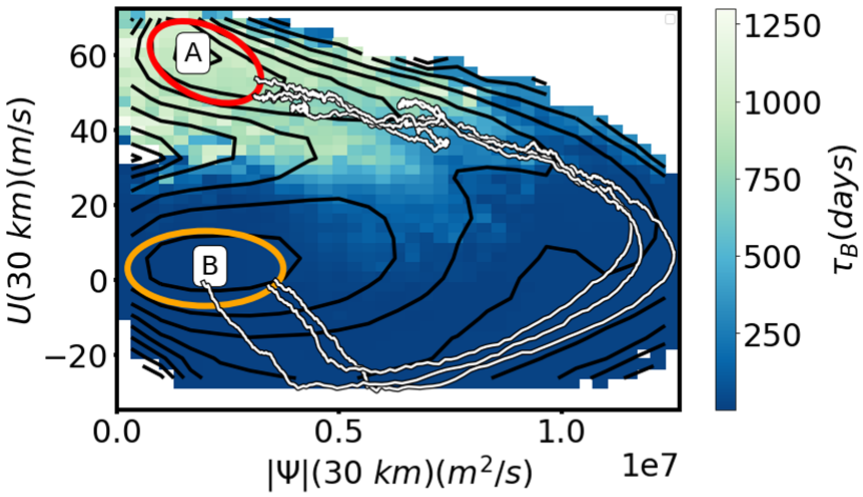


Figure 3.7: Illustration of some key properties of the Holton Mass model relevant to the prediction problems considered here. Red and yellow ellipses approximately mark the projections of states $A$ and $B$, respectively, on the collective variables. The background color shows the average time to hit state $B$, clipped to a maximum of 1300 days to show detail. Black contours show the negative logarithm of the stationary density marginalized on these collective variables. Three transition paths harvested from a long simulation are shown in white.

starting point. Simulation details are reported in [116]. We simulated with a time step of 0.005 days, and saved the state of the system every 0.1 days. To validate our neural network results, we use a long trajectory of 500,000 days to compute

$$\langle q(s)\rangle = \mathbb{E}[\mathbb{1}_B(X^\tau)|u_{\theta^*}(X^0) \in [s, s+\Delta s]] \text{ for } s \in [0,1]. \tag{3.30}$$

where $\theta^*$ are the parameters obtained from solving (3.9). This is the mean reference committor over the isocommittor surfaces from the neural network function. A perfect prediction corresponds to $\langle q(s)\rangle = s$. We use a similar construction for $m_{AB}q$, which we denote $\langle [m_{AB}q](s)\rangle$. For the committor, we take the overall error to be

$$\text{Error} = \sqrt{\int_0^1 (\langle q(s)\rangle - s)^2 ds} \tag{3.31}$$

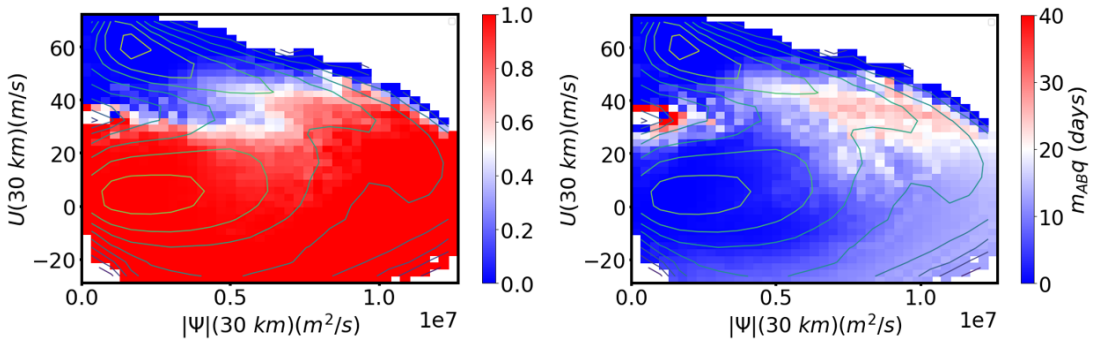


Figure 3.8: Reference statistics for the Holton-Mass model. (left) Committor and (right) lead time computed from a long trajectory and projected onto $U(30\text{ km})$ and $|\Psi|(30\text{ km})$. Colors show reference statistics, and contours show the effective free energy.

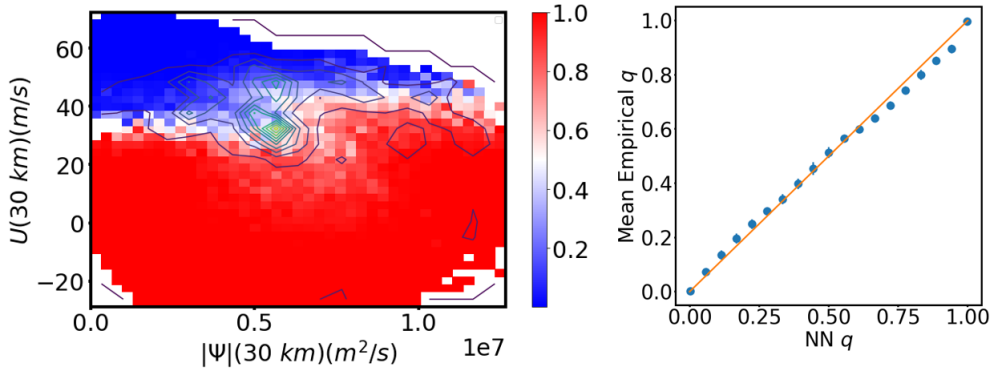


Figure 3.9: Committor for the Holton-Mass model. (left) Colors show predictions, and contour lines show the density of added points from the adaptive sampling scheme described in Section 3.5. (right) Comparison of predicted and reference committors. Symbols show (3.30). Error bars show the standard deviation from networks trained on ten separate data sets resulting from the adaptive sampling scheme.

Because the lead time does not have a fixed range and scales exponentially with the noise, for it, we instead compute the relative error

$$\sqrt{\int_0^{40} (\langle [m_{AB}q](s)\rangle - s)^2/s^2 ds}. \tag{3.32}$$

Figure 3.8 shows the reference committor and lead time projected onto the collective variables.

Figure 3.9(left) shows results for the committor obtained with the adaptive sampling method. As collective variables in the adaptive sampling scheme we use $\xi = (U(30\text{ km}), |\Psi|(30\text{ km}))$. The space between $U(30\text{ km}) = -29$ and 72.5 m/s and between $|\Psi|(30\text{ km}) = 0$ and $1.26 \times 10^7$ m/s$^2$ is partitioned into a $20 \times 20$ grid of bins. We choose this collective variable space because it is physically intuitive, coming directly from the model's state space, and because it resolves SSW events well. Physically, $U$ measures the strength of the vortex while $|\Psi|$ measures the strength of the disruptive wave. Their coupling is key to the nonlinear dynamics of the model. We begin with 50,000 pairs of short trajectories and add 22,000 new pairs of ten-day trajectories every 100 epochs for a total of 10 cycles. Thus the final number of trajectory pairs is 270,000. We take $\lambda = 10$ in (3.13). The network architecture is a fully connected feed forward network with 75 inputs, 10 hidden layers of width 50, with ReLU activation functions, and an output layer with a single sigmoid activation function. We stop training between each addition of data whenever the loss goes below zero (Figure 3.10). Networks for the lead time have the same structure, except that they have a quadratic output layer. The contour lines in Figure 3.9(left) indicate the density produced at the end of the training by the adaptive sampling procedure. The method concentrates new samples in the transition region without being given any information about its location. The method identifies the transition region on the fly.

To validate the results, we trained ten networks on the data set produced by the adaptive sampling method and computed (3.30) (Figure 3.9(right)). The error bars show the standard deviation in $\langle q(s) \rangle$. We see that the training is robust and consistently able to produce good estimates of the committor. We used the data set obtained from the adaptive sampling scheme for the committor to train the neural network to predict the lead time (Figure 3.11). Once again, we find that the method consistently produces good results compared with estimates from a long trajectory. We expect the errors in Figure 3.11 to be larger than those in Figure 3.9 because the estimated committor is used in the loss function for the lead time

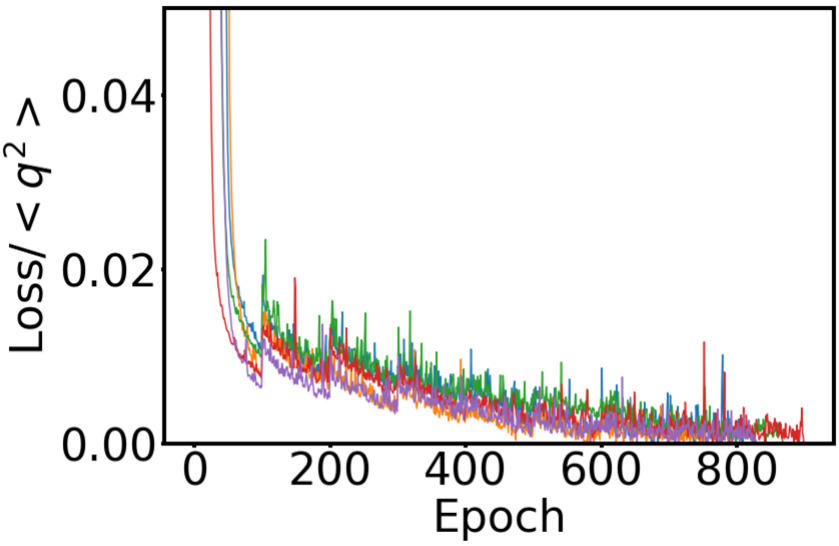


Figure 3.10: The value of the loss as the training progresses for several replicates. We add data adaptively every 100 epochs and halt training when the loss goes below zero. Synchronized spikes in the loss function result from adding data where the loss is high.

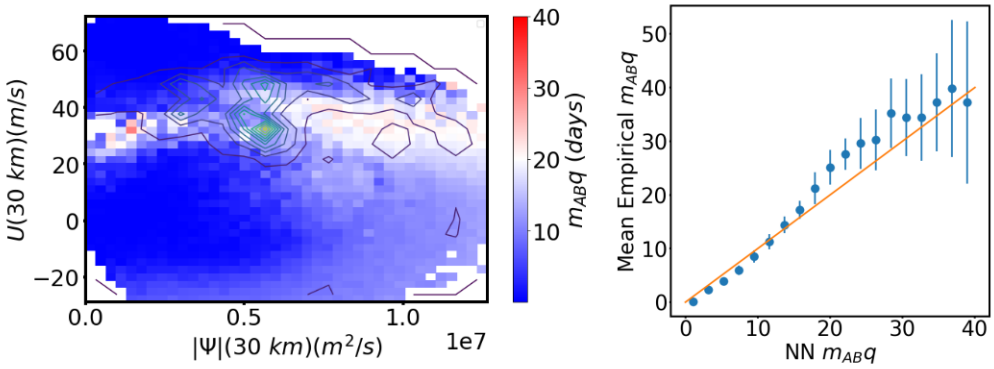


Figure 3.11: Lead time for the Holton-Mass model. (left) Colors show predictions, and contour lines show the density of points in the data set obtained from the adaptive sampling scheme for the committor. (right) Comparison of predicted and reference lead times. Symbols show $\langle [m_{ABq}](s) \rangle$. Error bars show the standard deviation from networks trained on ten indpendent data sets resulting from the adaptive sampling scheme used to train the committor.

(as discussed below (3.7)), allowing errors to compound.

Finally, we determined how the performance of our method depended on key hyperparameters. To elucidate trends, we trained 10 networks on the data set produced by our

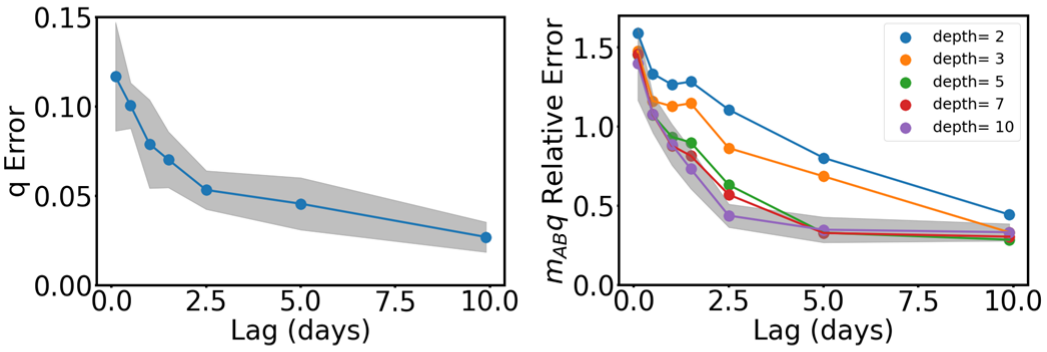


Figure 3.12: Dependence of performance as key hyperparameters are varied. (left) RMS error of the committor as a function of lag time. (right) Relative error in the product used to solve for the lead time as a function of the network depth and lag time. Shading shows the standard deviation from networks trained on ten indpendent data sets resulting from the adaptive sampling scheme used to train the committor; on the right, shading is only shown for the deepest network for clarity.

adaptive sampling method. Figure 3.12 shows the error in our scheme as the lag time is increased. As we observed in the case of the rugged Müller-Brown potential, the error decreases as the lag time increases. We note that as the lag time goes to infinity, all trajectories reach $A \cup B$, and the algorithm reduces to nonlinear regression of point estimates of the conditional expectation of being in state $B$ (see (3.2)). We also investigated the dependence of the performance on the network depth, as shown in the right panel of Figure 3.12. We found that deeper networks were able to achieve low errors at intermediate lag times, although there was relatively little sensitivity to this hyperparameter at short and long lag times.

## 3.7 Conclusions

In this work, we have proposed a machine learning method for solving prediction problems given a data set of short trajectories. By computing conditional expectations that solve Feynman-Kac equations rather than trying to learn the full dynamical law, we reduce the scope of the problem and hence render it more tractable. Our method has a number of

advantages over existing ones:

- it allows computation of any statistic that can be cast in Feynman-Kac form;
- it does not require explicit knowledge of either the model underlying the data or its dynamics (e.g., the form of the generator and its parameters, such as the diffusion tensor);
- it allows for use of arbitrary lag times;
- it allows use of an arbitrary sampling distribution;
- it does not require microscopic reversibility.

We illustrate these advantages using two numerical examples. Using a three-dimensional model for which we can compute an accurate reference solution, we show that our method using short trajectory data is often more robust than related methods that instead use the differential operator form of the Feynman-Kac equation [119, 120, 121, 122]. With the same model, we demonstrate the importance of having data in the low probability regions between metastable states and adequately weighting it against the data in the high probability regions. We propose a simple adaptive sampling scheme that allows us to add data so as to target the largest contributions to the loss during training. Finally, we show that we can compute key statistics for a 75-dimensional model of SSW events (not just the committor but also the lead time) from trajectories that are significantly shorter than the times between events.

Our method opens new possibilities for the study of rare events using experimental and observational data. For example, data sets of short trajectories generated by weather forecasting centers can be analysed by our method to study extreme weather and climate events [115]. However, the requirement that two trajectories be generated from each initial condition poses an obstacle to application of our method to many other data sets. Our next chapter will focus on relaxing this restriction.

# CHAPTER 4

# PREDICTING WITH INEXACT SUBSPACE ITERATION

## 4.1 Introduction

Modern observational, experimental, and computational approaches often yield high-dimensional time series data (trajectories) for complex systems. In principle, these trajectories contain rich information about dynamics and, in particular, the infrequent events that are often most consequential. In practice, however, high-dimensional trajectory data are often difficult to parse for useful insight. The need for more efficient statistical analysis tools for trajectory data is critical, especially when the goal is to understand rare-events that may not be well represented in the data.

We consider dynamics that can be treated as Markov processes. A common starting point for statistical analyses of Markov processes is the transition operator, which describes the evolution of function expectations. The eigenfunctions of the transition operator characterize the most slowly decorrelating features (modes) of the system [150, 23, 26, 151, 77]. These can be used for dimensionality reduction to obtain a qualitative understanding of the dynamics [152, 153], or they can be used as the starting point for further computations [154, 155, 156]. Similarly, prediction functions, which provide information about the likelihood and timing of future events as a function of the current state, are defined through linear equations of the transition operator[35, 156].

A straightforward numerical approach to obtaining these functions is to convert the transition operator to a matrix by projecting onto a finite basis for Galerkin approximation[157, 158, 159, 150, 23, 35, 156, 160]. The performance of such a linear approximation depends on the choice of basis [35, 156, 160], and previous work often resorts to a set of indicator functions on a partition of the state space (resulting in a Markov state model or MSM [159]) for lack of a better choice. While Galerkin approximation has yielded many insights [161, 162], the

limited expressivity of the basis expansion has stimulated interest in (nonlinear) alternatives.

In particular, artificial neural networks can be harnessed to learn eigenfunctions of the transition operator and prediction functions from data [163, 30, 141, 164, 165, 77, 166, 167, 121]. However, existing approaches based on neural networks suffer from various drawbacks. As discussed in Ref.[77], their performance can often be very sensitive to hyperparameters, requiring extensive tuning and varying with random initialization. Many use loss functions that are estimated against the stationary distribution [168, 169, 132, 121, 170, 171], so that metastable states contribute most heavily, which negatively impacts performance [171, 167]. Assumptions about the dynamics (e.g., microscopic reversibility) limit applicability. In Ref. [167] we introduced an approach that overcomes the issues above, but it uses multiple trajectories from each initial condition; this limits the approach to analysis of simulations and moreover requires specially prepared data sets.

The need to compute prediction functions from observed trajectory data also arises in reinforcement learning. There the goal is to optimize an expected future reward (the prediction function) over a policy (a Markov process). For a fixed Markov process, the prediction problem in reinforcement learning is often solved by temporal difference (TD) methods, which allow the use of arbitrary ensembles of trajectories without knowledge of the details of the underlying dynamics[172]. TD methods have a close relationship with an inexact form of power iteration, which, as we describe, can perform poorly on rare-event related problems.

Motivated by this relationship, as well as by an inexact power iteration scheme previously proposed for approximating the stationary probability distribution of a Markov process using trajectory data [173], we propose a computational framework for spectral estimation and rare-event prediction based on inexact iterative numerical linear algebra.

Our framework includes an inexact Richardson iteration for the prediction problem, as well as an extension to inexact subspace iteration for the prediction and spectral estimation problems. The theoretical properties of exact subspace iteration suggest that eigenfunctions

outside the span of the approximation will contribute significantly to the error of our inexact iterative schemes[174]. Consistent with this prediction, we demonstrate that learning additional eigenvalues and eigenfunctions simultaneously through inexact subspace iteration accelerates convergence dramatically relative to inexact Richardson and power iteration in the context of rare events.

While we assume the dynamics can be modeled by a Markov process, we do not require knowledge of their form or a specific underlying model. The method shares a number of further advantages with the approach discussed in Ref. 167 without the need for multiple trajectories from each initial condition in the data set. This opens the door to treating a wide range of observational, experimental, and computational data sets.

The remainder of the paper is organized as follows. In Section 4.1.1, we describe the quantities that we seek to compute in terms of linear operators. In Sections 4.2 and 4.3, we introduce an inexact subspace iteration algorithm that we use to solve for these quantities. Section 4.4 illustrates how the loss function can be tailored to the known properties of the desired quantity. Section 4.5 summarizes the two test systems that we use to illustrate our methods: a two-dimensional potential, for which we can compute accurate reference solutions, and a molecular example that is high-dimensional but still sufficiently tractable that statistics for comparison can be computed from long trajectories. In Section 4.6, we explain the details of the invariant subspace iteration and then demonstrate its application to our two examples. Lastly, Section 4.7 details how the subspace iteration can be modified to compute prediction functions and compares the effect of different loss functions, as well as the convergence properties of power iteration and subspace iteration. We conclude with implications for reinforcement learning.

### *4.1.1 Spectral estimation and the prediction problem*

We have two primary applications in mind in this article. First, we would like to estimate the *dominant eigenfunctions and eigenvalues* of the transition operator $\mathcal{T}^t$ for a Markov process $X^t \in \mathbb{R}^d$, defined as

$$\mathcal{T}^t f(x) = \mathbb{E}_x \left[ f(X^t) \right], \tag{4.1}$$

where $f$ is an arbitrary real-valued function and the subscript $x$ indicates the initial condition $X^0 = x$. The transition operator encodes how expectations of functions evolve in time. The right eigenfunctions of $\mathcal{T}^t$ with the largest eigenvalues characterize the most slowly decorrelating features (modes) of the Markov process [150, 23, 151, 77].

Our second application is to compute *prediction functions* of the general form

$$u(x) = \mathbb{E}_x \left[ \Psi(X^T) + \sum_{t=0}^{T-1} \Gamma(X^t) \right], \tag{4.2}$$

where $T$ is the first time $X^t \notin D$ from some domain $D$, and $\Psi$ and $\Gamma$ are functions associated with the escape event (rewards in reinforcement learning). Prototypical examples of prediction functions that appear in our numerical results are the mean first passage time (MFPT) from each $x$:

$$m(x) = \mathbb{E}_x \left[ T \right] \tag{4.3}$$

and the committor:

$$q(x) = \mathbb{E}_x \left[ \mathbb{1}_B(X^T) \right] = \mathbb{P}_x \left[ X^T \in B \right], \tag{4.4}$$

where $A$ and $B$ are disjoint sets ("reactant" and "product" states) and $D = (A \cup B)^{\mathsf{c}}$. The MFPT is important for estimating rates of transitions between regions of state space, while the committor can serve as a reaction coordinate[175, 176, 170, 177] and as a key ingredient in transition path theory statistics [167, 178, 57]. For any $\tau > 0$, the prediction function

$u(x)$ satisfies the linear equation

$$\left(\mathcal{I} - \mathcal{S}^{\tau}\right) u(x) = \mathbb{E}_x \left[ \sum_{t=0}^{(\tau \wedge T)-1} \Gamma(X^t) \right] \tag{4.5}$$

for $x \in D$, with boundary condition

$$u(x) = \Psi(x) \tag{4.6}$$

for $x \notin D$. In (4.5), $\mathcal{I}$ is the identity operator and

$$\mathcal{S}^{\tau} f(x) = \mathbb{E}_x \left[ f(X^{\tau \wedge T}) \right] \tag{4.7}$$

is the "stopped" transition operator [156]. We use the notation $\tau \wedge T = \min\{\tau, T\}$. The parameter $\tau$ is known as the lag time. While it is an integer corresponding to a number of integration steps, in our numerical examples, we often express it in terms of equivalent time units.

Our specific goal is to solve both the eigenproblem and the prediction problem for $X^t$ in high dimensions and without direct access to a model governing its evolution. Instead, we have access to trajectories of $X^t$ of a fixed, finite duration $\tau$. A natural and generally applicable approach to finding an approximate solution to the prediction problem is to attempt to minimize the residual of (4.5) over parameters $\theta$ of a neural network $u_\theta(x)$ representing $u(x)$. For example, we recently suggested an algorithm that minimizes the residual norm

$$\frac{1}{2} \left\| \left(\mathcal{I} - \mathcal{S}^{\tau}\right) u_\theta - r \right\|_{\mu}^2, \tag{4.8}$$

where $r(x)$ is the right-hand side of (4.5) and $\mu$ is an arbitrary distribution of initial conditions $X^0$ (boundary conditions were enforced by an additional term) [167]. The norm $\|\cdot\|_\mu$ is defined by $\|f\|_\mu^2 = \langle f, f \rangle_\mu$, where $\langle f, g \rangle_\mu = \int f(x) g(x) \mu(dx)$ for arbitrary functions $f$ and $g$.

The gradient of the residual norm in (4.8) with respect to neural-network parameters $\theta$ can be written

$$\left\langle \left(\mathcal{I} - \mathcal{S}^\tau\right) u_\theta - r, \nabla_\theta u_\theta \right\rangle_\mu - \left\langle \left(\mathcal{I} - \mathcal{S}^\tau\right) u_\theta - r, \mathcal{S}^\tau \nabla_\theta u_\theta \right\rangle_\mu \tag{4.9}$$

Given a data set of initial conditions $\{X_j^0\}_{j=1}^n$ drawn from $\mu$ and a single sample trajectory $\{X_j^t\}_{t=0}^\tau$ of $X^t$ for each $X_j^0$, the first term in the gradient (4.9) can be approximated without bias as

$$\left\langle \left(\mathcal{I} - \mathcal{S}^\tau\right) u_\theta - r, \nabla_\theta u_\theta \right\rangle_\mu \approx \frac{1}{n} \sum_{j=1}^{n} \left( u_\theta(X_j^0) - u_\theta(X_j^{\tau \wedge T_j}) - \sum_{t=0}^{(\tau \wedge T_j)-1} \Gamma(X_j^t) \right) \nabla_\theta u_\theta(X_j^0) \tag{4.10}$$

where $T_j$ is the first time $X_j^t \notin D$.

In contrast, unbiased estimation of the second term in (4.9) requires access to two independent trajectories of $X^t$ for each sample initial condition since it is quadratic in $\mathcal{S}^\tau$ [167, 172]. In the reinforcement learning community, TD methods were developed for the purpose of minimizing a loss of a very similar form to (4.8), and they perform a "semigradient" descent by following only the first term in (4.9) [172]. As in the semigradient approximation, the algorithms proposed in this paper only require access to inner products of the form $\langle f, \mathcal{A} g \rangle_\mu$ for an operator $\mathcal{A}$ related to $\mathcal{T}^\tau$ or $\mathcal{S}^\tau$, and they avoid terms that are non-linear in $\mathcal{A}$. As we explain, such inner products can be estimated using trajectory data. However, rather than attempting to minimize the residual directly by an approximate gradient descent, we view the eigenproblem and prediction problems through the lens of iterative numerical linear algebra schemes.

## 4.2 An inexact power iteration

To motivate the iterative numerical linear algebra point of view, observe that the first term in (4.9) is the exact gradient with respect to $\theta'$ of the loss

$$\frac{1}{2}\left\|u_{\theta'} - \mathcal{S}^\tau u_\theta - r\right\|_\mu^2, \tag{4.11}$$

evaluated at $\theta' = \theta$. The result of many steps of gradient descent (later, "inner iterations") on this loss with $\theta$ held fixed can then be viewed as an inexact Richardson iteration for (4.5), resulting in a sequence

$$u_{\theta^{s+1}} \approx \mathcal{S}^\tau u_{\theta^s} + r, \tag{4.12}$$

where, for each $s$, $u_{\theta^s}$ is a parametrized neural-network approximation of the solution to (4.5). To unify our discussion of the prediction and spectral estimation problems, it is helpful to observe that (4.12) can be recast as an equivalent inexact power iteration:

$$\bar{u}_{\theta^{s+1}} \approx \mathcal{A}\bar{u}_{\theta^s} \tag{4.13}$$

where

$$\bar{u}_\theta = \begin{pmatrix} 1 \\ u_\theta \end{pmatrix} \quad \text{and} \quad \mathcal{A} = \begin{bmatrix} 1 & 0 \\ r & \mathcal{S}^\tau \end{bmatrix}. \tag{4.14}$$

Note that $(1, u)^\top$ is the dominant eigenfunction of $\mathcal{A}$ and has eigenvalue equal to 1.

Ref. [173] introduced an inexact power iteration to compute the stationary probability measure of $\mathcal{T}^\tau$, i.e., its dominant left eigenfunction. As those authors note, an inexact power update such as (4.13) can be enforced using a variety of loss functions. In our setting, the $L^2_\mu$ norm in (4.11) can be replaced by any other measure of the difference between $u_{\theta'}$ and $\mathcal{S}^\tau u_\theta + r$, as long as an unbiased estimator of its gradient with respect to $\theta'$ is available. This flexibility is discussed in more detail in Section 4.4, and we exploit it in applications

presented later in this article. For now, we focus instead on another important implication of this viewpoint: the flexibility in the form of the iteration itself.

We will see that when the spectral gap of $\mathcal{A}$, the difference between its largest and second largest eigenvalues, is small, the inexact power iteration (or the equivalent Richardson iteration) described above fails to reach an accurate solution. The largest eigenvalue of $\mathcal{A}$ in (4.14) is 1 and its next largest eigenvalue is the dominant eigenvalue of $\mathcal{S}^\tau$. For rare-event problems the difference between these two eigenvalues is expected to be very small. Indeed, when $X^0$ is drawn from the quasi-stationary distribution of $X^t$ in $D$, the logarithm of the largest eigenvalue of $\mathcal{S}^\tau$ is exactly $-\tau/\,\mathbb{E}[T]$[179]. Thus, when the mean exit time from $D$ is large, we can expect the spectral gap of $\mathcal{A}$ in (4.14) to be very small. In this case, classical convergence results tell us that exact power iteration converges slowly, with the largest contributions to the error coming from the eigenfunctions of $\mathcal{S}^\tau$ with largest magnitude eigenvalues [174]. Iterative schemes that approximate multiple dominant eigenfunctions simultaneously, such as subspace iteration and Krylov methods, can converge much more rapidly[174]. In the next section, we introduce an inexact form of subspace iteration to address this shortcoming. Beyond dramatically improving performance on the prediction problem for rare-events when applied with $\mathcal{A}$ in (4.14), it can also be applied with $\mathcal{A} = \mathcal{T}^\tau$ to compute multiple dominant eigenfunctions and values of $\mathcal{T}^\tau$ itself.

## 4.3 An inexact subspace iteration

Our inexact subspace iteration for the $k$ dominant eigenfunctions/values of $\mathcal{A}$ proceeds as follows. Let $\{\varphi^a_{\theta^s}\}_{a=1}^k$ be a sequence of $k$ basis functions parametrized by $\theta^s$ (these can be scalar or vector valued functions depending on the form of $\mathcal{A}$). We can represent this basis as the vector valued function

$$U_{\theta^s} = \left(\varphi^1_{\theta^s}, \varphi^2_{\theta^s}, \ldots, \varphi^k_{\theta^s}\right). \tag{4.15}$$

Then, we can obtain a new set of $k$ basis functions by approximately applying $\mathcal{A}$ to each of the components of $U_{\theta^s}$:

$$U_{\theta^{s+1}} K^{s+1} \approx \mathcal{A}\, U_{\theta^s} \tag{4.16}$$

where $K^{s+1}$ is an invertible, upper-triangular $k \times k$ matrix that does not change the span of the components of $U_{\theta^{s+1}}$ but is included to facilitate training. One way the approximate application of $\mathcal{A}$ can be accomplished is by minimizing

$$\frac{1}{2} \sum_{a=1}^{k} \left\| \sum_{b=1}^{k} \varphi_\theta^b K_{ba} - \mathcal{A}\, \varphi_{\theta^s}^a \right\|_\mu^2 \tag{4.17}$$

over $\theta$ and $K$ with $\theta^s$ held fixed.

The eigenvalues and eigenfunctions of $\mathcal{A}$ are then approximated by solving the finite-dimensional generalized eigenproblem

$$C^\tau W = C^0 W \Lambda \tag{4.18}$$

where

$$C_{ab}^\tau = \langle \varphi_{\theta^s}^a, \mathcal{A} \varphi_{\theta^s}^b \rangle_\mu \tag{4.19}$$

$$C_{ab}^0 = \langle \varphi_{\theta^s}^a, \varphi_{\theta^s}^b \rangle_\mu, \tag{4.20}$$

each inner product is estimated using trajectory data, and $W$ and $\Lambda$ are $k \times k$ matrices. The matrix $\Lambda$ is diagonal, and the $a$-th eigenvalue $\lambda_a$ of $\mathcal{A}$ is approximated by $\Lambda_{aa}$; the corresponding eigenfunction $v_a$ is approximated by $\sum_{b=1}^k W_{ab}\, \varphi_{\theta^s}^b$.

Even when sampling is not required to estimate the matrices in (4.19) and (4.20), the numerical rank of $C^\tau$ becomes very small as the eigenfunctions become increasingly aligned with the single dominant eigenfunction. To overcome this issue, we apply an orthogonaliza-

tion step between iterations (or every few iterations). Just as the matrices $C^0$ and $C^\tau$ can be estimated using trajectory data, the orthogonalization procedure can also be implemented approximately using data.

Finally, in our experiments we find it advantageous to damp large parameter fluctuations during training by mixing the operator $\mathcal{A}$ with a multiple of the identity, i.e., we perform our inexact subspace iteration on the operator $(1-\alpha_s)\mathcal{I}+\alpha_s\mathcal{A}$ in place of $\mathcal{A}$ itself. This new operator has the same eigenfunctions as $\mathcal{A}$. In our experiments, decreasing the parameter $\alpha_s$ as the number of iterations increases results in better generalization properties of the final solution and helps ensure convergence of the iteration. For our numerical experiments we use

$$\alpha_s = \begin{cases} 1 & s < \sigma \\ 1/\sqrt{s+1-\sigma} & s \geq \sigma \end{cases} \tag{4.21}$$

where $\sigma$ is a user chosen hyperparameter that sets the number of iterations performed before damping begins.

The details, including estimators for all required inner products, in the case of the eigenproblem ($\mathcal{A} = \mathcal{T}^\tau$) are given in Section 4.6 and Algorithm 1. For the prediction problem with $\mathcal{A}$ as in (4.14), they are given in Section 4.7 and Algorithm 2.

## 4.4 Alternative loss functions

As mentioned above, the inexact application of the operator $\mathcal{A}$ can be accomplished by minimizing loss functions other than (4.17). The key requirement for a loss function in the present study is that $\mathcal{A}$ appears in its gradient only through terms of the form $\langle f, \mathcal{A}g\rangle_\mu$ for some functions $f$ and $g$, so that the gradient can be estimated using trajectory data. As a result, we have flexibility in the choice of loss and, in turn, the representation of $u$. In particular, we consider the representation $u_\theta = \zeta(z_\theta)$, where $\zeta$ is an increasing function, and $z_\theta$ is a function parameterized by a neural network. An advantage of doing so is that

the function $\zeta$ can restrict the output values of $u_\theta$ to some range. For example, when computing a probability such as the committor, a natural choice is the sigmoid function $\zeta(x) = (1 + e^{-x})^{-1}$.

Our goal is to train a sequence of parameter values so that $u_{\theta^s}$ approximately follows a subspace iteration, i.e., so that $\zeta(z_{\theta^{s+1}}) \approx \mathcal{A}u_{\theta^s}$. To this end, we minimize with respect to $\theta$ the loss function

$$\mathbb{E}_{X^0 \sim \mu}\left[V(z_\theta) - z_\theta \mathcal{A} u_{\theta^s}\right], \tag{4.22}$$

where $V$ is an antiderivative of $\zeta$, and $\theta^s$ is fixed. The subscript $X^0 \sim \mu$ in this expression indicates that $X^0$ is drawn from $\mu$. Note that, as desired, $\mathcal{A}$ appears in the gradient of (4.22) only in an inner product of the required form, and an approximate minimizer, $\theta^{s+1}$, of this loss satisfies $\zeta(z_{\theta^{s+1}}) \approx \mathcal{A}u_{\theta^s}$. This general form of loss function is adapted from variational expressions for the divergence of two probability measures [180, 173].

The $L^2_\mu$ loss in (4.17), which we use in several of our numerical experiments, corresponds to the choice $\zeta(x) = x$ and $V(x) = x^2/2$. The choice of $\zeta(x) = (1 + e^{-x})^{-1}$ mentioned above corresponds to $V(x) = \log(1 + e^x)$; we refer to the loss in (4.22) with this choice of $V$ as the "softplus" loss. We note that in the context of committor function estimation, in the limit of infinite lag time the "softplus" loss corresponds directly to the maximum log-likelihood loss (for independent Bernoulli random variables) that defines the logistic regression estimate of the probability of reaching $B$ before $A$. Logistic regression has previously been used in conjunction with data sets of trajectories integrated all the way until reaching $A$ or $B$ to estimate the committor function[181, 182, 183, 184, 185, 186].

## 4.5 Test problems

We illustrate our methods with two well-characterized systems that exemplify features of molecular transitions. In this section, we provide key details of these systems.

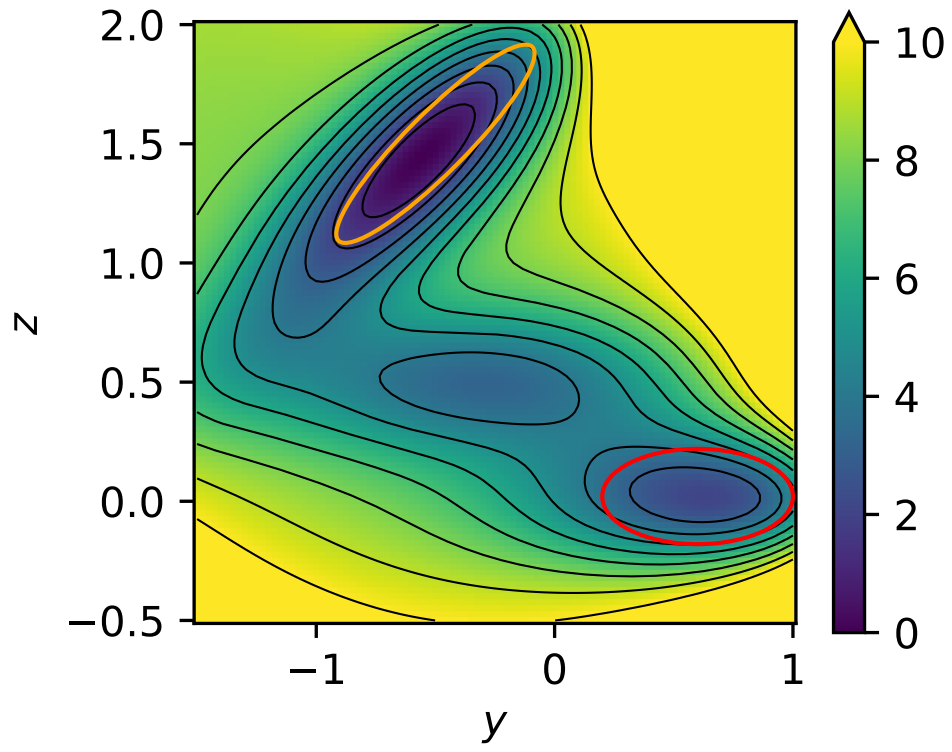


Figure 4.1: Müller-Brown potential energy surface. The orange and red ovals indicate the locations of states $A$ and $B$ respectively when computing predictions. Contour lines are drawn every 1 $\beta^{-1}$.

### *4.5.1 Müller-Brown potential*

The first system is defined by the Müller-Brown potential [2] (Figure 4.1):

$$V_{\mathrm{MB}}(y,z) = \frac{1}{20}\sum_{i=1}^{4} C_i \exp[a_i(y-y_i)^2 + b_i(y-y_i)(z-z_i) + c_i(z-z_i)^2]. \quad (4.23)$$

The two-dimensional nature of this model facilitates visualization. The presence of multiple minima and saddlepoints that are connected by a path that does not align with the coordinate axes makes the system challenging for both sampling and analysis methods. In Sections 4.6.1 and 4.7.1, we use $C_i = \{-200, -100, -170, 15\}$, $a_i = \{-1, -1, -6.5, 0.7\}$, $b_i =$

$\{0, 0, 11, 0.6\}$, $c_i = \{-10, -10, -6.5, 0.7\}$, $y_i = \{1, -0.27, -0.5, -1\}$, $z_i = \{0, 0.5, 1.5, 1\}$. In Section 4.7.2, we tune the parameters to make transitions between minima rarer; there, the parameters are $C_i = \{-250, -150, -170, 15\}$, $a_i = \{-1, -3, -6.5, 0.7\}$, $b_i = \{0, 0, 11, 0.6\}$, $c_i = \{-10, -30, -6.5, 0.7\}$, $y_i = \{1, -0.29, -0.5, -1\}$, $z_i = \{0, 0.5, 1.5, 1\}$. For prediction, we analyze transitions between the upper left minimum ($A$) and the lower right minimum ($B$) in Figure 4.1; these states are defined as

$$\begin{aligned} A &= \{y, z : 6.5(y + 0.5)^2 - 11(y + 0.5)(z - 1.5) + 6.5(z - 1.5)^2 < 0.3\} \\ B &= \{y, z : (y - 0.6)^2 + 0.5(z - 0.02)^2 < 0.2\}. \end{aligned} \tag{4.24}$$

To generate a data set, we randomly draw 50,000 initial conditions $X_j^0$ uniformly from the region

$$\Omega = \{y, z : -1.5 < y < 1.0,\ -0.5 < z < 2.5,\ V_{\mathrm{MB}}(y, z) < 12\} \tag{4.25}$$

and, from each of these initial conditions, generate one trajectory according to the discretized overdamped Langevin dynamics

$$X_j^t = X_j^{t-1} - \delta_t \nabla V_{\mathrm{MB}}(X_j^{t-1}) + \sqrt{\delta_t \, 2\beta^{-1}} \, \xi_j^t \tag{4.26}$$

where $1 \leq t \leq \tau$, the $\xi_j^t$ are independent standard Gaussian random variables, and the timestep is $\delta_t = 0.001$. We use an inverse temperature of $\beta = 2$, and we vary $\tau$ as indicated below (note that $\tau$ is an integer number of steps, but we present our results for the Müller-Brown model in terms of the dimensionless scale used for $\delta_t$ immediately above). For each test, we use independent random numbers and redraw the initial conditions because it is faster to generate the trajectories than to read them from disk in this case. All error bars are computed from the empirical standard deviation over 10 replicate data sets.

To validate our results, we compare the neural-network results against grid-based references, computed as described in Section S4 of the Supplementary Material of Ref. [35]

and the Appendix of Ref. [187] (our notation here follows the former more closely). The essential idea is that the terms in the infinitesimal generator of the transition operator can be approximated on a grid to second order in the spacing by expanding both the probability for transitions between sites and a test function. To this end, we define the transition matrix for neighboring sites $x = (y, z)$ and $x' = (y \pm \delta_x, z)$ or $(y, z \pm \delta_x)$ on a grid with spacings $\delta_x$:

$$P(x'|x) = \frac{1}{1 + e^{\beta[V(x') - V(x)]}} \tag{4.27}$$

(this definition differs from that in Ref. 35 by a factor of $1/4$, and we scale $P - I$, where $I$ is the identity matrix, accordingly to set the time units below). The diagonal entry $P(x|x)$ is fixed to make the transition matrix row-stochastic. We use $\delta_x = 0.0125$; the grid is a rectangle with $y \in [-1.5, 1.0]$, and $z \in [-0.5, 2.0]$. The reference invariant subspaces are computed by diagonalizing the matrix $2(P - I)/\beta\delta_x^2$ with a sparse eigensolver; we use scipy.sparse.linalg. We obtain the reference committor $\hat{q}_+$ for grid sites in $(A \cup B)^{\mathsf{c}}$ by solving

$$\begin{aligned} &\mathrm{diag}(\hat{\mathbb{1}}_{(A\cup B)^{\mathsf{c}}})(P - I)\hat{q}_+ \\ &\qquad = -\mathrm{diag}(\hat{\mathbb{1}}_{(A\cup B)^{\mathsf{c}}})(P - I)\hat{\mathbb{1}}_B \end{aligned} \tag{4.28}$$

and for grid sites in $A \cup B$ by setting $\hat{q}_+ = \hat{\mathbb{1}}_B$. Here, $\hat{q}_+$ and $\hat{\mathbb{1}}_B$ are vectors corresponding to the functions evaluated at the grid sites.

### *4.5.2 $AIB_9$ helix-to-helix transition*

The second system is a peptide of nine $\alpha$-aminoisobutyric acids ($AIB_9$; Figure 4.2). Because AIB is achiral around its $\alpha$-carbon atom, $AIB_9$ can form left- and right-handed helices with equal probabilities, and we study the transitions between these two states. This transition was previously studied using MSMs and long unbiased molecular dynamics simulations [188, 189]. $AIB_9$ poses a stringent test due to the presence of many metastable intermediate states.

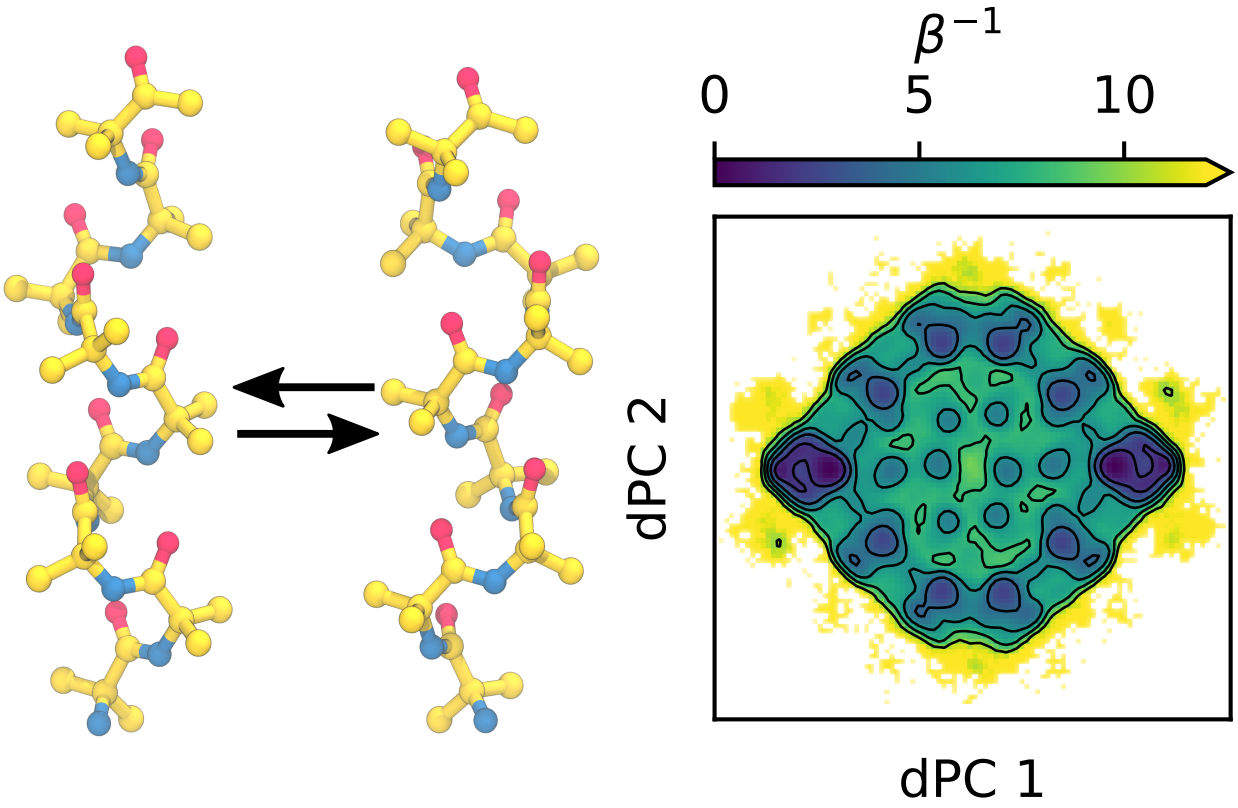


Figure 4.2: Helix-to-helix transition of $AIB_9$. (left) Left- and right-handed helices, which we use as states $A$ and $B$, respectively, when computing predictions. Carbon, nitrogen, and oxygen atoms are shown in yellow, blue, and red, respectively; hydrogen atoms are omitted. (right) Potential of mean force constructed from the histogram of value pairs of the first two dihedral angle principal components; data are from the 20 trajectories of 5 $\mu$s that we use to construct reference statistics (see text). The left-handed helix corresponds to the left-most basin, and the right-handed helix corresponds to the right-most basin. Contour lines are drawn every 2 $\beta^{-1}$ corresponding to a temperature of 300 K.

The states are defined in terms of the internal $\phi$ and $\psi$ dihedral angles. We classify an amino acid as being in the "left" state if its dihedral angle values are within a circle of radius $25^\circ$ centered at $(41^\circ, 43^\circ)$, that is

$$(\phi - 41^\circ)^2 + (\psi - 43^\circ)^2 \leq (25^\circ)^2.$$

Amino acids are classified as being in the "right" state using the same radius, but centered instead at $(-41^\circ, -43^\circ)$. States $A$ and $B$ are defined by the amino acids at sequence positions

3–7 being all left or all right, respectively. We do not use the two residues on each end of $AIB_9$ in defining the states as these are typically more flexible [189]. The states can be resolved by projecting onto dihedral angle principal components (dPCs; Figure 4.2, right) as described previously [190].

Following a procedure similar to that described in Ref. 189, we generate a data set of short trajectories. From each of the 691 starting configurations in Ref. 189, we simulate 10 trajectories of duration 20 ns with initial velocities drawn randomly from a Maxwell-Boltzmann distribution at a temperature of 300 K. The short trajectory data set thus contains 6,910 trajectories, corresponding to a total sampling time of 138.2 $\mu$s. We use a timestep of 4 fs together with a hydrogen mass repartitioning scheme [191], and configurations are saved every 40 ps. We employ the AIB parameters from Forcefield_NCAA [192] and the GB-Neck2 implicit-solvent model [193]. Simulations are performed with the Langevin integrator in OpenMM 7.7.0 [194] using a friction coefficient of $1\,\mathrm{ps}^{-1}$. To generate a reference for comparison to our results, we randomly select 20 configurations from the data set above and, from each, run a simulation of 5 $\mu$s with the same simulation parameters. For all following tests on $AIB_9$, batches consist of pairs of frames separated by $\tau$ drawn randomly with replacement from the short trajectories (i.e., from all possible such pairs in the data set). From each frame, we use only the atom positions because the momenta decorrelate within a few picoseconds, which is much shorter than the lag times that we consider. However, in principle, the momenta could impact the prediction functions [195] and be used as neural-network inputs as well.

## 4.6 Spectral estimation

In this section, we provide some further numerical details for the application of our method to spectral estimation and demonstrate the method on the test problems. For our subspace iteration with $\mathcal{A} = \mathcal{T}^\tau$, we require estimators for inner products of the form $\langle f, \mathcal{T}^\tau g\rangle_\mu$. For

example, the gradient of the loss function (4.17) involves inner products of the form

$$\left\langle \nabla_\theta \varphi_\theta^a, \mathcal{T}^\tau \varphi_\theta^b \right\rangle_\mu . \tag{4.29}$$

For these, we use the unbiased data-driven estimator

$$\langle f, \mathcal{T}^\tau g \rangle_\mu \approx \frac{1}{n} \sum_{j=1}^{n} f(X_j^0) g(X_j^\tau). \tag{4.30}$$

As discussed in Section 4.3, applying the operator $\mathcal{T}^\tau$ repeatedly causes each basis function to converge to the dominant eigenfunction and leads to numerical instabilities. To avoid this, we orthogonalize the outputs of the networks with a QR decomposition at the end of each subspace iteration by constructing the matrix $\Phi_{ia} = \varphi_\theta^a(X_i^0)$ and then computing the factorization $\Phi = QR$ where $Q$ is an $n \times k$ matrix with orthogonal columns and $R$ is an upper triangular $k \times k$ matrix. Finally, we set $\tilde{\varphi}_s^a = \sum_{b=1}^k \varphi_\theta^b \, (R^{-1}N)_{ba}$, where $N$ is a diagonal matrix with entries equal to the norms of the columns of $\Phi$ (before orthogonalization). To ensure that the networks remain well-separated (i.e., the eigenvalues of $C^0$ remain away from zero), we penalize large off-diagonal entries of $K$ by adding to the loss

$$\gamma_1 \| K - \mathrm{diag}(K) \|_{\mathrm{F}}^2, \tag{4.31}$$

where $\gamma_1$ allows us to tune the strength of this term relative to others, and $\|\cdot\|_{\mathrm{F}}$ is the Frobenius norm. We control the scale of each network output using the strategy from Ref. 173; that is, we add to the loss a term of the form

$$\gamma_2 \sum_{a=1}^{k} \left[ 2\nu_a \left( \frac{1}{n} \sum_{j=1}^{n} \varphi_\theta^a(X_j^0)^2 - 1 \right) - \nu_a^2 \right], \tag{4.32}$$

where we have introduced the conjugate variables $\nu_a$ which we maximize with gradient ascent

(or similar optimization). In general, our numerical experiments suggest that it is best to keep $\gamma_1$ and $\gamma_2$ relatively small. We find that stability of the algorithm over many subspace iterations is improved if the matrix $K$ is set at its optimal value before each inner loop. To do this, we set

$$K_{1:i,i} = \arg\min_{c} \frac{1}{n} \sum_{j=1}^{n} \left( \sum_{a=1}^{i} \varphi_\theta^a(X_j^0) c_a - \tilde{\varphi}_s^a(X_j^\tau) \right)^2 + \gamma_2 \sum_{a=1}^{i-1} c_a^2. \tag{4.33}$$

The above minimization can be solved with linear least squares. Finally, we note that in practice any optimizer can be used for the inner iteration steps, though the algorithm below implements stochastic gradient descent. In this work, we use Adam[196] for all numerical tests. We summarize our procedure for spectral estimation in Algorithm 1.

### *4.6.1 Müller-Brown model*

Table 4.1: Parameter choices used in this work

| | **Spectral Estimation** | | **Committor** | | | **MFPT** |
|---|---|---|---|---|---|---|
| **Hyperparameter** | Müller-Brown | $AIB_9$ | Müller-Brown | Modified Müller-Brown | $AIB_9$ | $AIB_9$ |
| subspace dimension $k$ | 3 | 5 | 1 | 2, 1[1] | 1 | 5 |
| input dimensionality | 2 | 174 | 2 | 2 | 174 | 174 |
| hidden layers | 6 | 6 | 6 | 6 | 6 | 6 |
| hidden layer width | 64 | 128 | 64 | 64 | 150 | 150 |
| hidden layer nonlinearity | CeLU | CeLU | ReLU | ReLU | ReLU | ReLU |
| output layer nonlinearity | none | tanh | sigmoid/none | none | none | none |
| outer iterations $S$ | 10 | 100 | 100 | 4 + 10[1] | 100 | 300 |
| inner iterations $M$ | 5000 | 2000 | 200 | 5000 | 2000 | 1000 |
| $\sigma$ | 2 | 50 | 50 | 0 | 50 | 0 |
| batch size $B$ | 2000 | 1024 | 5000 | 2000 | 1024 | 2000 |
| learning rate $\eta$ | 0.001 | 0.0001 | 0.001 | 0.001 | 0.001 | 0.001 |
| $\gamma_1$ | 0.15 | 0.001 | - | - | - | 0.1 |
| $\gamma_2$ | 0.01 | 0.01 | - | - | - | 10 |
| loss for $\varphi_\theta^1$ | $L_\mu^2$ | $L_\mu^2$ | $L_\mu^2$/softplus | softplus | softplus | $L_\mu^2$ |
| loss for $\varphi_\theta^a$ for $a > 1$ | $L_\mu^2$ | $L_\mu^2$ | - | $L_\mu^2$ | - | $L_\mu^2$ |

As a first test of our method, we compute the $k = 3$ dominant eigenpairs for the Müller-Brown model. Since we know that the dominant eigenfunction of the transition operator is the constant function $v_1(y, z) = 1$ with eigenvalue $\lambda_1 = 1$, we directly include this function

**Algorithm 1** Inexact subspace iteration (with $L^2_\mu$ loss) for spectral estimation

**Require:** Subspace dimension $k$, transition data $\{X_j^0, X_j^\tau\}_{j=1}^n$, batch size $B$, learning rate $\eta$, number of subspace iterations $S$, number of inner iterations $M$, regularization parameters $\gamma_1$ and $\gamma_2$

Initialize $\{\varphi_\theta^a\}_{a=1}^k$ and $\{\tilde{\varphi}_1^a\}_{a=1}^k$
**for** $s = 1 \dots S$ **do**
  **for** $m = 1 \dots M$ **do**
    Sample a batch of data $\{X_j^0, X_j^\tau\}_{j=1}^B$
    $\hat{\mathcal{L}}_1 \leftarrow \frac{1}{B}\sum_{j=1}^B \sum_{a=1}^k \left[\frac{1}{2}(\sum_{b=1}^k \varphi_\theta^b(X_j^0)K_{ba})^2 - \sum_{b=1}^k \varphi_\theta^b(X_j^0)K_{ba}\left\{\alpha_s \tilde{\varphi}_s^a(X_j^\tau) + (1-\alpha_s)\tilde{\varphi}_s^a(X_j^0)\right\}\right]$
    $\hat{\mathcal{L}}_K \leftarrow \gamma_1 \|K - \mathrm{diag}(K)\|_{\mathrm{F}}^2$
    $\hat{\mathcal{L}}_{\mathrm{norm}} \leftarrow \gamma_2 \sum_{a=1}^k (2\nu_a(\frac{1}{B}\sum_{j=1}^B(\varphi_\theta^a(X_j^0)^2) - 1) - \nu_a^2)$
    $\hat{\mathcal{L}} \leftarrow \hat{\mathcal{L}}_1 + \hat{\mathcal{L}}_K + \hat{\mathcal{L}}_{\mathrm{norm}}$
    $\theta \leftarrow \theta - \eta \nabla_\theta \hat{\mathcal{L}}$
    $K \leftarrow K - \eta\, \mathrm{triu}(\nabla_K \hat{\mathcal{L}})$
    $\nu_a \leftarrow \nu_a + \eta \nabla_{\nu_a} \hat{\mathcal{L}}$
  **end for**
  Compute the matrix $\Phi_{ia} = \varphi_\theta^a(X_i^0)$ ▷ $\Phi \in \mathbb{R}^{n\times k}$
  Compute diagonal matrix $N_{aa}^2 = \sum_i \varphi_\theta^a(X_i^0)^2$
  Compute QR-decomposition $\Phi = QR$ ▷ $Q \in \mathbb{R}^{n\times k}$ and $R \in \mathbb{R}^{k\times k}$
  $\tilde{\varphi}_s^a \leftarrow \sum_{b=1}^k \varphi_\theta^b\,(R^{-1}N)_{ba}$
  $K_{1:i,i} \leftarrow \arg\min_c \frac{1}{n}\sum_{j=1}^n \left(\sum_{a=1}^i \varphi_\theta^a(X_j^0)c_a - \tilde{\varphi}_s^a(X_j^\tau)\right)^2 + \gamma_2 \sum_{a=1}^{i-1} c_a^2$
**end for**
Compute the matrices $C_{ab}^t = \frac{1}{n}\sum_{j=1}^n \tilde{\varphi}_s^a(X_j^0)\tilde{\varphi}_s^b(X_j^t)$ for $t = 0, \tau$ ▷ $C^t \in \mathbb{R}^{k\times k}$
Solve the generalized eigenproblem $C^\tau W = C^0 W \Lambda$ for $W$ and $\Lambda$
**return** eigenvalues $\Lambda$, eigenfunctions $v_a = \sum_{b=1}^k W_{ab}\tilde{\varphi}_s^b$

in the basis as a non-trainable function, i.e. $\varphi_\theta^1(y,z) = 1$. To initialize $\tilde{\varphi}_1^a$ for each $a > 1$, we choose a standard Gaussian vector $(Y^a, Z^a) \in \mathbb{R}^2$, and set $\tilde{\varphi}_1^a(y,z) = y\,Y^a + z\,Z^a$. This ensures that the initial basis vectors are well-separated and the first QR step is numerically stable. Here and in all subsequent Müller-Brown tests, batches of trajectories are drawn from the entire data set with replacement. Other hyperparameters are listed in Table 4.1.

Figure 4.3 shows that we obtain good agreement between the estimate produced by the inexact subspace iteration in Algorithm 1 and reference eigenfunctions. Figure 4.4 (upper

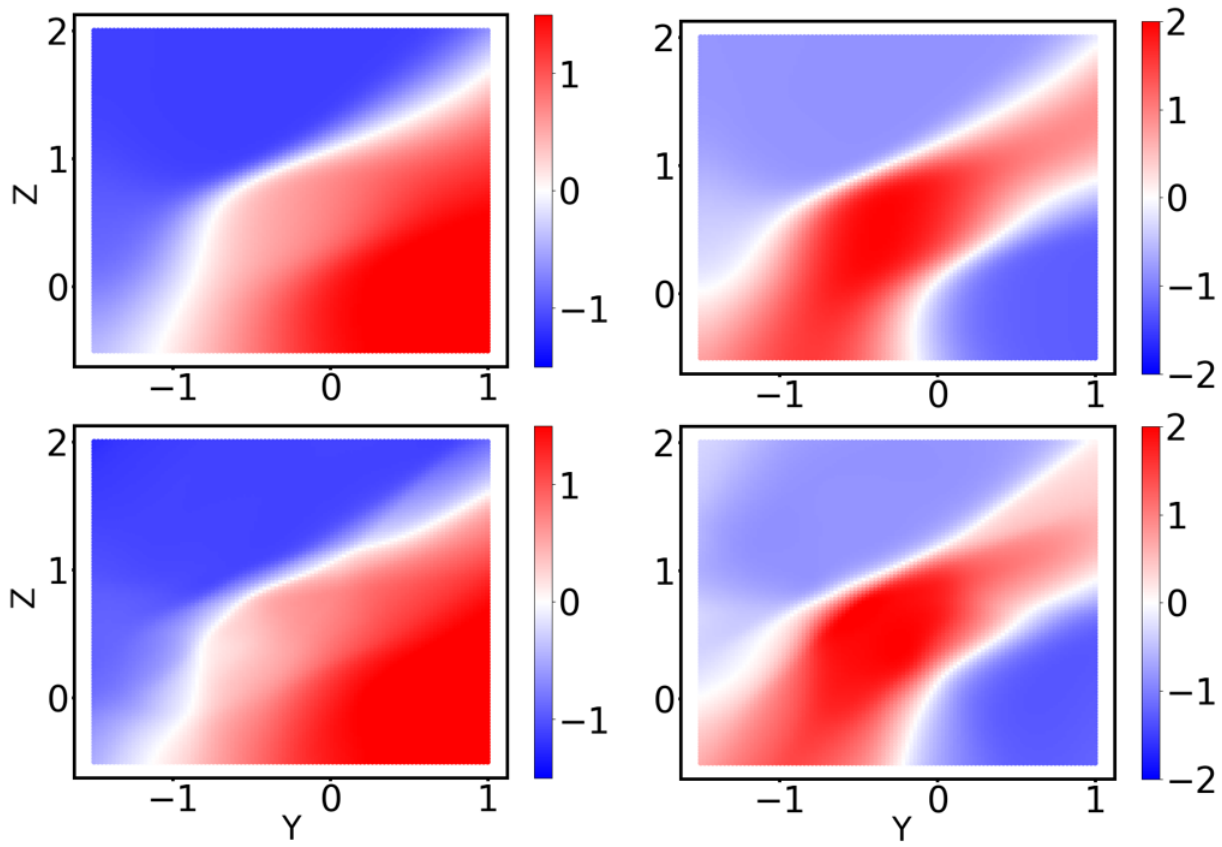


Figure 4.3: First two non-trivial eigenfunctions of the Müller-Brown model. (top) Grid-based reference. (bottom) Neural network subspace after ten subspace iteration steps, computed with $\tau = 300$ steps (i.e., $0.3\,\delta_t^{-1}$).

panels) shows how the corresponding eigenvalues vary with lag time; again there is good agreement with the reference. Furthermore, there is a significant gap between $\lambda_3$ and $\lambda_4$, indicating that a three-dimensional subspace captures the dynamics of interest for this system.

We compare the subspace that we obtain from our method with that from an MSM constructed from the same amount of data by using $k$-means to cluster the configurations into 400 states and counting the transitions between clusters. This is a very fine discretization for this system, and the MSM is sufficiently expressive to yield eigenfunctions in good agreement with the reference. The relative error of $1-\lambda_2$ is comparable for the two methods (Figure 4.4,

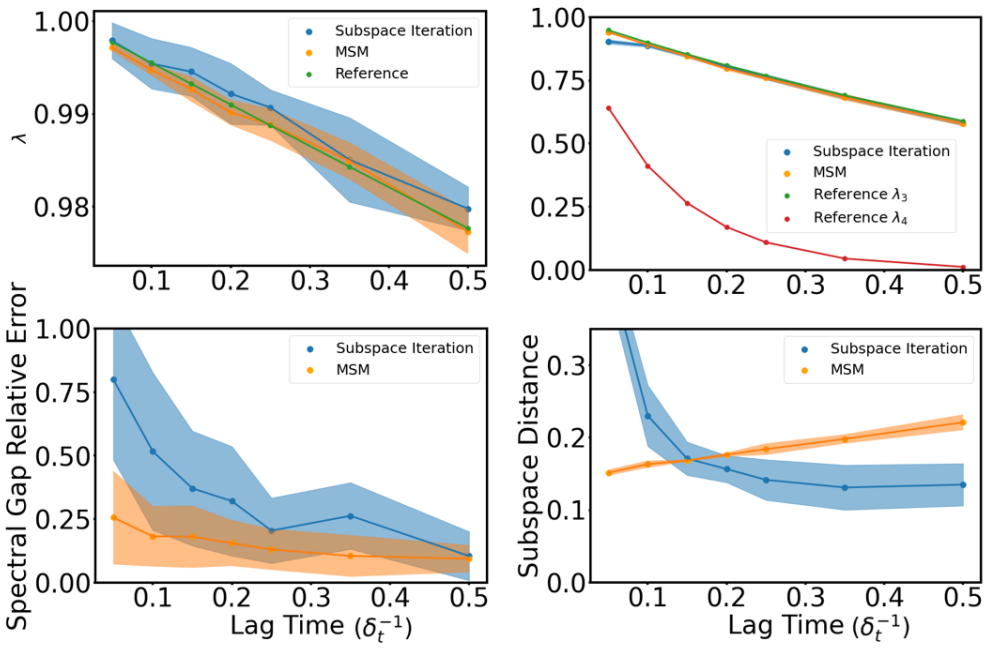


Figure 4.4: Spectral estimation as a function of lag time (in units of $\delta_t^{-1}$) for the Müller-Brown model. (top left) Second eigenvalue. (top right) Third and fourth eigenvalues; only the reference fourth eigenvalue is shown to illustrate the spectral gap. (bottom left) Relative error in the first spectral gap (i.e., $1 - \lambda_2$). (bottom right) Subspace distance between estimated and reference three-dimensional invariant subspaces.

lower left). To compare two finite dimensional subspaces, $\mathcal{U}$ and $\mathcal{V}$, we define the subspace distance as[151]

$$d(\mathcal{U}, \mathcal{V}) = \left\| (I - P_{\mathcal{U}}) P_{\mathcal{V}} \right\|_{\mathrm{F}}, \tag{4.34}$$

where $P_{\mathcal{U}}$ and $P_{\mathcal{V}}$ denote the orthogonal projections onto $\mathcal{U}$ and $\mathcal{V}$, respectively, and $\|\cdot\|_{\mathrm{F}}$ is the Frobenius norm. Figure 4.4 (lower right) shows the subspace distances from the reference as functions of lag time. We see that the inexact subspace iteration better approximates the three-dimensional dominant eigenspace for moderate to long lag times, even though the eigenvalues are comparable.

### 4.6.2 $AIB_9$

For the molecular test system, we compute the dominant five-dimensional subspace. We compare the inexact subspace iteration in Algorithm 1 with MSMs constructed on dihedral angles ("dihedral MSM") and on Cartesian coordinates ("Cartesian MSM"). We expect the dihedral MSM to be accurate given that the dynamics of $AIB_9$ are well-described by the backbone dihedral angles [188, 189], and we thus use it as a reference. It is constructed by clustering the sine and cosine of each of the backbone dihedral angles ($\phi$ and $\psi$) for the nine residues (for a total of $2 \times 2 \times 9 = 36$ input features) into 1000 clusters using $k$-means and counting transitions between clusters. The Cartesian MSM is constructed by similarly counting transitions between 1000 clusters from the $k$-means algorithm, but the clusters are based on the Cartesian coordinates of all non-hydrogen atoms after aligning the backbone atoms of the trajectories, for a total of 174 input features. Because of the difficulty of clustering high-dimensional data, we expect the Cartesian MSM basis to give poor results. The neural network for the inexact subspace iteration receives the same 174 Cartesian coordinates as input features. We choose to use Cartesian coordinates rather than dihedral angles as inputs because it requires the network to identify nontrivial representations for describing the dynamics.

As in the Müller-Brown example, we use $\varphi_\theta^1 = 1$ and a random linear combination of coordinate functions to initialize $\tilde{\varphi}_1^a$ for $a > 1$. Other hyperparameters are listed in Table 4.1. With these choices, the neural-network training typically requires about 20 minutes on a single NVIDIA A40 GPU; this is much longer than the time required for diagonalization of the $1000 \times 1000$ MSM transition matrix, which is nearly instantaneous. However, the time for neural-network training is negligible compared with the time to generate the data set, which is the same for both approaches.

Taking the dihedral MSM as a reference, the Cartesian MSM systematically underestimates the eigenvalues (Figure 4.5). The subspace iteration is very accurate for the first four

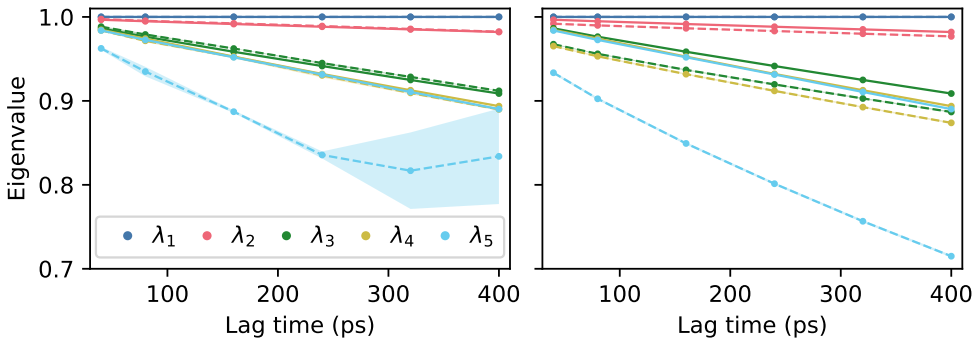


Figure 4.5: First five eigenvalues of the transition operator for $AIB_9$ as a function of lag time. (left) Comparison between eigenvalues computed using the dihedral MSM with 1000 clusters (solid lines) and the inexact subspace iteration (dashed lines). The shading indicates standard deviations over five trained networks for the subspace iteration. (right) Comparison between a dihedral MSM (solid lines) and Cartesian MSMs with 1000 clusters (dashed lines). The standard deviations for the Cartesian MSMs over five random seeds for $k$-means clustering are too narrow to be seen.

eigenvalues but the estimates for the fifth are low and vary considerably from run to run. A very small gap between $\lambda_4$ and $\lambda_5$ may contribute to the difficulty in estimating $\lambda_5$. In Figure 4.6, we plot the first two non-trivial eigenfunctions ($v_2$ and $v_3$), which align with the axes of the dPC projection. The eigenfunction $v_2$ corresponds to the transition between the left- and right-handed helices; the eigenfunction $v_3$ is nearly orthogonal to $v_2$ and corresponds to transitions between intermediate states. It is challenging to visualize the remaining two eigenfunctions by projecting onto the first two dPCs because $v_4$ and $v_5$ are orthogonal to $v_2$ and $v_3$. The estimates for $v_2$ are in qualitative agreement for all lag times tested (Figure 4.6 shows results for $\tau$ corresponding to 40 ps), but the subspace iteration results are less noisy for the shortest lag times. Moreover, the estimate for $v_3$ from subspace iteration agrees more closely with that from the dihedral MSM than does the estimate for $v_3$ from the Cartesian MSM. The subspace distance for $v_2$ and $v_3$ between the subspace iteration and the dihedral MSM is 0.947, compared with a value of 0.969 for the subspace distance between the two MSMs. Together, our results indicate that the neural networks are able to learn the leading eigenfunctions and eigenvalues of the transition operator (dynamical modes) of this system

despite being presented with coordinates that are not the natural ones for describing the dynamics.

## 4.7 Prediction

Inexact subspace iteration for $\mathcal{A}$ in (4.14) is equivalent to performing the inexact Richardson iteration in (4.12) on the first basis function $\varphi_\theta^1$ and then performing an inexact subspace iteration for the operator $\mathcal{S}^\tau$ on the rest of the basis functions. The iteration requires unbiased estimators of the forms

$$\langle f, \mathcal{S}^\tau g \rangle_\mu \approx \frac{1}{n} \sum_{j=1}^{n} f(X_j^0) g(X_j^{\tau \wedge T_j}) \tag{4.35}$$

and

$$\langle f, r \rangle_\mu \approx \frac{1}{n} \sum_{j=1}^{n} f(X_j^0) \sum_{t=0}^{(\tau \wedge T_j)-1} \Gamma(X_j^t), \tag{4.36}$$

where $T_j$ is the first time $X_j^t$ enters $D^{\mathsf{c}}$ and $r$ is the right-hand side of (4.5), as previously.

The Richardson iterate, $\varphi_\theta^1$, must satisfy the boundary condition $\varphi_\theta^1(x) = \Psi(x)$ for $x \notin D$. The other basis functions should satisfy $\varphi_\theta^a(x) = 0$ for $x \notin D$. In practice, we enforce the boundary conditions by explicitly setting $\varphi_\theta^1(x) = \Psi(x)$ and $\varphi_\theta^a(x) = 0$ for $a > 1$ when $x \notin D$.

When the boundary condition is zero, as for the MFPT, we find an approximate solution of the form

$$u_\theta = \sum_{a=1}^{k} w_a \varphi_\theta^a \tag{4.37}$$

by solving the $k$-dimensional linear system

$$\left( C^0 - C^\tau \right) w = p \tag{4.38}$$

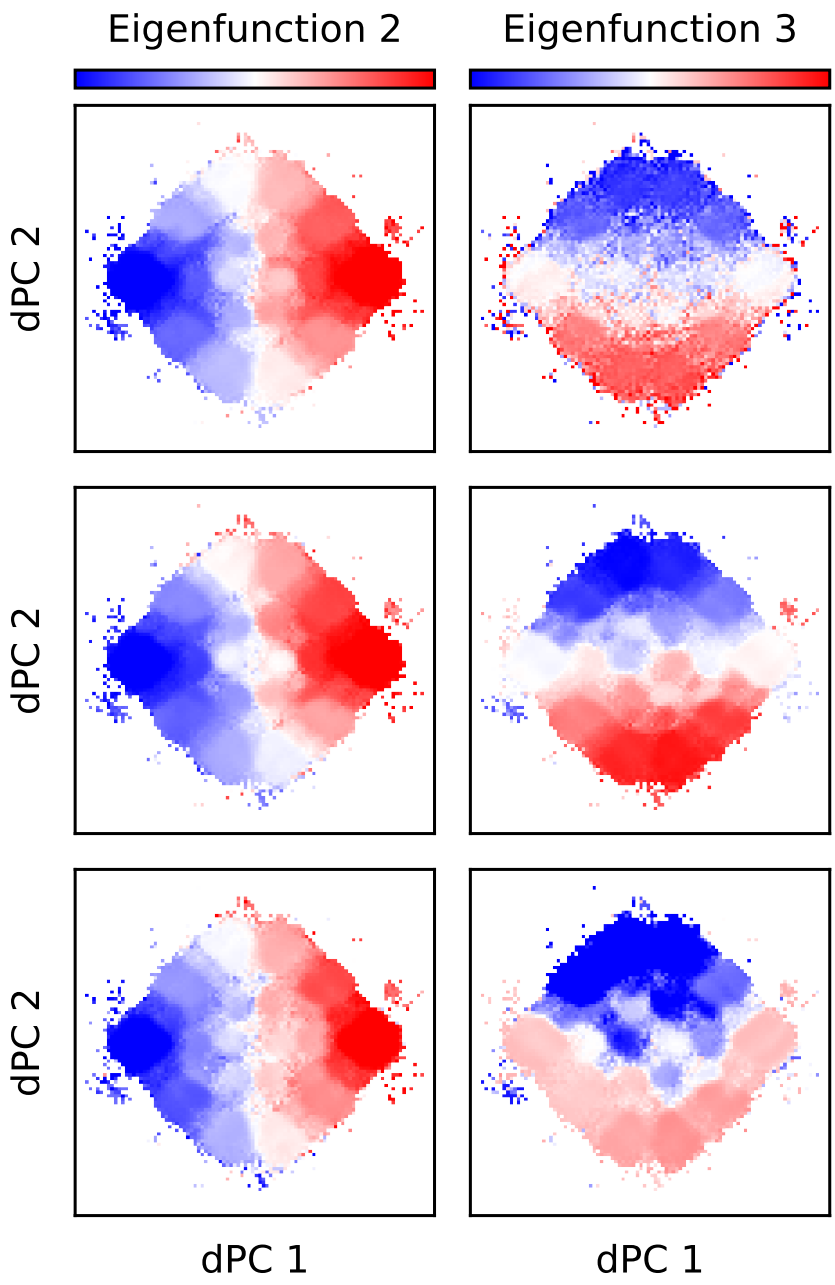


Figure 4.6: First two non-trivial eigenfunctions of $AIB_9$ projected onto the first two dPCs (i.e., averaged for bins in the two-dimensional space shown). (top) MSM constructed on sine and cosine of dihedral angles with 1000 clusters and lag time corresponding to 40 ps. (middle) Inexact subspace iteration using Cartesian coordinates and the same lag time. (bottom) MSM constructed on Cartesian coordinates with 1000 clusters and the same lag time.

where, for $a, b \geq 1$,

$$C_{ab}^t = \langle \varphi_\theta^a, \mathcal{S}^t \varphi_\theta^b \rangle_\mu \tag{4.39}$$

for $t = 0, \tau$, and

$$p_a = \left\langle \varphi_\theta^a, \mathbb{E}_x \left[ \rho(X) \right] \right\rangle_\mu. \tag{4.40}$$

In (4.40), we introduce the notation

$$\rho(X) = \sum_{t=0}^{(\tau \wedge T)-1} \Gamma(X^t) \tag{4.41}$$

for use in Algorithm 2.

When the boundary condition is non-zero, as for the committor, we restrict (4.38) to a $(k-1)$-dimensional linear system by excluding the indices $a = 1$ and $b = 1$ in (4.39) and (4.40) and setting

$$\rho(X) = \varphi_\theta^1(X^{\tau \wedge T}) - \varphi_\theta^1(X^0) + \sum_{t=0}^{(\tau \wedge T)-1} \Gamma(X^t). \tag{4.42}$$

In this case the corresponding approximate solution is

$$u_\theta = \varphi_\theta^1 + \sum_{a=2}^{k} w_a \varphi_\theta^a. \tag{4.43}$$

This approximate solution corresponds to the one given by dynamical Galerkin approximation [35, 156] with the basis $\{\varphi_\theta^a\}_{a=2}^k$ and a "guess" function of $\varphi_\theta^1$.

When the boundary conditions are zero, the orthogonalization procedure and the matrix $K$ are applied to all basis functions as in Section 4.6. When the boundary conditions are non-zero, the orthogonalization procedure is only applied to the basis functions $\{\varphi_\theta^a\}_{a=2}^k$, and $K_{a1} = I_{a1}$, the $a1$ element of the identity matrix. We summarize our procedure for prediction in Algorithm 2.

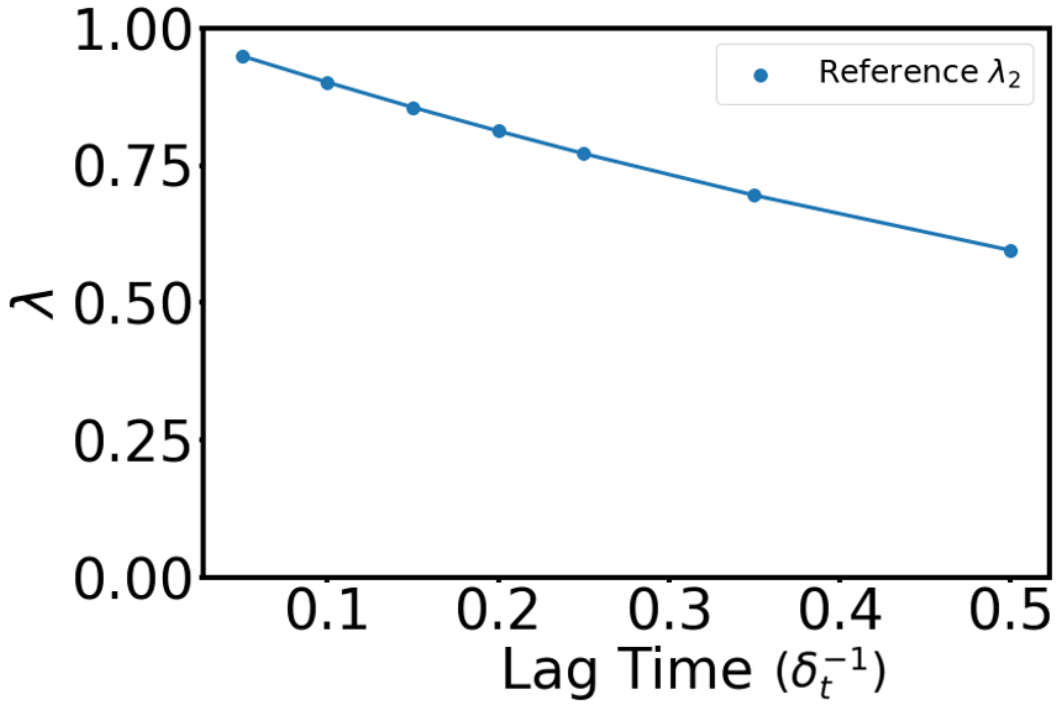


Figure 4.7: First eigenvalue of $\mathcal{S}^\tau$ (second of $\mathcal{A}$ in (4.14)) for the Müller-Brown model as a function of lag time (in units of $\delta_t^{-1}$). The gap between this eigenvalue and the dominant eigenvalue, which is one, determines the rate of convergence of the Richardson iteration.

### *4.7.1 Müller-Brown committor*

In this section, we demonstrate the use of our method for prediction by estimating the committor for the Müller-Brown model with a shallow intermediate basin at $(-0.25, 0.5)$ (Figure 4.1). Here the sets $A$ and $B$ are defined as in Eq. (5.40) and $T$ is the time of first entrance to $D^{\mathsf{c}} = A \cup B$. In this case, a one-dimensional subspace iteration (i.e., $k = 1$ in Algorithm 2) appears sufficient to accurately solve the prediction problem. Figure 4.7 shows the largest eigenvalue of the stopped transition operator $\mathcal{S}^\tau$ (the second largest of $\mathcal{A}$ in (4.14)) computed from our grid-based reference scheme (Section 4.5.1). Richardson iteration should converge geometrically in this eigenvalue[174], and so, for moderate lag times, we can expect our method to converge in a few dozen iterations. To initialize the algorithm we choose $\tilde{\varphi}_1^1 = \mathbb{1}_B$. All other hyperparameters are listed in Table 4.1.

**Algorithm 2** Inexact subspace iteration (with $L^2_\mu$ loss) for prediction functions

**Require:** Subspace dimension $k$, stopped transition data $\{X_j^0, X_j^{\tau\wedge T_j}\}_{j=1}^n$, reward data $\{\rho(X_j)\}_{j=1}^n$, batch size $B$, learning rate $\eta$, number of subspace iterations $S$, number of inner iterations $M$, regularization parameters $\gamma_1$ and $\gamma_2$

Initialize $\{\varphi_\theta^a\}_{a=1}^k$ and $\{\tilde{\varphi}_1^a\}_{a=1}^k$
**for** $s = 1 \dots S$ **do**
  **for** $m = 1 \dots M$ **do**
    Sample a batch of data $\{X_j^0, X_j^{\tau\wedge T_j}\}_{j=1}^B$, $\{\rho(X_j)\}_{j=1}^B$
    $\hat{\mathcal{L}}_1 \leftarrow \frac{1}{B}\sum_{j=1}^B \sum_{a=1}^k \left[\frac{1}{2}(\sum_{b=1}^k \varphi_\theta^b(X_j^0)K_{ba})^2 - \alpha_s\left(\sum_{b=1}^k \varphi_\theta^b K_{ba}(\tilde{\varphi}_s^a(X_j^{\tau\wedge T_j}) + \rho(X_j)I_{a1})\right)\right]$
    $\hat{\mathcal{L}}_2 \leftarrow -\frac{1}{B}\sum_{j=1}^B \sum_{a=1}^k (1-\alpha_s)\sum_{b=1}^k \varphi_\theta^b K_{ba}\tilde{\varphi}_s^a(X_j^0)$
    $\hat{\mathcal{L}}_K \leftarrow \gamma_1 \|K - \mathrm{diag}(K)\|_F^2$
    $\hat{\mathcal{L}}_{\mathrm{norm}} \leftarrow \gamma_2 \sum_{a=2}^k (2\nu_a(\frac{1}{B}\sum_{j=1}^B(\varphi_\theta^a(X_j^0)^2) - 1) - \nu_a^2)$
    $\hat{\mathcal{L}} \leftarrow \hat{\mathcal{L}}_1 + \hat{\mathcal{L}}_2 + \hat{\mathcal{L}}_K + \hat{\mathcal{L}}_{\mathrm{norm}}$
    $\theta \leftarrow \theta - \eta\nabla_\theta\hat{\mathcal{L}}$
    $K \leftarrow K - \eta\,(\mathrm{triu}(\nabla_K\hat{\mathcal{L}}))$
    $\nu_a \leftarrow \nu_a + \eta\nabla_{\nu_a}\hat{\mathcal{L}}$
  **end for**
  **if** $\Psi(x) = 0$ **then**
    Compute the matrix $\Phi_{ia} = \varphi_\theta^a(X_i^0)$ ▷ $\Phi \in \mathbb{R}^{n\times k}$
  **else**
    Compute the matrix $\Phi_{ia} = \varphi_\theta^a(X_i^0)$ for $a > 1$ ▷ $\Phi \in \mathbb{R}^{n\times(k-1)}$
  **end if**
  Compute QR-decomposition $\Phi = QR$
  Compute diagonal matrix $N_{aa}^2 = \sum_i \varphi_\theta^a(X_i^0)^2$
  $\tilde{\varphi}_s^a \leftarrow \sum_{b=1}^k \varphi_\theta^b\,(R^{-1}N)_{ba}$ ▷ if $\Psi(x) = 0$ exclude $a = 1$
  $K_{1:i,i} \leftarrow \arg\min_c \frac{1}{n}\sum_{j=1}^n \left(\sum_{a=1}^i \varphi_\theta^a(X_j^0)c_a - \tilde{\varphi}_s^a(X_j^{\tau\wedge T_j})\right)^2 + \gamma_2\sum_{a=1}^{i-1} c_a^2$
**end for**
Compute the matrix $C_{ab}^t = \frac{1}{n}\sum_{j=1}^n \varphi_\theta^a(X_j^0)\varphi_\theta^b(X_j^t)$ for $t = 0, \tau\wedge T_j$ ▷ $C^t \in \mathbb{R}^{k\times k}$
Compute the vector $p_a = \frac{1}{n}\sum_{j=1}^n \varphi_\theta^a(X_j^0)\rho(X_j)$
Solve linear system $(C^0 - C^\tau)w = p$ ▷ if $\Psi(x) = 0$ enforce $w_1 = 1$
**return** $u = \sum_{a=1}^k w_a\varphi_\theta^a$

We compare the estimate of the committor from our approach with that from an MSM constructed from the same amount of data by using $k$-means to cluster the configurations

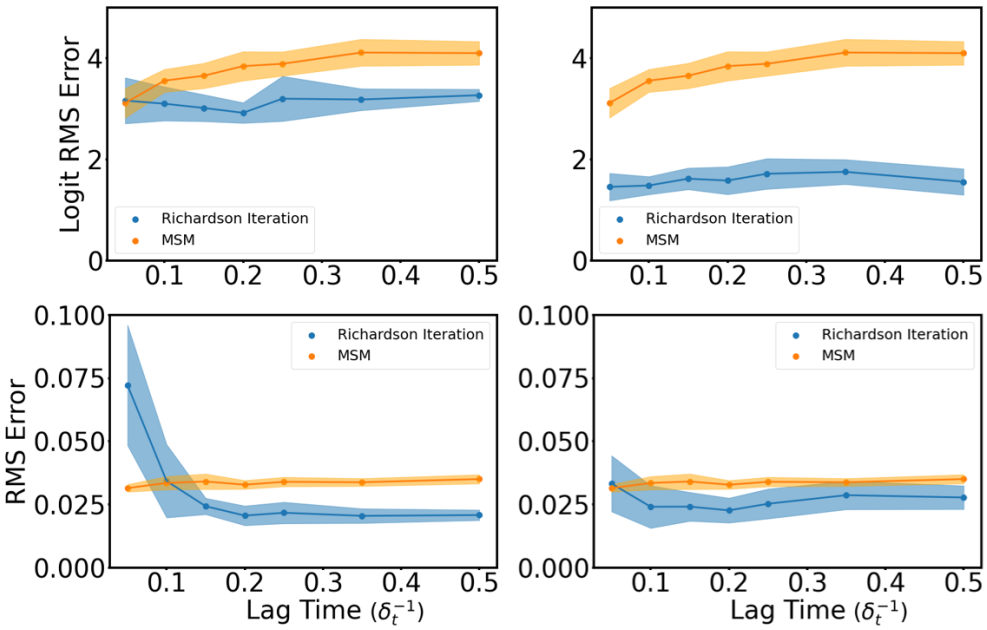


Figure 4.8: Committor prediction for the Müller-Brown system as a function of lag time (in units of $\delta_t^{-1}$). (left) Comparison of the inexact Richardson iteration using the $L^2_\mu$ loss and an MSM with 400 states. (right) Same comparison using the softplus loss in place of the $L^2_\mu$ loss.

outside $A$ and $B$ into 400 states and counting the transitions between clusters. In addition to the root mean square error (RMSE) for the committor itself, we show the RMSE of

$$\text{logit}_\varepsilon(q) = \log\left(\frac{\varepsilon + q}{1 + \varepsilon - q}\right) \tag{4.44}$$

for points outside $A$ and $B$. This function amplifies the importance of values close to zero and one. We include $\varepsilon$ because we want to assign only a finite penalty if the procedure estimates $q$ to be exactly zero or one; we use $\varepsilon = e^{-20}$.

Results as a function of lag time are shown in Figure 4.8. We see that the Richardson iterate is more accurate than the MSM for all but the shortest lag times. When using the $L^2_\mu$ loss, the results are comparable, whereas the softplus loss allows the Richardson iterate to improve the RMSE of the logit function in (4.44) with no decrease in performance with

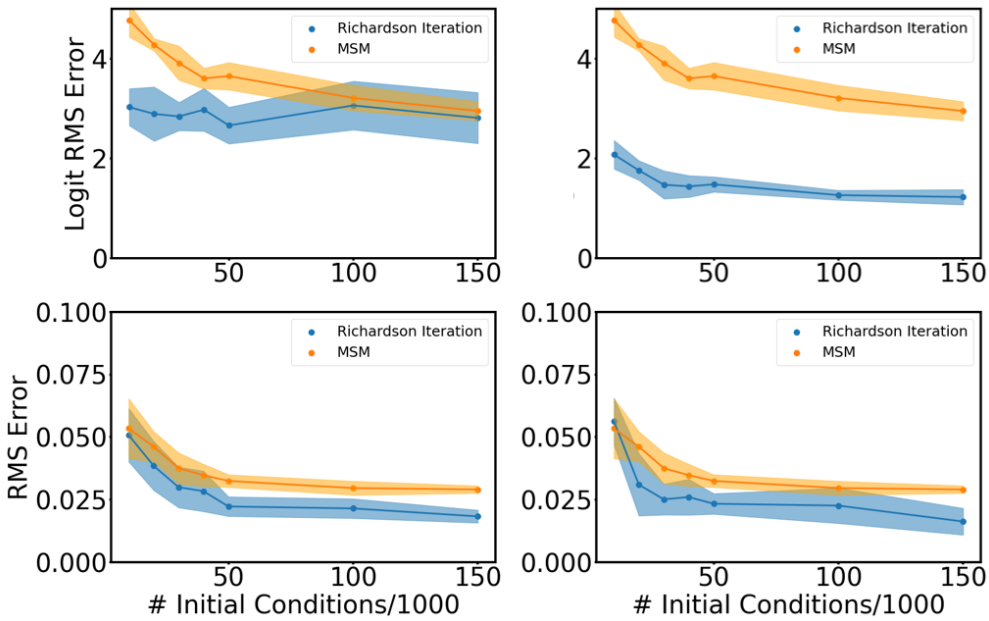


Figure 4.9: Committor prediction for the Müller-Brown as a function of number of initial conditions for a fixed lag time of $\tau = 0.1\delta_t^{-1}$. (left) Comparison of inexact Richardson iteration using the $L^2_\mu$ loss and an MSM with 400 states. (right) Same comparison using the softplus loss in place of the $L^2_\mu$ loss.

respect to the RMSE of the committor. Results as a function of the size of the data set are shown in Figure 4.9 for a fixed lag time of $\tau = 0.1\delta_t^{-1}$. The Richardson iterate generally does as well or better than the MSM. Again, the differences are more apparent in the RMSE of the logit function in (4.44). By that measure, the Richardson iterate obtained with both loss functions is significantly more accurate than the MSM for small numbers of trajectories. The softplus loss maintains an advantage even for large numbers of trajectories.

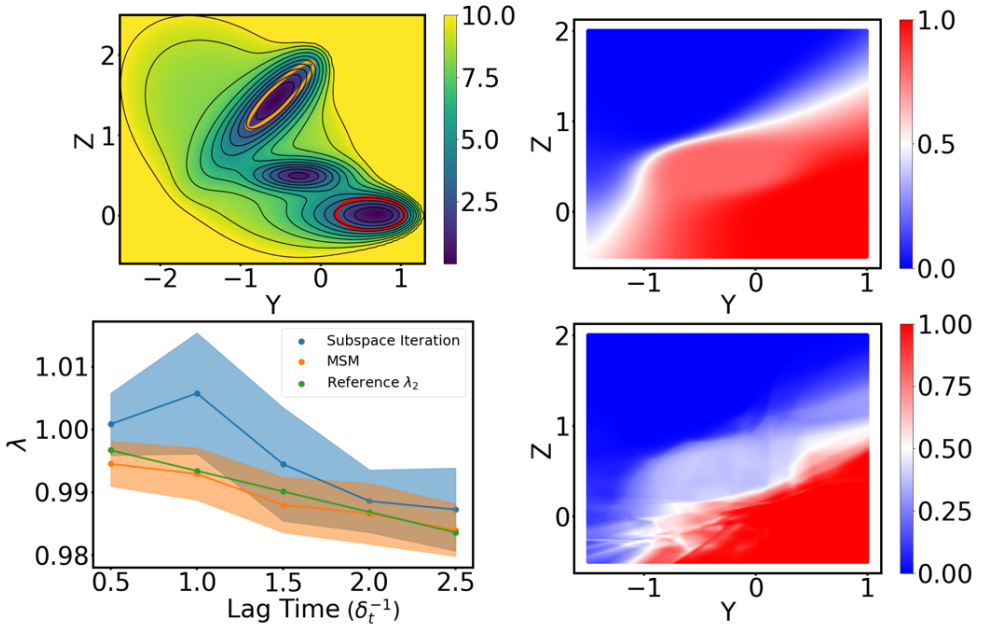


Figure 4.10: Richardson iteration for the committor converges slowly for a Müller-Brown potential with a deepened intermediate. (top left) Potential energy surface, with states $A$ and $B$ indicated. Contour lines are drawn every 1 $\beta^{-1}$. (top right) Reference committor. (bottom left) Dominant eigenvalue as a function of lag time (in units of $\delta_t^{-1}$) from an MSM with 400 states, subspace iteration, and the grid-based reference. (bottom right) Example of the Richardson iteration after 400 iterations. Note the overfitting artifacts and lack of convergence near the intermediate state.

### *4.7.2 Accelerating convergence by incorporating eigenfunctions*

As discussed in Section 4.2, we expect Richardson iteration to converge slowly when the largest eigenvalue of $\mathcal{S}^\tau$, $\lambda_1$, is close to 1. More precisely, the number of iterations required to reach convergence should scale with $-1/\log\lambda_1 = \mathbb{E}[T]/\tau$, the mean escape time from the quasi-stationary distribution to the boundary of $D$ divided by the lag time. With this in mind, we can expect inexact Richardson iteration for the Müller-Brown to perform poorly if we deepen the intermediate basin at $(-0.25, 0.5)$ as in Figure 4.10 (top left). Again, the sets $A$ and $B$ are defined as in (5.40), and $T$ is the time of first entrance to $D^{\mathsf{c}} = A \cup B$. In this case, $-1/\log\lambda_1$ is on the order of 100 for the lag times we consider and, as expected, inexact Richardson iteration converges slowly (Figure 4.10, bottom left). Estimates

of the committor by inexact Richardson iteration do not reach the correct values even after hundreds of iterations (Figure 4.10, bottom right).

We now show that convergence can be accelerated dramatically by incorporating additional eigenfunctions of $\mathcal{S}^\tau$ (i.e., $k > 1$ in Algorithm 2). For the Müller-Brown model with a deepened intermediate basin, the second eigenvalue of $\mathcal{S}^\tau$ is of order $10^{-4}$ for a lag time of $\tau = 1000$ steps or $1\,{\delta_t}^{-1}$ (while the first is near one as discussed above). We therefore choose $k = 2$ with $\tilde{\varphi}_1^2$ initialized as a random linear combination of coordinate functions as in previous examples. We run the subspace iteration for four iterations, compute the committor as a linear combination of the resulting functions, and then refine this result with a further ten Richardson iterations (i.e., $k = 1$ with the starting vector as the output of the $k = 2$ subspace iteration). To combine the functions, we use a linear solve which incorporates memory (Algorithm 3) [197, 198]. We find that the use of memory improves the data-efficiency substantially for poorly conditioned problems. For our tests here, we use three memory kernels, corresponding to $\tau_{\text{mem}} = \lfloor \tau/4 \rfloor$.

The bottom row of Figure 4.11 illustrates the idea of the subspace iteration. The second eigenfunction (Figure 4.11, center) is peaked at the intermediate. As a result, the two neural-network functions linearly combined by the Galerkin approach with memory can yield a good result for the committor (Figure 4.11, bottom right). Figure 4.12 compares the RMSE for the committor and the RMSE for the logit in (4.44) for Algorithm 2 with $k = 1$ (pure Richardson iteration) and $k = 2$ (incorporating the first non-trivial eigenfunction), and an MSM with 400 states. We see that the Richardson iteration suffers large errors at all lag times; as noted previously, this error is mainly in the vicinity of the intermediate. The MSM cannot accurately compute the small probabilities, but does as well as the subspace iteration in terms of RMSE.

**Algorithm 3** Memory-corrected linear solve for predictions

**Require:** Stopped transition data $\{X_j^0, X_j^{1\wedge T_j}, \ldots, X_j^{\tau\wedge T_j}\}_{j=1}^n$, guess function $h$, reward data $\{\rho(X_j)\}_{j=1}^n$, basis set $\{f^a\}_{a=1}^k$, lag between memory kernels $\tau_{\text{mem}}$.

**for** $s = 0 \ldots (\tau/\tau_{\text{mem}})$ **do**
  Initialize the matrix $C^s$ with zeros ▷ $C^s \in \mathbb{R}^{(k+1)\times(k+1)}$
  $C^s_{11} \leftarrow 1$
  **for** $a = 2 \ldots k$ **do**
    $C^s_{a1} \leftarrow \frac{1}{n}\sum_{j=1}^n f^a(X_j^0)\rho(X_j^{s\tau_{\text{mem}}\wedge T_j})$
    **for** $b = 2 \ldots k$ **do**
      $C^s_{ab} \leftarrow \frac{1}{n}\sum_{j=1}^n f^a(X_j^0) f^b(X_j^{s\tau_{\text{mem}}\wedge T_j})$
    **end for**
  **end for**
**end for**
$A \leftarrow C^1 - C^0$
**for** $s = 0 \ldots (\tau/\tau_{\text{mem}}) - 2$ **do**
  $M^s \leftarrow C^{s+2} - C^{s+1} - A(C^0)^{-1}C^{s+1} - \sum_{j=0}^s M^j (C^0)^{-1} C^{s-j}$
**end for**
$A_{\text{mem}} \leftarrow A + \sum_{s=0}^{(\tau/\tau_{\text{mem}})-2} M^s$
Solve $A_{\text{mem}} w = 0$
**return** $u = h + \sum_{a=2}^k w_a f^a$

### *4.7.3 $AIB_9$ prediction results*

As an example of prediction in a high-dimensional system, we compute the committor for the transition between the left- and right-handed helices of $\text{AIB}_9$ using the inexact Richardson iteration scheme ($k = 1$ in Algorithm 2) with the softplus loss function. Specifically, for this committor calculation $T$ is the time of first entrance to $D^{\mathsf{c}} = A \cup B$ with $A$ and $B$ defined in Section 4.5.2. As before, we initialize $\tilde{\varphi}_1^1 = \mathbb{1}_B$.

To validate our results, we use the 5 $\mu$s reference trajectories to compute an empirical committor as a function of the neural network outputs, binned into intervals:

$$\bar{q}(s) = \mathbb{P}\left[X^T \in B \;\middle|\; u_\theta(X^0) \in [s, s+\Delta s]\right] \tag{4.45}$$

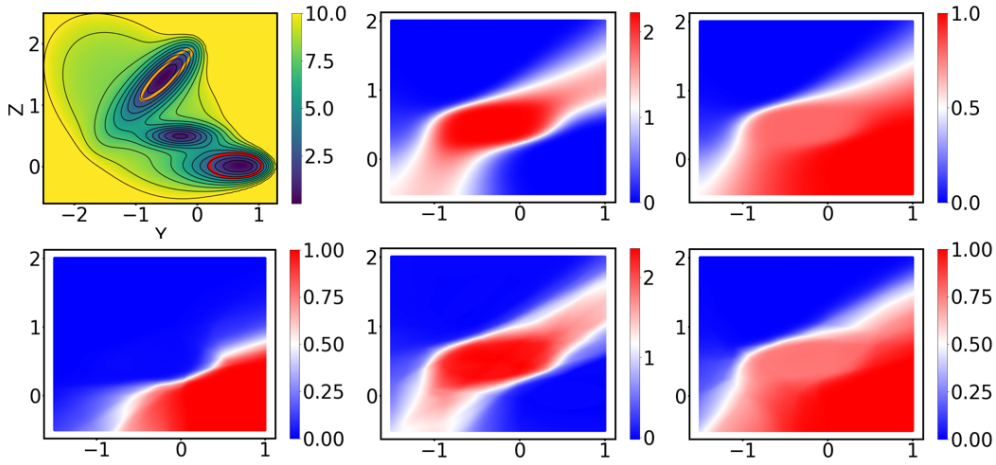


Figure 4.11: Illustration of the subspace iteration for the Müller-Brown committor. (top left) Modified Müller-Brown potential. (top center) Reference second eigenfunction. (top right) Reference committor. (bottom left) Neural-network Richardson iterate after four iterations. (bottom center) First non-dominant eigenfunction obtained from the neural network after four iterations. (bottom right) Committor resulting from linear combination of the Richardson iterate and second eigenfunction. Results shown are for $\tau = 1000$ steps (i.e., $1\,{\delta_t}^{-1}$).

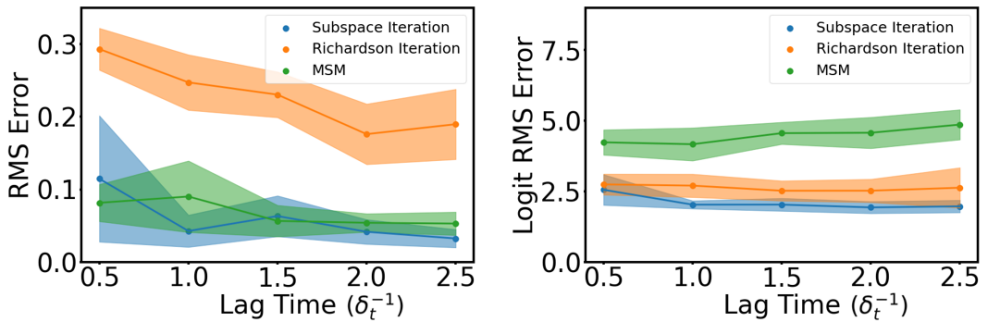


Figure 4.12: Committor for the Müller-Brown potential with deepened intermediate as a function of lag time (in units of ${\delta_t}^{-1}$). (left) Comparison of RMSE for subspace iteration as described above, Richardson iteration (as in Section 4.7.1 but instead with 500 subspace iterations), and an MSM with 400 states. (right) RMSE of the logit function in (4.44).

for $s \in [0, 1 - \Delta s]$. Here, we use $\Delta s = 0.05$. The overall error in the committor estimate is defined as

$$q \text{ error} = \left( \Delta s \sum_{n=0}^{1/\Delta s - 1} \left[ \bar{q}(n\Delta s) - n\Delta s \right]^2 \right)^{1/2} . \tag{4.46}$$

While this measure of error can only be used when the data set contains trajectories of long enough duration to reach $D^{\mathsf{c}}$, it has the advantage that it does not depend on the choice of

projection that we use to visualize the results.

Results for the full data set with $\tau$ corresponding to 400 ps are shown in Figure 4.13. The projection on the principal components is consistent with the symmetry of the system, and the predictions show good agreement with the empirical committors. As $\tau$ decreases, the results become less accurate (Figure 4.14, top left); at shorter lag times we would expect further increases in the error. We also examine the dependence of the results on the size of the data set by subsampling the short trajectories and then training neural networks on the reduced set of trajectories (Figure 4.14, top right). We find that the performance steadily drops as the number of trajectories is reduced and degrades rapidly for the data sets subsampled more than 20-fold (Figure 4.14, bottom), corresponding to about 7 $\mu$s of total sampling.

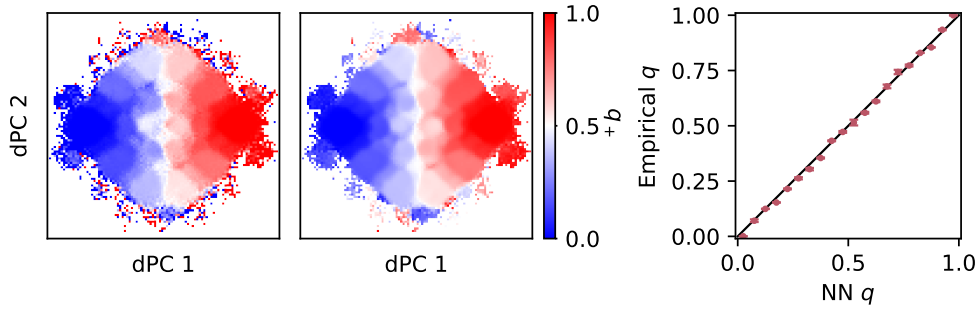


Figure 4.13: $AIB_9$ committor for the transition between left- and right-handed helices. (left) Averages of $\mathbb{1}_B(X^T)$ for initial conditions in bins in the first two dPCs computed from 20 long (5 $\mu$s) trajectories. (middle) Averages of representative neural-network committors trained on the data set of 6,910 short (20 ns) trajectories; $\tau$ corresponds to 400 ps. (right) Comparison between empirical committors (as defined in (4.45)) and the neural-network committors (trained as for the middle panel). Error bars indicate standard deviations over ten different initializations of the neural-network parameters.

Finally, we compute the MFPT to reach the right-handed helix using the same data set. For the MFPT calculation $T$ is the time of first entrance to $D^{\mathsf{c}} = B$. Note that the time of first entrance to $B$ includes long dwell times in $A$ and is expected to be much larger than the time of first entrance to $A \cup B$.

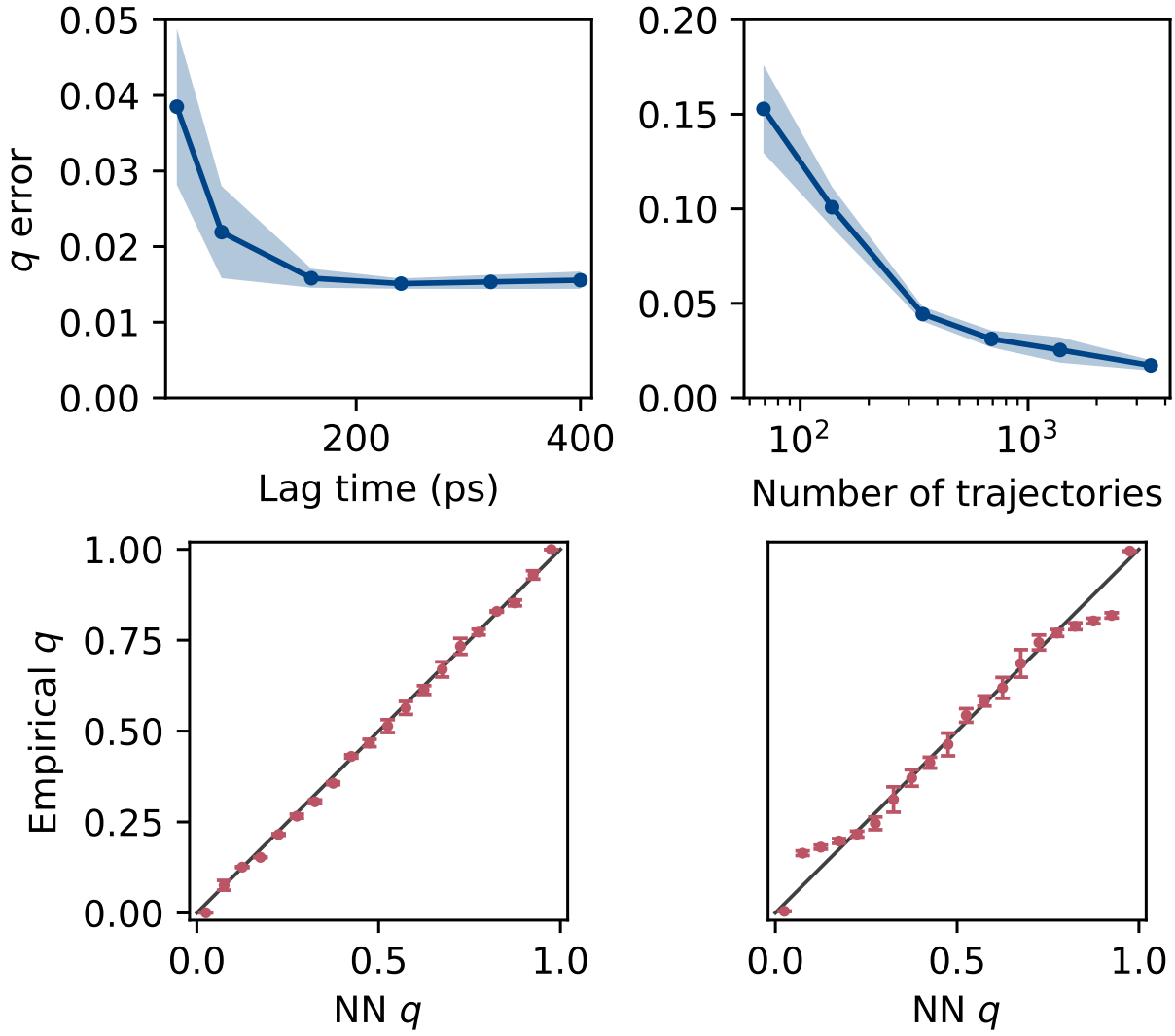


Figure 4.14: $AIB_9$ committor for the transition between left- and right-handed helices, as functions of lag time (in ps) and number of initial conditions. (top left) Error in the committor as a function of lag time (in ps). Shading indicates the standard deviation over ten different initializations of the neural-network parameters. (top right) Error in the committor as a function of the number of initial conditions with $\tau$ corresponding to 160 ps. Shading indicates the standard deviation over ten different random samples of the trajectories. (bottom) Comparison between empirical committors and neural-network committors trained on data sets with (left) 1/2 and (right) 1/20 of the short trajectories. Error bars indicate standard deviations over ten random samples of the trajectories.

We compare against an empirical estimate of the MFPT defined by

$$\bar{m}(s) = \mathbb{E}\left[T \middle| u_\theta(X^0) \in [s, s + \Delta s]\right] \tag{4.47}$$

for $s \in [0, m_{\max} - \Delta s]$ where $\Delta s = 3$ and $m_{\max} = 57$ ns. Overall error is defined analogously to Eq. (4.46).

In Figure 4.15, we show the MFPT obtained from Algorithm 2 with $k = 5$ and the $L^2_\mu$ loss function. Initially we set $\tilde{\varphi}^1_1$ equal to an arbitrary positive function (we use $5\mathbb{1}_A$) and $\tilde{\varphi}^a_s$ for $a > 1$ to a random linear combination of coordinate functions. In Figure 4.16 we examine the convergence of the MFPT from the left-handed helix to the right-handed helix for the MFPT computed with $k = 1$ (pure Richardson iteration) and $k = 5$. The horizontal line indicates a MFPT of about 56 ns estimated from the long reference trajectories. We see that the algorithm with $k = 5$ converges much faster (note the differing scales of the horizontal axes) and yields accurate results at all lag times other than the shortest shown. The need to choose $k > 1$ for this MFPT calculation is again consistent with theoretical convergence behavior of exact subspace iteration. Because the typical time of first entrance to $B$ from points in $A$ is very large, we expect the dominant eigenvalue of $\mathcal{S}^\tau$ to be very near to one when $D = B^{\mathsf{c}}$. In contrast, the committor calculation benefits from the fact that the time of first entrance to $A \cup B$ is much shorter, implying a smaller dominant eigenvalue of $\mathcal{S}^\tau$ when $D = (A \cup B)^{\mathsf{c}}$.

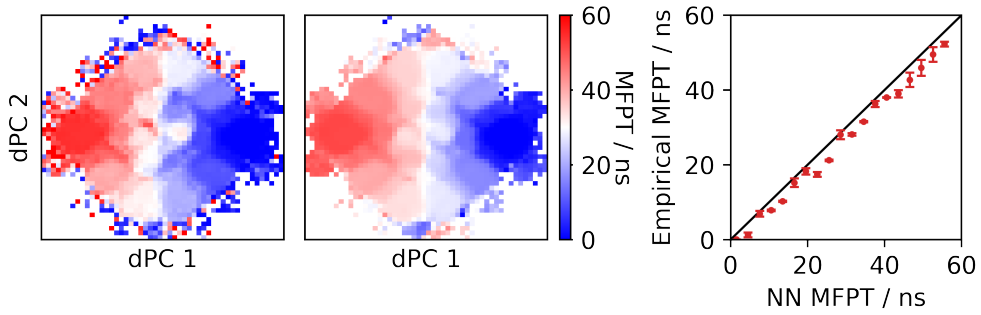


Figure 4.15: $AIB_9$ MFPT to the right-handed helix. (left) Averages of the time to next reach $B$ for initial conditions in bins in the first two dPCs computed from 20 long (5 $\mu$s) trajectories. (middle) Averages of representative neural-network committors trained on the data set of 6,910 short (20 ns) trajectories; $\tau$ corresponds to 400 ps. (right) Comparison between empirical committors (as defined in (4.47)) and the neural-network committors (trained as for the middle panel). Error bars indicate standard deviations over ten different initializations of the neural-network parameters.

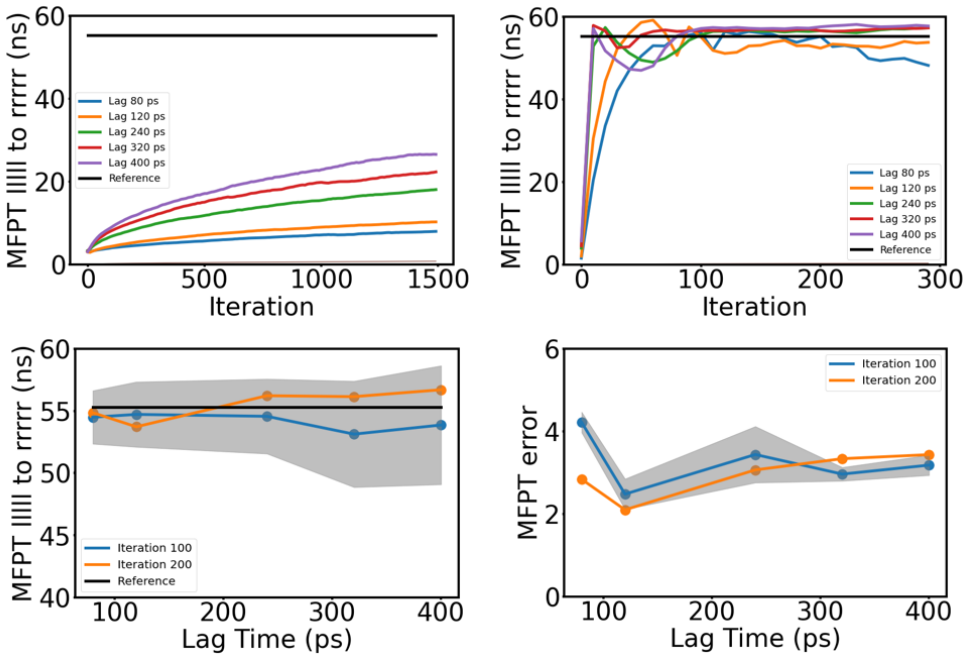


Figure 4.16: MFPT between left- and right-handed helices for the $AIB_9$ system. (top left) Convergence of Richardson iteration. The llllll to rrrrr MFPT is computed by averaging the richardson iteration result over each frame of each of the long reference trajectories in the llllll state. (top right) Convergence of a five-dimensional subspace iteration. (bottom left) MFPT after 100 and 200 subspace iterations as a function of lag time. Shading indicates standard deviations over ten different initializations of the neural-network parameters. (bottom right) Overall error in MFPT. To obtain the results shown in this figure, we first use the short-trajectory dataset to train neural networks to predict the MFPT; we then use these networks with fixed parameters to evaluate the MFPT for all structures in the long reference trajectories and average the results for those structures in the left-handed helix state. The horizontal lines in the top panels are obtained from averaging the time to the right-handed helix for the same structures.

## 4.8 Conclusions

In this work we have presented a method for spectral estimation and rare-event prediction from short-trajectory data. The key idea is that we use the data as the basis for an inexact subspace iteration. For the test systems that we considered, the method not only outperforms high-resolution MSMs, but it can be tuned through the choice of loss function to compute committor probabilities accurately near the reactants, transition states, and products. Other

than the Markov assumption, our method requires no knowledge of the underlying model and puts no restrictions on its dynamics.

As discussed in prior neural-network based prediction work[167, 171], our method is sensitive to the quality and distribution of the initial sampling data. However, our work shares with Ref. 167 the major advantage of allowing the use of arbitrary inner products. This enables adaptive sampling of the state space [199, 167] and—together with the features described above—the application to observational and experimental data, for which the stationary distribution is generally unavailable.

In the present work, we focused on statistics of transition operators, but our method can readily be extended to solve problems involving their adjoint operators as well. By this means, we can obtain the stationary distribution as well as the backward committor. The combination of forward and backward predictions allows the analysis of transition paths using transition path theory without needing to generate full transition paths[178, 57, 187] and has been used to understand rare transition events in molecular dynamics[156, 158, 161, 162, 200, 201] and geophysical flows[202, 203, 204, 205, 206]. We leave these extensions to future work.

In cases in which trajectories that reach the reactant and product states are available, it would be interesting to compare our inexact iterative schemes against schemes for committor approximation based on logistic regression and related approaches [176, 181, 182, 81, 183, 184, 185, 186]. These schemes are closely related to what is called "Monte-Carlo" approximation in reinforcement learning [172], and also to the inexact Richardson iteration that we propose here with $\tau \to \infty$.

We have seen that temporal difference (TD) learning, a workhorse for prediction in reinforcement learning, is closely related to an inexact form of Richardson iteration. Variants like TD($\lambda$), have similar relationships to inexact iterative schemes. As we showed, subspace iteration is a natural way of addressing slow convergence. We thus anticipate that our results

have implications for reinforcement learning, particularly in scenarios in which the value function depends on the occurrence of some rare-event. Finally, we note that our framework can be extended to the wider range of iterative numerical linear algebra algorithms. In particular, Krylov or block Krylov subspace methods may offer further acceleration. In fact, very recently an approach along these lines was introduced for value function estimation in reinforcement learning[207].

# CHAPTER 5

# COMPUTING STATIONARY DISTRIBUTIONS BY RAPIDLY CONVERGING TRAJECTORY STRATIFICATION

## 5.1 Introduction

The sampling of a stochastic process can be controlled by evolving an ensemble and splitting and merging the trajectories of its members, and various algorithms based on this strategy have been introduced[20, 208]. Because the trajectory segments between splitting events are unbiased, such algorithms can yield dynamical statistics, such as transition probabilities and mean first passage times (MFPTs) to selected states, in addition to steady-state probability distributions and averages. Furthermore, splitting algorithms generally require little communication between the members of the ensemble, making them relatively straightforward to implement regardless of the underlying dynamics, and community software is available [209]. As a result, splitting algorithms are widely used.

Recent mathematical analysis of one of the oldest splitting algorithms in the molecular simulation literature, weighted ensemble (WE), shows that it is asymptotically unbiased and can dramatically reduce the variance of statistics when the splitting criteria, which are based on a partition of the state space, are chosen appropriately [210, 211]. However, the method relies on convergence of the steady-state ensemble of trajectories, which is known to be slow. In fact, as we argue below and as was observed previously[211], it is as slow as running direct unbiased simulations. Prior to convergence, the method is systematically biased.

Convergence can be accelerated by introducing an element of stratification. That is, to manipulate the weights of not just individual trajectory segments but groups of them based on their common features. In particular, nonequilibrium umbrella sampling (NEUS)[212, 213, 214, 11, 215, 14] groups trajectories that are in the same regions of state space, as defined by collective variables (CVs), and estimates the steady-state probabilities of the

regions by solving a global flux balance equation; this strategy was subsequently adopted in exact milestoning (EM) and extensions of WE[216, 217]. A unified framework for trajectory stratification that incorporates elements of all of the above algorithms and is unbiased in the limit that each region contains an infinite number of ensemble members is presented in Ref. 14, which also shows that the regions can be defined in terms of path-based quantities.

In the present work, we show how to accelerate convergence further. Our starting point is the fact that, as noted above, splitting and trajectory stratification algorithms sample unbiased trajectory segments. As a result, their data can be used to construct Markov State Models (MSMs)[159]. Indeed, Zuckerman and co-workers used history-augmented MSMs, which separate trajectories based on the last metastable state from which they came, to improve MFPTs from WE[218]. In that case, the MSMs are used after sampling to improve statistical estimates. Here, we extend this general approach to accelerate the sampling itself. In formulating our algorithm, we first present the weighted ensemble method in such a way that it is clear exactly how it can be modified to accelerate convergence without introducing bias. We show that the NEUS algorithm is a special case of our new scheme.

The remainder of the paper is organized as follows. In Section 5.2, we review the WE algorithm and recast it to show why an initialization bias is slow to disappear. This formulation also clarifies the relation of WE to the NEUS algorithm as described in Ref. 14, and we show how software for WE can be easily extended to enable trajectory stratification in Section 5.3. In Section 5.4, we introduce a generalization of MSMs that allows more freedom in the choice of basis functions[35, 156, 160] and in turn our strategy for accelerating sampling, which we term Basis-Accelerated NEUS (BAD NEUS). We show that NEUS corresponds to a particular choice of basis set. In Section 5.6, we demonstrate BAD NEUS on a two-dimensional model with a known steady-state distribution.

## 5.2 Weighted Ensemble (WE)

As described above, in WE, the state space is partitioned into regions, and then trajectories are cloned into multiple copies (splitting) or removed from the ensemble (killing) criteria are based on them. Here, we use a relatively simple procedure and represent the splitting and killing by resampling the ensemble within each region. Mathematically, we denote the state of ensemble member (henceforth, walker) $i$ at time $t$ by $X_i^t$, and we associate with $i$ a weight $w_i^t$ such that $\sum_i w_i^0 = 1$. We track the index of the region containing $X_i^t$ by an index process $J_i^t$. For example, $J_i^t$ might track the element of a partition of state space in which $X_i^t$ currently resides. Other options for the index process are available and advantageous (see Refs. [14, 219] for further discussion). Each iteration of the sampling consists of two steps: evolution according to the underlying dynamics and resampling the ensemble. In the evolution step, for each walker $i$, we numerically integrate for a time interval $\Delta$ to obtain $X_i^{t+\Delta}$ from $X_i^t$ and update $J_i^{t+\Delta}$ accordingly. In this step the weights are unchanged; $w_i^{t+\Delta} = w_i^t$. We note that $\Delta$ can be a stopping time, not just a fixed time interval. An often useful choice is to take $\Delta$ to be the first time that the value of the index process changes. In this case, each walker has a different value for $\Delta$. In the resampling step, for each value $k$ of the index process (e.g., for each region in a partition of state space), if there is at least one walker with $J_i^{t+\Delta} = k$, we select a number $N_k$. Then we sample from the set $\Phi = \{i : J_i^{t+\Delta} = k\}$ $N_k$ times with replacement according to the probabilities $p_i = w_i^{t+\Delta} / \sum_{i\in\Phi} w_i^{t+\Delta}$. For each index $r$ so chosen, we append $(X_r^{t+\Delta}, J_r^{t+\Delta}, \sum_{i\in\Phi} w_i^{t+\Delta}/N_k)$ to the new ensemble. If the underlying dynamics are ergodic, WE can be used to compute steady-state averages of functions as

$$\mathbb{E}_{X^0\sim\pi}[g(X^0, X^1, \ldots, X^{\Delta-1})]$$
$$= \lim_{T\to\infty} \frac{1}{T} \sum_{n=0}^{T} \frac{1}{N} \sum_{i=0}^{N} g(X_i^{n\Delta}, X_i^{n\Delta+1}, \ldots, X_i^{n\Delta+\Delta-1}) w_i^{n\Delta}, \quad (5.1)$$

where the subscript on the expectation indicates that we draw the initial state $X^0$ from the steady-state distribution $\pi$, and $n$ indexes successive weighted ensemble iterations. Here and below, $X$ without a subscript indicates a realization of the underlying Markov process rather than a walker in an ensemble. An important example is the steady state flux into a particular set $D$ of interest, which is obtained by setting $g(X^0, \ldots, X^\Delta) = \mathbb{1}_{D^c}(X_0)\mathbb{1}_D(X^1)$, where $\mathbb{1}_D(x)$ is an indicator function that is 1 if $x \in D$ and 0 otherwise. This amounts to counting the number of walkers that enter $D$ in a single time step and summing the total weight of those walkers.

Having stated the basic WE algorithm, we now present WE in a new way. While this may initially appear to complicate the description, it reveals why WE is slow to converge and facilitates the introduction of approaches to accelerate convergence. In this description, which we call the distribution representation of WE, we directly evolve distributions rather than approximating them through individual walkers. To this end, we define the flux distributions

$$\overline{\pi}_\ell(dx|k) = P[X^{S(\ell)} \in dx | J^{S(\ell)} = k] \quad (5.2)$$

where $dx$ is an infinitesimal volume in state space that is located at a specific value of $x$ and $S(\ell)$ is an increasing sequence of stopping times. For example, if we choose $S(\ell) = \ell\Delta$ then the following description will correspond to the standard version of WE already sketched. Alternatively, we might let $S(\ell)$ be the time of the $\ell$-th change in the value of the index

process $J$:

$$\begin{aligned} S(0) &= 0 \\ S(1) &= \min\{t > 0 : J_t \neq J^{t-1}\} \\ &\vdots \\ S(\ell) &= \min\{t > S(\ell-1) : J^t \neq J^{t-1}\}. \end{aligned} \tag{5.3}$$

In all practical implementations of the sampling algorithms that we discuss, the flux distributions take the form of empirical distributions of $N$ walkers:

$$\overline{\pi}_\ell(dx|k) = \frac{1}{\overline{z}_\ell^k} \sum_{i=1}^{N} w_i^{S_i(\ell)} \delta_{X_i^{S_i(\ell)}}(dx) \mathbb{1}_{\{k\}}(J_i^{S_i(\ell)}), \tag{5.4}$$

where $\delta_x$ is the Dirac delta function centered at position $x$, and the normalization constant $\overline{z}_\ell^k$ is the distribution of weight restricted to region $k$ at time $S(\ell)$:

$$\overline{z}_\ell^k = P[J^{S(\ell)} = k] = \sum_{i=1}^{N} w_i^{S_i(\ell)} \mathbb{1}_{\{k\}}(J_i^{S_i(\ell)}). \tag{5.5}$$

We now write the evolution and resampling steps in terms of operators. For the former, we define the operator $\mathcal{U}$, which propagates a flux distribution for one change of the index process:

$$\mathcal{U}\overline{\pi}_\ell(dx, j) = \overline{\pi}_{\ell+1}(dx, j). \tag{5.6}$$

In the long-time limit, this yields the eigenequation

$$\overline{\pi}(dx, j) = \mathcal{U}\overline{\pi}(dx, j), \tag{5.7}$$

where $\overline{\pi}$ is the steady-state flux distribution. In practice, one approximates the distributions

through walkers, and we denote the corresponding evolution by $\tilde{\mathcal{U}}$. Mathematically,

$$\tilde{\mathcal{U}}\left[\frac{1}{N}\sum_{i=1}^{N} w_i^{S_i(\ell)}\delta_{X_i^{S_i(\ell)}}(dx)\mathbb{1}_k(J_i^{S_i(\ell)})\right] = \frac{1}{N}\sum_{i=1}^{N} w_i^{S_i(\ell)}\delta_{X_i^{S_i(\ell+1)}}(dx)\mathbb{1}_k(J_i^{S_i(\ell+1)}). \quad (5.8)$$

The subscript on the weight does not change because the evolution step only affects the state and index of a walker, not its weight. $\tilde{\mathcal{U}}$ as defined in (5.8) is a unbiased stochastic approximation of $\mathcal{U}$ in the sense that, for any function $f$ and any distribution $\rho$,

$$\sum_j \int f(x,j)[\mathcal{U}\rho](dx,j) = \mathbb{E}\left[\sum_j \int f(x,j)[\tilde{\mathcal{U}}\rho](dx,j)\right]. \quad (5.9)$$

If distributions are represented with some approximate ansatz such as a neural network, one can also approximately apply the propagator using a variational method as outlined in [173]. This scheme would not require new trajectories to be run at each iteration, nor would it require a resampling step.

As noted above, we also represent resampling through an operator, $\mathcal{R}_k$. We define it such that, for any distribution $\rho$ and any function $f$

$$\sum_j \int f(x,j)\left(\frac{1}{\zeta_k}\mathbb{1}_{\{k\}}(j)\rho(dx,j)\right) = \mathbb{E}\left[\sum_j \int f(x,j)\mathcal{R}_k\rho\right] \quad (5.10)$$

and

$$\zeta_k = \sum_j \int \mathbb{1}_{\{k\}}(j)\rho(dx,j). \quad (5.11)$$

Above, the LHS selects the portion of the distribution $\rho$ in region $k$ and then renormalizes it, while the RHS corresponds to resampling from the distribution. A simple choice which is commonly employed for the operator $\mathcal{R}_k$ is to sample the set of walkers in region $k$ at the end of the evolution (denoted $K$) from a multinomial distribution with probability proportional

to $\{w_i^{S_i(\ell)} \mathbb{1}_{\{k\}}(J_i^{S_i(\ell)})\}_{i \in K}$ with $N_k$ trials and then return the distribution

$$\mathcal{R}_k \left[ \frac{1}{N} \sum_{i=1}^{N} w_i^{S(\ell)} \delta_{X_i^{S(\ell)}}(dx) \mathbb{1}_{\{k\}}(J_i^{S(\ell)}) \right] = \frac{1}{N_k} \sum_{j \in K} \delta_{X_j^{S(\ell)}}(dx) \tag{5.12}$$

and normalization

$$\overline{z}_\ell^k = \sum_i \mathbb{1}_{\{k\}}(J_i^{S_i(\ell)}) w_i^{S_i(\ell)}. \tag{5.13}$$

In words, (5.12) shows that $\mathcal{R}_k$ constructs the state distribution from a sum over the resampled walkers in each region. There are other possible ways of resampling in the walker representation, such as stratified or pivotal resampling, and these may in practice be better. However, any choice that satisfies (5.10) suffices for our discussion.

Finally, we connect the flux distributions $\overline{\pi}(dx|k)$ to the stationary distribution $\pi(x, j)$ for the process $(X^t, J^t)$, where the absence of subscripts here indicates the dynamics in general rather than those of an individual walker. We do this with a key identity, which we now state. For a function $g$ of a length $\tau$ trajectory,

$$\mathbb{E}_{(x,k)\sim\pi}[g(X^0, J^0, \ldots, X^t, J^t)] = \frac{\sum_k \overline{z}^k \int \overline{\pi}(dx|k) \mathbb{E}_{X^0=x, J^0=k} \left[ \sum_{t=0}^{S(1)-1} g(X^t, J^t, \ldots, X^{t+\tau}, J^{t+\tau}) \right]}{\sum_k \overline{z}^k \int \overline{\pi}(dx|k) \mathbb{E}_{X^0=x, J^0=k} [S(1)]}. \tag{5.14}$$

This identity says that, for each region $k$, we draw initial walker states $X_0$ from $\overline{\pi}(dx|k)$ and set $J_0 = k$; we evolve those walkers until they leave region $k$; then we compute averages over them and weight the contribution from the walkers that started in region $k$ by $\overline{z}^k$, which is the steady-state limit of $\overline{z}_\ell^k$. We derive (5.14) in Appendix 5.8.1. The distribution representation of WE is summarized in Algorithm 7. The distribution representation of WE is summarized in Algorithm 7.

**Algorithm 4** WE (distribution representation)

**Require:** Approximate Markov propagator $\tilde{\mathcal{U}}$, Resampling operator $\mathcal{R}_k$, starting flux distributions $\{\overline{\pi}_0(dx|k)\}_{k=1}^n$, initial region weights $\{\overline{z}_0^k\}_{k=1}^n$ with $\sum_k \overline{z}_0^k = 1$.

1: **for** $\ell = 0 \dots L$ **do**
2: $\quad \overline{\pi}_\ell(dx, j) \leftarrow \overline{z}_\ell^j \overline{\pi}_\ell(dx|j)$
3: $\quad \tilde{\overline{\pi}} \leftarrow [\tilde{\mathcal{U}}\overline{\pi}_\ell](dx, j)$ ▷ One Power Iteration Step
4: $\quad$ **for** $k = 1 \dots n$ **do**
5: $\quad\quad \overline{\pi}_{\ell+1}(dx|k), \overline{z}_{\ell+1}^k \leftarrow \mathcal{R}_k[\tilde{\overline{\pi}}]$
6: $\quad$ **end for**
7: **end for**
8: **return** $\overline{\pi}_L(dx|j)$ and $\overline{z}_L^j$

We see that each step of WE applies the operator $\mathcal{U}$ to the previous distribution $\overline{\pi}_\ell(dx, j)$. Therefore, WE can be seen as performing a power iteration in $\mathcal{U}$, starting from some initial distribution. The resampling step in WE plays a role analogous to the renormalization step in standard power iteration in the following sense. In regular power iteration, the renormalization step serves to prevent the iterate from becoming too small or too large and introducing numerical instability. In WE, the resampling step serves to prevent any of the region-distributions from becoming poorly sampled, and thus increasing variance. However, the renormalization step in power iteration and, in turn, the resampling step in WE does not accelerate decay of initialization bias, nor is the WE autocorrelation time reduced relative to direct sampling.

Algorithm 6 suggests that, if $\tilde{\overline{\pi}} = \overline{\pi}$ in step 5, convergence would be achieved. Therefore, our strategy for accelerating WE is to replace the power iteration step with a more accurate approximation. Specifically, we compute changes of measure $\nu_\ell$ such that

$$\overline{\pi}(dx, j) \approx \nu_\ell(x, j)\overline{\pi}_\ell(dx, j). \tag{5.15}$$

If we set up our approximation such that whenever $\pi_\ell = \pi$, we get $\nu_\ell = 1$, then the resulting modification of WE will have a fixed point at the true steady-state distribution, and there will be no approximation error resulting from the finite basis, but our scheme can approach steady state potentially much faster than regular WE.

## 5.3 Nonequilibrium umbrellla sampling (NEUS)

We now present NEUS as a simple extension to WE that accelerates convergence in the way suggested in (5.15). Our development is based on the algorithm in Ref. [14], which corrects a small systematic bias in earlier NEUS papers [212, 213, 214, 11, 215]. We begin by making the observation that, at steady state,

$$\mathbb{E}_{X_0\sim\overline{\pi}}[\mathbb{1}_{\{j\}}(J^0)] = \mathbb{E}_{X^0\sim\overline{\pi}}[\mathbb{1}_{\{j\}}(J^{S(1)})] \tag{5.16}$$

Or, in terms of the density we wish to compute:

$$\int \overline{\pi}(dx, j) = \sum_k \int \overline{\pi}(dx, k)\mathbb{E}_{X^0=x,J^0=k}[\mathbb{1}_{\{j\}}(J^{S(1)})] \tag{5.17}$$

This observation introduces one equation per possible value of the index $j$. Therefore, we can parameterize the change of measure with as many free parameters:

$$\overline{\pi}(dx, j) \approx c_\ell^j \mathbb{1}_{\{j\}}\overline{\pi}_\ell(dx, j) = c_\ell^j \mathbb{1}_{\{j\}}\overline{z}_\ell^j\overline{\pi}_\ell(dx|j) \tag{5.18}$$

Substituting this ansatz into (5.17) and simplifying yields the matrix equation

$$c_\ell^j\overline{z}_\ell^j = \sum_k c_\ell^k\overline{z}_\ell^k\overline{G}_{kj}, \tag{5.19}$$

where

$$\overline{G}_{kj} = \int \overline{\pi}_*(dx|k)\mathbb{E}_{X^0=x,J^0=k}[\mathbb{1}_{\{j\}}(J^{S(1)})]. \tag{5.20}$$

We summarize NEUS in Algorithm 5.

**Algorithm 5** NEUS

**Require:** Approximate Markov propagator $\tilde{\mathcal{U}}$, Resampling operator $\mathcal{R}_k$, starting flux distributions $\{\overline{\pi}_0(dx|k)\}_{k=1}^n$, initial region weights $\{\overline{z}_0^k\}_{k=1}^n$ with $\sum_k \overline{z}_0^k = 1$.

1: **for** $\ell = 0 \dots L$ **do**
2: $\quad \overline{G}_{kj} \leftarrow \int \overline{\pi}_\ell(dx|k)\mathbb{E}_{X^0=x,J^0=k}[\mathbb{1}_{\{j\}}(J_{S(1)})]$
3: $\quad$ Solve $c_\ell \overline{z}_\ell \overline{G} = c_\ell \overline{z}_\ell$ for $c_\ell \overline{z}_\ell$
4: $\quad \overline{\pi}_\ell(dx, j) \leftarrow c_\ell^j \mathbb{1}_{\{j\}} \overline{z}_\ell^j \overline{\pi}_\ell(dx|j)$
5: $\quad \tilde{\overline{\pi}} \leftarrow \tilde{\mathcal{U}} \overline{\pi}_\ell(dx, j)$
6: $\quad$ **for** $k = 1 \dots n$ **do**
7: $\quad\quad \overline{\pi}_{\ell+1}(dx|k), \overline{z}_{\ell+1}^k \leftarrow \mathcal{R}_k[\tilde{\overline{\pi}}]$
8: $\quad$ **end for**
9: **end for**
10: **return** $\overline{\pi}_L(dx|j)$ and $\overline{z}_L^j$

We thus see that NEUS is distinguished from WE by steps 2 and 3. [note here about distribution algorithm vs walker algorithm, theory vs practice] In practice, walkers are drawn and their dynamics are simulated to determine $S(\ell+1)$ in the loop over $\ell$ prior to step 2 (the resulting state distribution is used later in step 5); then, $\overline{G}_{kj}$ is computed in step 2 from walkers at iteration $\ell$ as

$$\overline{G}_{kj} \approx \frac{\sum_i \mathbb{1}_{\{k\}}(J_i^{S_i(\ell)})\mathbb{1}_{\{j\}}(J_i^{S_i(\ell+1)})}{\sum_i \mathbb{1}_{\{k\}}(J_i^{S_i(\ell)})}. \tag{5.21}$$

We solve for the product $c_\ell \overline{z}_\ell$ directly and update the approximation for $\overline{\pi}$ accordingly. We

then proceed as in WE.

We note that, if the flux distributions $\overline{\pi}(dx|k)$ and normalization constants $\overline{z}_k$ assume their steady-state values, then (5.19) is satisfied by $c_j = 1$, and NEUS has the correct fixed point.

## 5.4 BAD NEUS

To improve on NEUS, it is necessary to improve the approximation of the change of measure in (5.15). Here we do so by introducing a basis expansion. This allows us to vary the expressivity of the approximation through the number of basis functions, which can exceed the number of regions, in contrast to the steady-state condition in (5.16).

To this end, we note that, for any lag time $\tau$ and any function $f(x, j)$,

$$\mathbb{E}_{(X^0,J^0)\sim\pi}[f(X^0, J^0) - f(X^\tau, J^\tau)] = 0. \tag{5.22}$$

We stress that this relation holds for any fixed time $\tau$. Expanding this expectation using (5.14), with $g(X^0, J^0, \ldots, X^\tau, J^\tau) = f(X^0, J^0) - f(X^\tau, J^\tau)$ and multiplying through by the normalization in the denominator, we obtain

$$0 = \sum_k \overline{z}^k \int \overline{\pi}(dx|k)\mathbb{E}_{X^0=x,J^0=k}\left[\sum_{t=0}^{S(1)-1} f(X^t, J^t) - f(X^{t+\tau}, J^{t+\tau})\right]. \tag{5.23}$$

We then introduce the basis set $\{\phi_p(x, k)\}_p$, and write

$$\overline{\pi}(dx, k) = \overline{z}^k\overline{\pi}(dx|k) \approx \sum_p c_\ell^p \phi_p(x, k)\overline{z}_\ell^k\overline{\pi}_\ell(dx|k). \tag{5.24}$$

Making the choice $f = \phi_r$ and inserting the basis expansion into (5.23) gives

$$0 = \sum_{kp} c_\ell^p \overline{z}_\ell^k \int \phi_p(x,k) \overline{\pi}_\ell(dx|k) \mathbb{E}_{X^0=x, J^0=k} \left[ \sum_{t=0}^{S(1)-1} \phi_r(X^t, J^t) - \phi_r(X^{t+\tau}, J^{t+\tau}) \right]. \tag{5.25}$$

This is a linear system which can be solved for the expansion coefficients:

$$0 = \sum_p c_\ell^p M_{pr}. \tag{5.26}$$

The basis set $\phi_p$ must include the constant function so that the system has a unique nontrivial solution. We note here that the NEUS algorithm corresponds to the choice

$$\phi_p(x,k) = \mathbb{1}_{\{p\}}(k) \tag{5.27}$$

and $\tau = 1$. We summarize the BAD NEUS algorithm in Algorithm 6.

**Algorithm 6** BAD NEUS

**Require:** Approximate Markov propagator $\tilde{\mathcal{U}}$, Resampling operator $\mathcal{R}_k$, starting flux distributions $\{\overline{\pi}_0(dx|k)\}_{k=1}^n$, initial region weights $\{\overline{z}_0^k\}_{k=1}^n$ with $\sum_k \overline{z}_0^k = 1$, basis set $\{\phi_p(x,k)\}_p$, lag time $\tau$

1: **for** $\ell = 0 \dots L$ **do**
2: $\quad \overline{\pi}_\ell \leftarrow \overline{\pi}_{\ell,h}$
3: $\quad M_{pr} \leftarrow \sum_k \int \overline{z}_\ell^k \phi_p(x,k) \overline{\pi}_\ell(dx|k) \mathbb{E}_{X^0=x,J^0=k} \left[ \sum_{t=S(0)}^{S(1)-1} \phi_r(X^t,J^t) - \phi_r(X^{t+\tau},J^{t+\tau}) \right]$
4: $\quad$ Solve $cM = 0$
5: $\quad \overline{\pi}_\ell(dx,k) \leftarrow \sum c_\ell^p \phi_p(x,k) \overline{z}_\ell^k \overline{\pi}_\ell(dx|k)$
6: $\quad \tilde{\overline{\pi}} \leftarrow \tilde{\mathcal{U}} \overline{\pi}_\ell(dx,j)$
7: $\quad$ **for** $k = 1 \dots n$ **do**
8: $\quad\quad \overline{\pi}_{\ell+1}(dx|k), \overline{z}_{\ell+1}^k \leftarrow \mathcal{R}_k[\tilde{\overline{\pi}}]$
9: $\quad$ **end for**
10: **end for**
11: **return** $\overline{\pi}_L(dx|j)$ and $\overline{z}_L^j$

Consistent with WE and NEUS, in practice, we estimate the integral $M_{pr}$ through a sum over walkers:

$$M_{pr} \approx \sum_{ki} \overline{z}_\ell^k \phi_p(X_i^{S_i(\ell)},k) \mathbb{1}_{\{k\}}(J_i^0) \left[ \sum_{t=S_i(\ell)}^{S_i(\ell+1)-1} (\phi_r(X_i^t,J_i^t) - \phi_r(X_i^{t+\tau},J_i^{t+\tau})) \right] \tag{5.28}$$

Given $M$ and, in turn, the estimated expansion coefficients, we update the weights as

$$w_i^{S_i(\ell)} = \frac{\overline{z}_\ell^{J_i^{S_i(\ell)}}}{Z} \sum_p c_\ell^p \phi_p(X_i^{S_i(\ell)}, J_i^{S_i(\ell)}), \tag{5.29}$$

where $Z$ normalizes the total of the walker weights to one. If the basis set contains the constant function in its span, and the flux distributions $\overline{\pi}_\ell(dx|j)$ and region weights $\overline{z}_\ell^j$ take

their steady-state values, then the linear system (5.25) is solved by $\sum_p c_p \phi(x,j) = 1$, and BAD NEUS has the correct fixed point. To reduce variance in our estimate of the matrix $M$ to ensure a stable solve, it is often useful to work with the mixture distribution for the last $h$ iterations (including the current iteration):

$$\overline{\pi}_{\ell,h}(dx|k) = \frac{1}{h} \sum_{t=\ell-h+1}^{\ell} \overline{\pi}_t(dx|k). \tag{5.30}$$

When working with walkers, this mixture distribution corresponds to concatenating the walkers from the last $h$ iterations, and therefore contains more data than using a single iteration. We may simply substitute $\overline{\pi}_\ell$ for $\overline{\pi}_{\ell,h}$ in all of our algorithms.

Finally, we note that the general strategy of improving the approximation of the change of measure is not limited to using a basis expansion. Just as one can use various means to solve equations of the transition operator that encodes the statistics of the dynamics [35, 156, 220, 221], one can represent the change of measure here by either a basis expansion or a nonlinear representation. In particular, our tests based on the approach of repeatedly approximately applying the propagator using a variational formulation in Ref. 221 show significant promise (unpublished results).

## 5.5 Implementable algorithms

Here we give algorithms written in terms of walkers. These are the routines which we implement directly.

**Algorithm 7** Trajectory Stratification (walker representation)

**Require:** Starting ensemble of walkers $E_0 = \{(X_i^0, J_i^0, w_i^0)\}_{i=1}^N$, with $\sum_{i=1}^N w_i^0 = 1$, number of past iterations to retain $h$, total iterations $L$, number of strata $n$. lag time $\tau$

**for** $\ell = 0 \dots L$ **do**

  $N \leftarrow \text{length}(E_\ell)$

  Generate a list of trajectories

$$T_\ell \leftarrow \{[(X_i^{S_i(\ell)}, J_i^{S_i(\ell)}, w_i^{S_i(\ell)}), (X_i^{S_i(\ell)+1}, J_i^{S_i(\ell)+1}, w_i^{S_i(\ell)}) \dots (X_i^{S_i(\ell+1)+\tau}, J_i^{S_i(\ell+1)+\tau}, w_i^{S_i(\ell)})]\}_{i=1}^N,$$

  where $(X_i^{S_i(\ell)}, J_i^{S_i(\ell)}, w_i^{S_i(\ell)}) = E_\ell[i]$

  $T_\ell \leftarrow \text{concatenate}(\{T_\ell, T_{\ell-1}, \dots T_{\ell-h-1}\})$

  $N \leftarrow \text{length}(T_\ell)$

  renormalize the weights in $T_\ell$ by dividing each by $h$.

  Update weights $\{w_i^{S_i(\ell)}\}_{i=1}^N$ using Algorithm 8.

  Initialize an empty list $E_{\ell+1} = \{\}$

  **for** $k = 1 \dots n$ **do**

    $\overline{z}_{\ell+1}^k \leftarrow \sum_{i=1}^N w_i^{S_i(\ell)} \mathbb{1}_{\{k\}}(J_i^{S_i(\ell+1)})$

    **for** $r = 1 \dots N_k$ **do**

      Sample an index $b$ with probability proportional to $w_b^{S_b(\ell)} \mathbb{1}_{\{k\}}(J_b^{S_b(\ell+1)})$

      Append to $E_\ell$ the configuration $(X_b^{S_b(\ell+1)}, J_b^{S_b(\ell+1)}, \overline{z}_{\ell+1}^k / N_k)$

    **end for**

  **end for**

**end for**

**return** $T_L$ and $\overline{z}_L^k$

**Algorithm 8** Approximating the steady-state flux distribution (walker representation)

**Require:** List of trajectories $T_\ell$, lag time $\tau$, basis functions $\phi_p$, bin weights $\overline{z}_\ell$

$M_{pr} \leftarrow \sum_{ki} \overline{z}_\ell^k \phi_p(X_i^{S_i(\ell)}, k) \mathbb{1}_{\{k\}}(J_i^0) \left[\sum_{t=S_i(\ell)}^{S_i(\ell+1)-1} (\phi_r(X_i^t, J_i^t) - \phi_r(X_i^{t+\tau}, J_i^{t+\tau}))\right]$

Solve $c_\ell M = 0$

$w_i^{S(\ell)} \leftarrow (\overline{z}_\ell^{J_i^{S_i(\ell)}} / Z) \sum_p c_\ell^p \phi_p \left(X_i^{S_i(\ell)}, J_i^{S_i(\ell)}\right)$

**return** $\{w_i^{S_i(\ell)}\}_{i=1}^N$

## 5.6 Two-dimensional test system

We first illustrate our approach by sampling a two-dimensional system, which enables us to compare estimates of the steady-state distribution to the Boltzmann probability and to run the simulations until convergence. Specifically, the system is defined by the Müller-Brown potential [2], which is a sum of four Gaussian functions:

$$V(u,v) = \frac{1}{20}\sum_{i=1}^{4} C_i \exp[a_i(u-v_i)^2 + b_i(u-u_i)(v-v_i) + c_i(v-v_i)^2]. \quad (5.31)$$

For all results shown, we use $C_i = \{-200, -100, -170, 15\}$, $a_i = \{-1, -1, -6.5, 0.7\}$, $b_i = \{0, 0, 11, 0.6\}$, $c_i = \{-10, -10, -6.5, 0.7\}$, $u_i = \{1, -0.27, -0.5, -1\}$, $v_i = \{0, 0.5, 1.5, 1\}$. The potential is shown in Figure 5.2.

We evolve the system with overdamped Langevin dynamics, discretized with the BAOAB algorithm [137]:

$$X^{t+dt} = X^t - \nabla V(X^t)dt + \sqrt{\frac{dt}{2\beta}}(Z^t + Z^{t-dt}), \quad (5.32)$$

where $dt$ is the time step, $\beta$ is the inverse temperature, and $Z_t \sim N(0, I_2)$ is a random vector drawn with components drawn from the normal distribution with zero mean and unit standard deviation. For all results shown, we use a time step of $dt = 0.001$ We use $\beta = 2$. The time until, $X_t$ makes a transition between the local minima of $V$ is exponential in beta. For our rate calculations, we define states A and B, indicated by the orange and red ovals in figure 5.2 respectively. These states are defined as:

$$\begin{aligned} A &= \{y, v : 6.5(u+0.5)^2 - 11(u+0.5)(v-1.5) + 6.5(u-1.5)^2 < 0.3\} \\ B &= \{u, v : (u-0.6)^2 + 0.5(v-0.02)^2 < 0.2\}. \end{aligned} \quad (5.33)$$

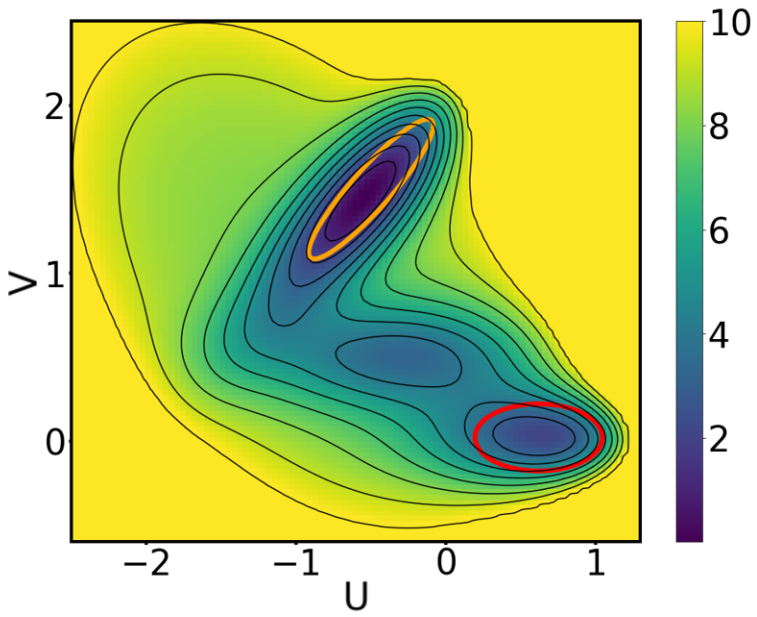


Figure 5.1: Müller-Brown potential. Orange and red ovals indicate the states referenced above.

### *5.6.1 Comparison of algorithms*

A key point of our theoretical development is that NEUS and BAD-NEUS are simple elaborations of WE. This enables us to define a unified walker-based algorithm that we use in practice (Algorithm 7, with the added operations needed for NEUS and BAD-NEUS in Algorithm 8). Splitting and stratification are defined through the rule for switching the index process. The sequence of stopping times $S(\ell)$ is then determined by the index process through (5.3). The update rule for $J_t$ that we employ for all three algorithms is as follows. Let $\{\psi_k(x)\}_{k=1,\dots,n}$ be a set of nonnegative functions. If $\psi_{J^t}(X^{t+dt}) > 0$, then $J^{t+dt} = J^t$; otherwise, $\mathbb{P}[J^{t+dt} = k] = \psi_k(X^{t+dt}) / \sum_k \psi_k(X^{t+dt})$. That is, the value of the index process remains the same until the walker leaves the support of $\psi_{J^t}$, and then a new index value

$k$ is drawn with likelihood proportional to $\psi_k(X^{t+dt})$. For the Müller-Brown model, we use

$$\psi_k(u,v) = \begin{cases} 1 & \text{if } |v - v_k| < \varepsilon_k \text{ and } 1 < k < n \\ 1 & \text{if } v - v_k < \varepsilon_k \text{ and } k = 1 \\ 1 & \text{if } v_k - v > \varepsilon_k \text{ and } k = n \\ 0 & \text{otherwise.} \end{cases} \tag{5.34}$$

Unless otherwise indicated, we set $n = 10$, space the $v_k$ uniformly in the interval $[-0.2, 1.8]$, and set $\varepsilon_k = 0.6(v_{k+1} - v_k)$, so that the regions of support (strata) overlap. This choice prevents walkers in barrier regions from rapidly switching back and forth between index values, limiting the overhead of the algorithms. Unless otherwise indicated, we use 2000 walkers per region and run them until their index processes switch values. Initial configurations for $J^0 = k$ are generated by uniformly sampling the support of $\psi_k$.

For BAD-NEUS, we use a basis set consisting of 10 indicator functions per stratum. The indicator functions are determined by clustering all the samples in a stratum with $k$-means clustering and partitioning it into Voronoi polyhedra based on the cluster means. Unless otherwise indicated, we compute statistics using data from the last $h = 3$ iterations and use a lag time of $\tau = 10$ time steps.

We measure convergence by computing the root mean square (RMS) difference in $\ln(\tilde{\pi}(x))$ from $-\beta V$, where $\tilde{\pi}(x)$ is an estimate of the steady-state distribution from the normalized histogram of samples. For the histogram, we use a uniform $50 \times 50$ grid on the rectangle defined by the minimum and maximum values in the NEUS dataset. Since some grid regions are empty because they correspond to energies too high for NEUS to sample, we only compute errors over bins with $V < 7$. We stop each simulation when the RMS difference drops below one. Results are shown in Figure 5.2. We see that WE requires about 73 times more iterations than NEUS, which in turn requires about 13 times more iterations than BAD-NEUS to reach

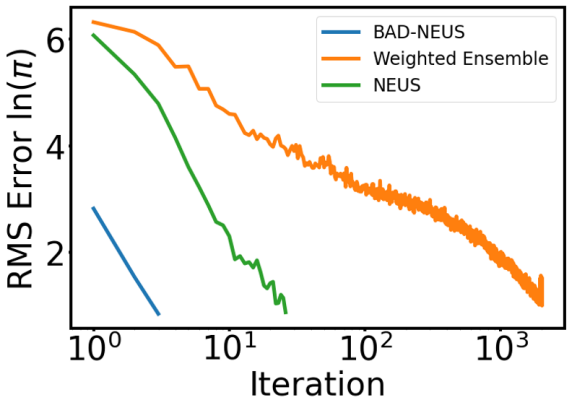


Figure 5.2: Convergence of the steady-state distribution of the Müller-Brown system for different algorithms as indicated. For these simulations, there are 2000 walkers per stratum, $h = 3$ iterations are retained, and the lag time is $\tau = 10$ time steps.

the convergence criterion.

### *5.6.2 Choice of hyperparameters*

Having demonstrated that trajectory stratification outperforms WE for this system, we now examine the effect of key hyperparameters in NEUS and BAD-NEUS.

First, we investigate the effect of the number of strata. To compare simulations that require essentially the same resources, we hold the total number of walkers fixed at 40000, and we divide the walkers uniformly across the strata. For these simulations, we use a lag time of $\tau = 1$ and retain $h = 3$ past iterations. In Figure 5.3 we plot both the number of iterations to reach convergence and the total number of time steps needed for convergence as we vary the number of strata. We see that the number of BAD-NEUS iterations required for convergence increases linearly with the number of strata. However, the lengths of trajectories decrease because the strata are spaced more closely. Due to this interplay, the total effort to achieve convergence decreases until about 12 strata and then levels off. While actual performance will depend on the overhead incurred by stopping and starting the dynamics engine and computing the BAD-NEUS weights, these results indicate that finer stratification

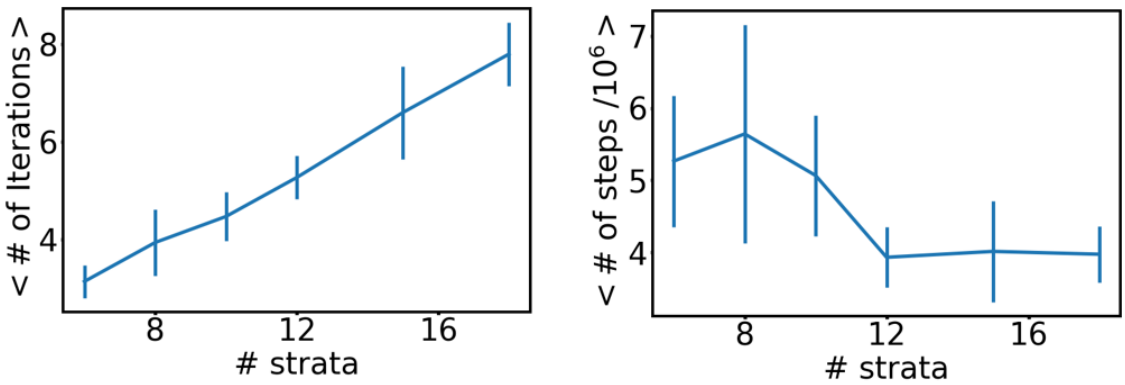


Figure 5.3: Effect of the number of strata on convergence of BAD-NEUS applied to the Müller-Brown system. The number of iterations required (top) and the total number of time steps (bottom)—a proxy for computational effort—until convergence. The total number of walkers is held fixed. Error bars are the standard deviation over 10 replicate simulations. For these simulations, there are 40000 walkers total, $h = 3$ iterations are retained, and the lag time is $\tau = 1$ time step.

is better.

Next, we examine the dependence of convergence on the number of iterations used to compute statistics, $h$ (other hyperparameters are set to the default values given above). Using data from a larger number of iterations allows for more averaging, but it can also contaminate the statistics with samples obtained before the steady-state distribution has converged. While the trends are clearer for NEUS than BAD-NEUS because the former converges more slowly, Figure 5.4 indicates that retaining fewer iterations is better for both NEUS and BAD-NEUS. It is therefore important to use enough walkers per iteration that data from past iterations need not be retained. If one has access to a large number of processing elements, and many walkers can be simulated in parallel, this is not a severe restriction. Once convergence is achieved one can begin accumulating data to reduce variance.

Finally, we investigate the impact of the lag time in Figure 5.5 (other hyperparameters are set to the default values given above). We find that using longer lag times weakly decreases the number of iterations required for convergence.

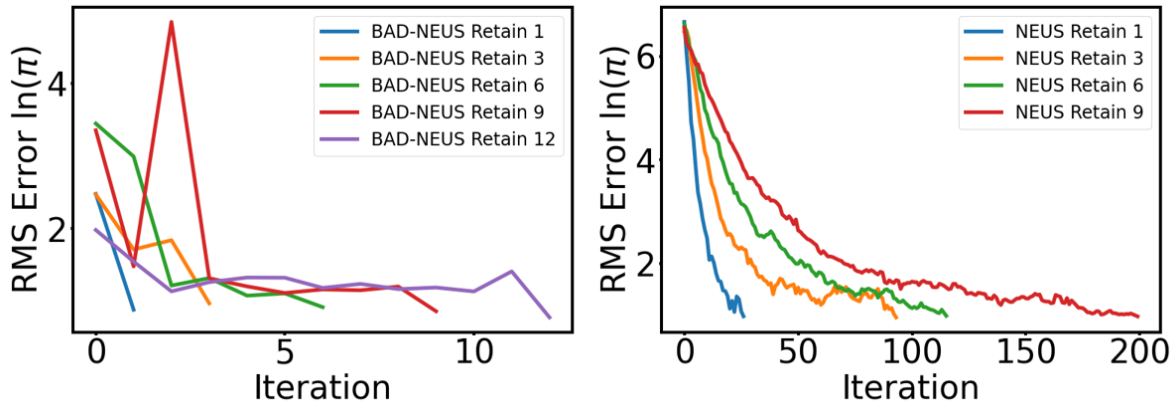


Figure 5.4: Effect of retaining past iterations on convergence of NEUS (top) and BAD-NEUS (bottom) applied to the Müller-Brown system. For these simulations, there are 2000 walkers per stratum, and the lag time is $\tau = 10$ time steps.

### *5.6.3 Kinetic statistics*

A common problem in molecular dynamics is that while equilibrium averages are relatively straightforward to calculate from biased simulations (via, for example, the various methods reviewed in Ref. 222), dynamical averages are much harder. In this section, we use BAD-NEUS to compute dynamical averages efficiently. Specifically, we consider the backward committor, $q_-$, which is the probability that the system last came from a reactant state $A$ rather than a product state $B$, and the transition path theory (TPT) rate, defined as the

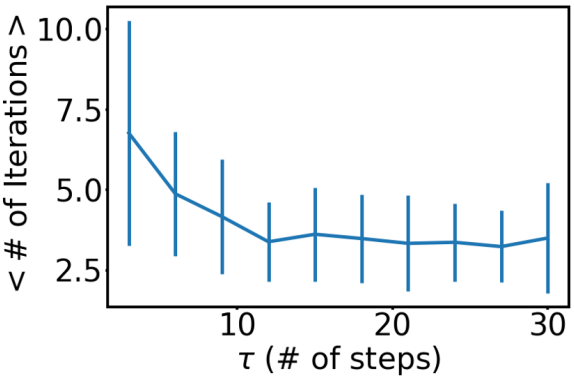


Figure 5.5: Dependence of convergence on the lag time for BAD-NEUS applied to the Müller-Brown system. For these simulations, there are 2000 walkers per stratum, and $h = 3$ iterations are retained.

mean number of $A$ to $B$ transitions per unit time divided by the fraction of time spent last in $A$. Both of the systems that we consider are microscopically reversible, so $q_- = 1 - q_+$, where $q_+$ is the forward committor, the probability that the system next goes to state $B$ rather than state $A$.

As shown previously [223, 214, 219], these quantities can be obtained by splitting the ensemble of transition paths. Here, we consider the following history dependent stratification. Let $\{\psi_k^A(x)\}_{k=1,\ldots,n}$ and $\{\psi_k^B(x)\}_{k=1,\ldots,m}$ be non-negative functions, let $\psi_k = \psi_k^A$ if $k \leq n$; otherwise, let $\psi_k = \psi_{k-n}^B$, and let $\mathbb{1}_A$ and $\mathbb{1}_B$ be indicator functions on sets $A$ and $B$, respectively. Let $T^-(t+dt)$ be the time the system was last in $A$ or $B$, so that $\mathbb{1}_A(T^-(t+dt)) = 1 - \mathbb{1}_B(T^-(t+dt))$ reports whether $A$ rather than $B$ was last visited. Define the index process by the following update rule:

$$\mathbb{P}[J_{t+dt} = k] = \begin{cases} \psi_k(X^{t+dt})\mathbb{1}_A(T^-(t+dt))/\sum_i \psi_i^A(X^{t+dt}) & k \leq n, \psi_{J^t}(X^{t+dt}) = 0 \text{ or} \\ & \mathbb{1}_A(T^-(t+dt)) \neq \mathbb{1}_A(T^-(t)) \\ \psi_k(X^{t+dt})\mathbb{1}_B(T^-(t+dt))/\sum_i \psi_i^B(X^{t+dt}) & k > n, \psi_{J^t}(X^{t+dt}) = 0 \text{ or} \\ & \mathbb{1}_A(T^-(t+dt)) \neq \mathbb{1}_A(T^-(t)) \\ \delta_{J^t k} & \text{otherwise} \end{cases} \tag{5.35}$$

where $\delta_{J^t k}$ is the Kronecker delta function.

Thus there are $n+m$ total strata, the first $n$ of which contain walkers that last visited $A$, and the remainder of which contain walkers that last visited $B$. The steady-state distribution of walkers which last visited $A$ is proportional to $\pi(x)q_-(x)$, and that which last visited $B$ is proportional to $\pi(1 - q_-(x))$. This ensemble lets us compute dynamical averages. The

backward committor projected onto a space of CVs can be computed using

$$q_-^{\theta}(s) = \frac{\mathbb{E}[\mathbb{1}_{\{J^t \leq n\}} \mathbb{1}_{\{\theta(X^t) \in ds\}}]}{\mathbb{E}[\mathbb{1}_{\{\theta(X^t) \in ds\}}]}, \tag{5.36}$$

where $\theta$ is a vector with components that are the CVs and $ds$ is a bin in the space. We obtain the TPT rate constant

$$k_{AB} = \frac{\mathbb{E}_\pi[q_-(x)\mathbb{1}_{B^c}(x)\mathcal{T}_1\mathbb{1}_B(x)]}{\mathbb{E}_\pi[q_-(x)]} \tag{5.37}$$

by choosing

$$g(X^0, J^0 ..., X^t, J^t) = \mathbb{1}_{B^c}(X^0)\mathbb{1}_B(X^1)\mathbb{1}_{\{J^0 < n\}} \tag{5.38}$$

in (5.14) for the numerator and

$$g(X^0, J^0 ..., X^t, J^t) = \mathbb{1}_{\{J^0 < n\}} \tag{5.39}$$

for the denominator. This corresponds to the steady-state flux into $B$ from trajectories that originated in $A$.

For the Müller-Brown system, we define states $A$ and $B$ as

$$\begin{aligned} A &= \{y, v : 6.5(u + 0.5)^2 - 11(u + 0.5)(v - 1.5) + 6.5(u - 1.5)^2 < 0.3\} \\ B &= \{u, v : (u - 0.6)^2 + 0.5(v - 0.02)^2 < 0.2\}. \end{aligned} \tag{5.40}$$

We use a similar index process construction as for the equilibrium calculations, except there are 10 $v_k$ that are evenly spaced in the interval $[0, 1.6]$ for $\psi_k^A$ and 10 $v_k$ that are evenly spaced in the interval $[-0.3, 0.8]$ for $\psi_k^B$, for a total of 20 strata. We use different definitions for $\psi_k^A$ and $\psi_k^B$ because, for walkers originating from state $A$ ($B$), a stratum located below (above) state $B$ ($A$) is unlikely be populated. Both NEUS and BAD-NEUS use 2000 walkers

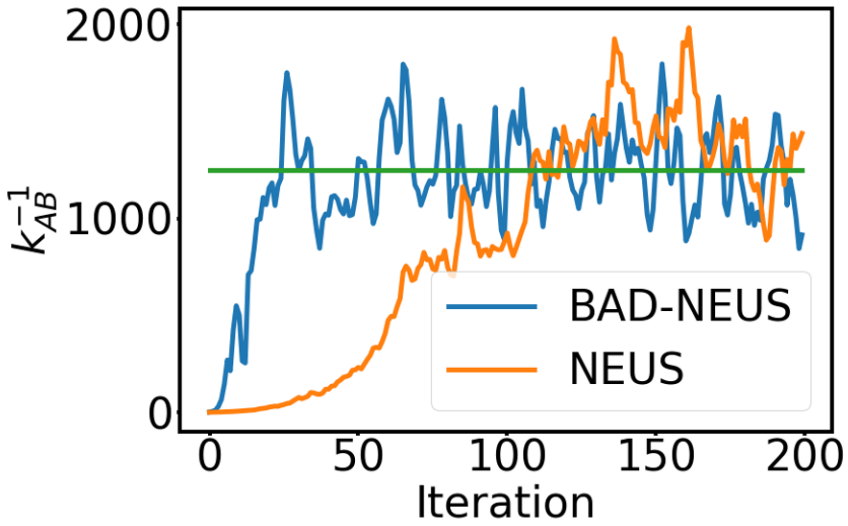


Figure 5.6: Convergence of the TPT inverse rate estimate for the Müller-Brown system. The green line is the grid-based reference obtained from 5.41. For these simulations, there are 2000 walkers per stratum, $h = 4$ iterations are retained, and the lag time is $\tau = 10$ time steps.

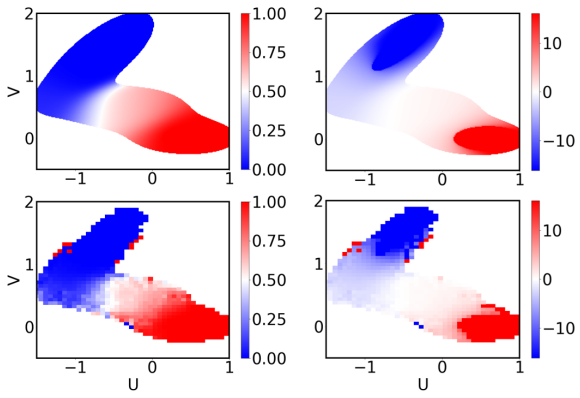


Figure 5.7: Illustration of the backward committor computed from BAD-NEUS. (top left) Reference backward committor. (top right) Reference $\ln(q_-/(1-q_-))$. (bottom left) BAD-NEUS backward committor at iteration 20 computed from (5.36). (bottom right) BAD-NEUS $\ln(q_-/(1-q_-))$, showing that BAD-NEUS effectively computes backward committors close to states $A$ and $B$. These results are obtained from the same simulations as Fig. 5.6.

per stratum, and we compute statistics using data from the last $h = 3$ iterations. For BAD-NEUS, we use a basis set consisting of 10 indicator functions per stratum, again based on $k$-means clustering; we use a lag time of $\tau = 1$.

To compute reference values for these kinetic statistics, we use the grid-based approxi-

mation to the generator in Ref. 221, with the same grid parameters. Since the dynamics are microscopically reversible, we obtain the backward committor by solving for the forward committor using the approach in Ref. 221, then setting $q_- = 1 - q_+$. We solve for the reaction rate using

$$k_{AB} = \frac{2(\vec{\pi q}_-)^\top (P\vec{\mathbb{1}}_B)}{(\vec{\pi}^\top \vec{q}_-)(\beta\epsilon_x)}, \tag{5.41}$$

where $P$ is the discretized transition matrix defined on the grid, $\epsilon_x$ is the grid spacing, and arrows indicate vectors of function values on the grid points ($\vec{\pi q}_-$ is a single vector of product values).

Figure 5.6 shows estimates for the TPT rate. We see that BAD-NEUS converges several fold faster than NEUS. Each BAD-NEUS iteration requires an average of 407 time units of sampling (arising from 2000 walkers in each of 20 strata, with an average time before leaving the stratum of 10.2 time steps). Each iteration is therefore significantly less total computational cost than generating a single $A$ to $B$ transition on average (1200 time units, as evidenced by $k_{AB}^{-1}$) and is amenable to parallelization.

Figure 5.7 shows the backward committor computed at BAD-NEUS iteration 20 using (5.36) with the variables of numerical integration as the CVs. We represent the results in two ways: the committor itself and its logit function. The former emphasizes the transition region ($q_- \approx 0.5$) while the latter emphasizes values close to states $A$ and $B$ ($q_- \approx 0$ and $q_- \approx 1$). Both show excellent agreement with the reference.

## 5.7 Conclusions

We have introduced a new trajectory stratification scheme which converges faster than existing methods. We show that our scheme is a natural generalization of the weighted ensemble scheme originally introduced by Huber and Kim [20]. The key modification we introduce is that we insert an approximation to the stationary distribution before the resampling step.

This approximation algorithm is set up such that it preserves the stationary distribution of the dynamics, and therefore does not introduce systematic errors into the algorithm in the large-data limit. We show that this simple modification accelerates the convergence of statistics for the Müller-Brown model by several orders of magnitude.

Our work leaves open several possible extensions and generalizations. Our algorithm can be easily modified by replacing our basis expansion approximation by another approximation to the stationary distribution. Examples include machine learning methods [173, 221] and basis expansions which include memory effects [197]. These approximations alleviate the need to identify basis sets which describe the dynamics well.

## 5.8 Appendix

### *5.8.1 Derivation of* (5.14)

(5.14) Here we derive (5.14) by modifying the proof from Ref. 224. Let $Z^t$ be a markov process, and let $S^n = \sum_{t=0}^{n-1} g(Z^t)$ be the partial sum of a function $g$. Let $\omega(z)$ satisfy the linear problem:

$$(\mathcal{T}^1 - \mathcal{I})\omega(z) = -(g(z) - \mathbb{E}_\pi[g(z)]) \tag{5.42}$$

subject to the constraint $\mathbb{E}_\pi[\omega] = 0$. Where the forward transition operator is defined as:

$$\mathcal{T}^t g(z) = \mathbb{E}[g(Z^t)|Z^0 = z] \tag{5.43}$$

Consider the process:

$$U^n = S^n - \mathbb{E}_\pi[g]n + \omega(Z^n) \tag{5.44}$$

Then $U^t$ is a martingale with respect to $Z^0, ..., Z^t$. To see this, we compute:

$$\mathbb{E}[U^{t+1} - U^t|Z^0, ...Z^t] = \mathbb{E}[S^{t+1} - \mathbb{E}_\pi[g](t+1) + \omega(Z^{t+1}) - \Big(S^t - \mathbb{E}_\pi[g]t + \omega(Z^t)\Big)\,|Z^0, ...Z^t]$$

$$= \mathbb{E}[g(Z^t) - \mathbb{E}_\pi[g] + \omega(Z^{t+1}) - \omega(Z^t)|Z^0, ...Z^t]$$

$$= g(Z^t) - \mathbb{E}_\pi[g] + \mathbb{E}[\omega(Z^{t+1}) - \omega(Z^t)|Z^0, ...Z^t]$$

$$= g(Z^t) - \mathbb{E}_\pi[g] + (\mathcal{T}^1 - \mathcal{I})\omega(Z^t)$$

$$= g(Z^t) - \mathbb{E}_\pi[g] - g(Z^t) + \mathbb{E}_\pi[g] = 0$$

We refer the reader to 224 for a more technical discussion and conditions under which the relevant expectations are bounded such that we may apply the optional stopping theorem. Then, for any stopping time $T$ with $\mathbb{E}[T] < \infty$:

$$0 = \mathbb{E}[U^T - U^0] = \mathbb{E}[S^T] - \mathbb{E}_\pi[g]\,\mathbb{E}[T] + \mathbb{E}[\omega(Z^T)] - \mathbb{E}[\omega(Z^0)] \tag{5.45}$$

Which yields the result:

$$\frac{\mathbb{E}[\sum_{t=0}^{T-1} g(Z^t)]}{\mathbb{E}[T]} = \mathbb{E}_\pi[g] + \frac{\mathbb{E}[\omega(Z^0)] - \mathbb{E}[\omega(Z^T)]}{\mathbb{E}[T]} \tag{5.46}$$

Now take $Z^t = (X^t, J^t, ..., X^{t+\tau}, J^{t+\tau})$ and take the stopping time to be $S(1)$. In the case where $X^0, J^0$ is distributed according to the steady state flux distribution $\overline{\pi}(dx, k)$, (5.7) implies that $X^T, J^T \sim \overline{\pi}(dx, k)$, and hence the distribution of $Z^0$ and $Z^T$ are the same, and so the residual term in (5.46) is zero. In this case, we note that the distribution of $X^0, J^0$ is given by:

$$\pi(dx) = P[X^0 \in dx] \tag{5.47}$$

$$= \sum_k P[X^0 \in dx \mid J^0 = k]P[J^0 = k] \tag{5.48}$$

$$= \sum_k \overline{z}^k \overline{\pi}(dx|k). \tag{5.49}$$

The joint distribution for $(X, J)$ is $\overline{\pi}(dx, k) = \overline{z}^k \overline{\pi}(dx|k)$. Therefore,

$$\mathbb{E}\left[\sum_{t=0}^{S(1)-1} g(X^t, J^t, ..., X^{t+\tau}, J^{t+\tau})\right] =$$

$$\sum_k \overline{z}^k \int \overline{\pi}(dx|k)\mathbb{E}_{X^0=x,J^0=k}\left[\sum_{t=0}^{S(1)-1} g(X^t, J^t, ..., X^{t+\tau}, J^{t+\tau})\right] \quad (5.50)$$

and

$$\mathbb{E}[S(1)] = \sum_k \overline{z}^k \int \overline{\pi}(dx|k)\mathbb{E}_{X^0=x,J^0=k}[S(1)] \quad (5.51)$$

Substituting these into (5.46) and remembering that the residual term is zero yields (5.14).

# CHAPTER 6

# APPLICATION OF TRAJECTORY STRATIFICATION TO NTL9

## 6.1 Introduction

In this chapter, we apply BAD NEUS to a realistic model of a molecular system. Our test molecule is the 56-residue N-terminal domain of the ribosomal L9 protein, known as NTL9. The native secondary structure is $\beta\beta\alpha\beta\alpha$ sequentially; in the native state, the three $\beta$-strands form an anti-parallel $\beta$-sheet, and the two $\alpha$-helices pack on either side of it (Fig. 6.1). The reversible folding of NTL9 has been extensively studied both experimentally and computationally [5, 225, 226]. In addition, the timescale of folding is on the order of a millisecond which is at the upper limit of what may be accessed with traditional molecular dynamics simulations. For these reasons, the NTL9 peptide is a good test case for our new method.

We use our study of NTL9 to illustrate a general workflow for examining reversible molecular rare events. The outline of this workflow is as follows. We biased our system away from the stable folded state to generate initial structures for BAD NEUS sampling. We found that this procedure populated all the strata at the start of the simulation. Next, we select a set of CVs which we suspect describe the folding mechanism, and we use these to construct a basis set for BAD NEUS.

We also illustrate how BAD NEUS results may be used for mechanistic analysis. One of the key features of BAD NEUS is that it naturally yields both the TPT rate for folding and unfolding, as well as the backward committor and stationary distribution. We therefore show how one can use logistic regression to extract the backward committor from simulation data, and then use the backward committor as a CV for visualization, and to extract transition state structures. Finally, we summarize our conclusions and present future research directions and applications.

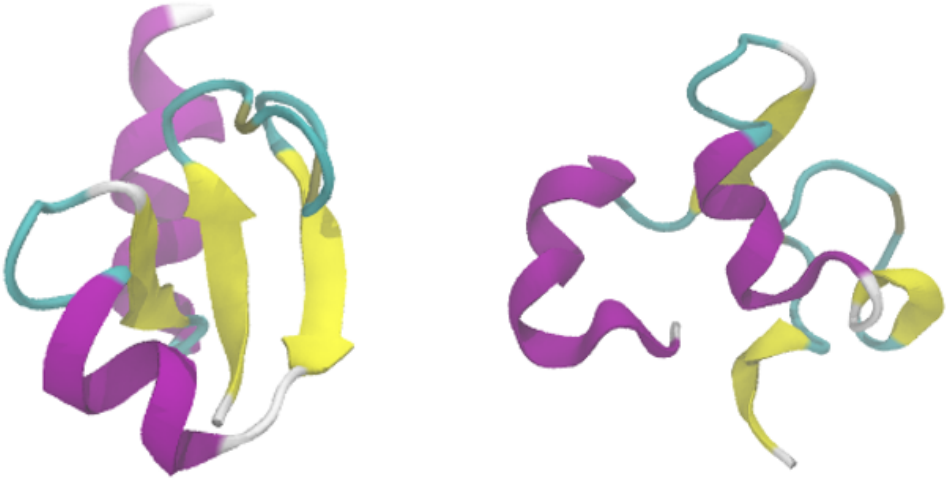

Figure 6.1: Representative native and melted NTL9 structures. Native structure is the crystal structure, and the melted structure is taken from the BAD NEUS simulation.

## 6.2 System setup and Molecular Dynamics Details

All MD simulations were performed with the OpenMM engine [194]. Since our algorithm only requires us to stop and start the MD engine while running unbiased dynamics, we do not need to modify OpenMM in any way. To enable direct comparison to previous work on this system [5], we model NTL9 using the AMBER FF14SB force field [227] and the Hawkins, Cramer and Truhlar generalized Born implicit solvent model [228, 229]. We use a Langevin thermostat with a timestep of 2 fs, a temperature of 303.15 K, and a friction constant of $\gamma = 80$ ps$^{-1}$, which corresponds to the high-viscosity case considered in Ref. 5. We characterize the structure with the full root mean square deviation in atom positions (RMSD) and the fraction of native contacts ($Q$). We define the folded state as a configuration having full RMSD) $< 0.28$ nm and $Q > 0.85$. The unfolded state has full RMSD $> 0.80$ nm and $Q < 0.35$.

The stratification is similar to that for the Müller-Brown in the previous chapter. Let $\{\psi_{Ak}(x)\}_{k=1,\ldots,n}$ and $\{\psi_{Bk}(x)\}_{k=1,\ldots,m}$ be non-negative partitions of unity, let $\psi_k = \psi_{Ak}$ if

$k \leq n$ otherwise let $\psi_k = \psi_{Bk-n}$, and let $\mathbb{1}_A$ and $\mathbb{1}_B$ be the indicator function on a sets $A$ and $B$. Let $\mathbb{1}_A(T^-_{A\cup B}(t+dt))$ be the augmented variable that tracks whether a trajectory last visited $A$ rather than $B$. Define the index process by the following update rule:

$$P[J_{t+dt} = k] = \begin{cases} \psi_k(X_{t+dt})\mathbb{1}_A(T^-_{A\cup B}(t+dt)) & k \leq n, \psi_{J_t}(X_{t+dt}) = 0 \\ & \text{or } \mathbb{1}_A(T^-_{A\cup B}(t+dt)) \neq \mathbb{1}_A(T^-_{A\cup B}(t)) \\ \psi_k(X_{t+dt})(1 - \mathbb{1}_A(T^-_{A\cup B}(t+dt))) & k > n, \psi_{J_t}(X_{t+dt}) = 0 \\ & \text{or } \mathbb{1}_A(T^-_{A\cup B}(t+dt)) \neq \mathbb{1}_A(T^-_{A\cup B}(t)) \\ \delta_{J_t k} & \text{otherwise} \end{cases} \tag{6.1}$$

Thus there are $n + m$ total strata, the first $n$ of which contain walkers which have last visited $A$, and the remainder of which have last visited $B$. The upshot of this is that the steady state law of walkers which have last visited $A$ is proportional to $\pi(x)q_-(x)$, where $q_-$ is the backward committor probability. For the NTL9 system, we set

$$\psi_{Ak}(X) = \psi_{Bk}(X) = \begin{cases} 1 & \text{if } |Q(X) - Q_k| < \delta_k \\ 0 & \text{otherwise,} \end{cases} \tag{6.2}$$

where $Q(X)$ is the fraction of native contacts using the definition of Ref. 230. We determine the native contacts from the crystal structure, PDB ID 2HBB [231]. We space 20 $Q_k$ uniformly in the interval $[0.35, 0.85]$ and set $\delta_k = 0.6(Q_{k+1} - Q_k)$, so that the regions overlap. Thus there are 40 total strata.

To initialize the simulation, we first denature the protein by simulating for 80 ns at 380 K starting from the crystal structure and saving every 1.0 ps. We sample 4000 frames in each of the 40 strata from this trajectory and weight them uniformly; this set forms our starting walkers, $\{X^i_0, J^i_0, 1/N\}_{i=1}^N$, where $N = 40$. We use a lag time of 10 ps, and we retain $h = 4$

past ensembles. The initialization pipeline, integrator settings, and stratification choice are the same between NEUS and BAD NEUS.

To construct the basis set for BAD-NEUS, we define the following eight CVs: (1) the fraction of native contacts, (2) the full backbone RMSD, (3-5) the backbone RMSD for each of the three $\beta$-strands individually (residues 1-5, 16-20, and 36-39), (6-7) the backbone RMSD for the two $\alpha$-helices individually (residues 23-33, and 41-51), and (8) the backbone RMSD for the three $\beta$-strands together. Our basis set consists of 10 indicator functions on each stratum constructed by $k$-means clustering on the 8 dimensional CV space. At each iteration (i.e., one pass through the outer loop in Algorithm 7 of Chapter 5), we form our basis set by taking the cluster centers from the prior iteration's basis set and refining them with 10 iterations of Lloyd's algorithm [232].

## 6.3 Extraction of the Backward Committor by Logistic Regression

Our trajectory stratification algorithm, using the history dependent stratification scheme, allows us to compute expectations against the committor of the form:

$$\mathbb{E}_\pi[f(x)q_-(x)]$$

for a given function $f$. In Section 5.6.3 we showed how to use expectations of this form to project the backward committor onto a CV space of interest. However, to use the backward committor itself as a collective variable, we need the value of the backward committor at each point in the dataset. For example, to compute the PMF along the backward committor, one needs to evaluate:

$$\rho(s) = -k_b T \log \left[ \sum_i w_i \mathbb{1}_{|q_-(X_i)-s|<ds/2} \right]. \tag{6.3}$$

However, the fact that $q_-$ appears by itself prevents this sum from being written in the form of an expectation over $\pi q_-$.

Since NEUS returns a change of measure to the distribution $\pi q_-$, it is not possible to extract $q_-$ pointwise with out approximation, even though expectations against $\pi q_-$ can be computed without bias. However, it is possible to extract the backward committor by regression. To do this, suppose we have a parameterized representation of the backward committor with parameters $\vec{\beta}$, and recall that the logistic regression loss function for the backward committor may be written in the form:

$$L(\vec{\beta}) = -\mathbb{E}_\mu \left[ \mathbb{1}_A(X^{T^-(0)}) \log(q_-^{\vec{\beta}}) \right] - \mathbb{E}_\mu \left[ \mathbb{1}_B(X^{T^-(0)}) \log(1 - q_-^{\vec{\beta}}) \right] \tag{6.4}$$

Since

$$q_-(x) = \mathbb{E}_{X^0=x} \left[ \mathbb{1}_A(X^{T^-(0)}) \right] \tag{6.5}$$

and

$$1 - q_-(x) = \mathbb{E}_{X^0=x} \left[ \mathbb{1}_B(X^{T^-(0)}) \right] \tag{6.6}$$

we have

$$L(\vec{\beta}) = -\mathbb{E}_\mu \left[ q_-(x) \log(q_-^{\vec{\beta}}(x)) \right] - \mathbb{E}_\mu \left[ (1 - q_-(x)) \log(1 - q_-^{\vec{\beta}}(x)) \right]. \tag{6.7}$$

If we make the choice $\mu = \pi$, the loss function (6.7) is an expectation we can compute from NEUS using (5.14) with the choices:

$$f(x, j) = \log(q_-^{\vec{\beta}}(x)) \mathbb{1}_{j \leq n} \tag{6.8}$$

and

$$f(x, j) = \log(1 - q_-^{\vec{\beta}}(x)) \mathbb{1}_{j > n} \tag{6.9}$$

for the first and second terms, respectively.

In this chapter, we use a small collection (8-12) of physical CVs, and represent the backward committor using a sigmoid composed with a linear expansion:

$$q_-^{\vec{\beta}}(x) = \frac{1}{1 + \exp\left(-\vec{\beta} \cdot \vec{\theta}(x)\right)} \tag{6.10}$$

where $\vec{\theta}(x)$ is the vector of CV functions. We perform all of our optimizations using the Nelder-Mead simplex algorithm. In many cases, it may be advantageous to use a more flexible representation with a larger input feature space and perform the optimization with gradient based methods, however we leave this complication to future work.

## 6.4 Mechanism of NTL9 Folding

Before discussing the mechanism, we verify covergence by examining the inverse folding rate. Figure 6.2 shows the results of the BAD-NEUS inverse rate, the NEUS inverse rate, and the haMSM accelerated weighted ensemble (haMSM-WE) inverse rate reported in Ref. 5. We find that the BAD-NEUS estimate is within the 95% credibility interval of the haMSM-WE estimate, while the NEUS estimate is relatively low. The BAD-NEUS estimate is slightly higher than the haMSM-WE estimate; while we do not have a true reference, we expect it is more accurate as in both Ref. 5 and the present work, methods tend to approach the folding time from below. Both BAD-NEUS and haMSM-WE required significantly less aggregate simulation than the time needed to see a single folding event, which is about 1 ms. The haMSM-WE calculation used an aggregate simulation time of 252 $\mu$s, while the BAD-NEUS calculation used an average of 4 $\mu$s per iteration, for a total of about 160 $\mu$s. Not only does BAD-NEUS require less sampling, as shown for the Müller-Brown system in the previous chapter, we can compute statistics in addition to the rate, such as the logarithm of the steady-state distribution (the potential of mean force) and backward committor.

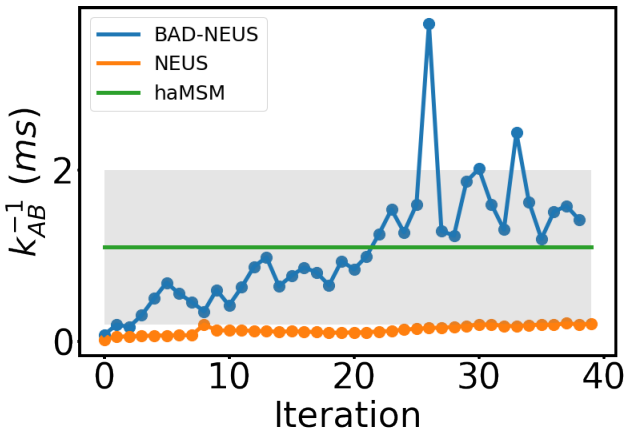


Figure 6.2: Comparison of NEUS, BAD-NEUS and the haMSM-WE [5] estimates for the average time for NTL9 folding. The shaded area shows the Bayesian 95% credibility region reported in Ref. [5].

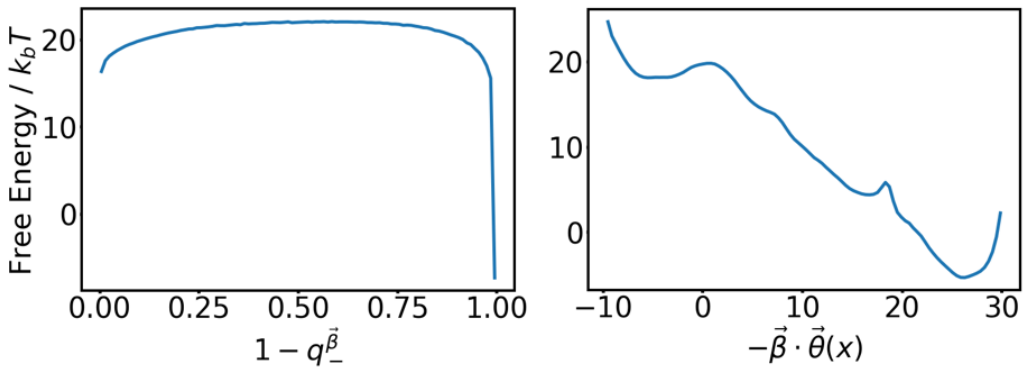


Figure 6.3: Potential of mean force on the regressed backward committor (top), and the regressed linear function that is composed with a sigmoid to yield the backward committor, $-\vec{\beta} \cdot \vec{\theta}(x)$.

We next move to a detailed analysis of the folding mechanism of NTL9. As outlined in Section 6.3, we extract the backward committor using logistic regression. This enables us to use both $q_-^{\beta}$ and the function $-\vec{\beta} \cdot \vec{\theta}(x)$ as CVs; we show the PMFs for these CVs in Figure 6.3. The PMF for the backward committor has a generic shape for a two-state reaction. However, examining the PMF for $-\vec{\beta} \cdot \vec{\theta}(x)$ reveals a second metastable well at backward committor values close to zero (or equivalently forward committor values close to

one, as plotted). It is important to note that the backward committor, often used as the ideal reaction coordinate, contains nearly the same information as the function $-\vec{\beta} \cdot \vec{\theta}(x)$, since the two are related by an monotone function. The latter CV 'stretches out' the x-axis near backward committor values close to 0 or 1. This allows us to resolve details in the PMF which are not visible if we visualize using the backward committor. We therefore focus on $-\vec{\beta} \cdot \vec{\theta}(x)$ for further analysis.

To interpret the PMFs, we examine representative structures from the native state, a melted structure, the barrier located at $-\vec{\beta} \cdot \vec{\theta}(x) = 0.99$, and the intermediate basin located at $-\vec{\beta} \cdot \vec{\theta}(x) = 16.75$. Figure 6.4 shows the results. This sequence of structures show a clear folding pathway: The melted state has less secondary structure, in particular the $\beta$ strands are fully melted. The transition state involves a partially formed $\beta$-sheet, In particular, the hairpin formed by the first two $\beta$-strands is partially formed. The intermediate has the $\beta$-sheet fully formed, with the C-terminal $\alpha$-helix out of place and partially melted, and the mid-sequence $\alpha$-helix in place, but partially melted. The final step is the simultaneous formation of both $\alpha$ helices, and the movement of the C-terminal helix into its native orientation relative to the $\beta$-sheet.

To verify our proposed mechanism, we compute the average values of the $\beta$-sheet RMSD (i.e., the RMSD for residues 1-5, 16-20, and 35-39) and the two $\alpha$-helix RMSDs (residues 23-33 and 41-51) as functions of the $-\vec{\beta} \cdot \vec{\theta}$ CV. That is, we compute the conditional mean for function $f$:

$$\mathbb{E}_\pi[f(X^t)|\theta(X^t) = \theta^*] = \frac{\mathbb{E}_\pi[f(x)\delta(\theta(x) - \theta^*)]}{\mathbb{E}_\pi[\delta(\theta(x) - \theta^*)]}. \tag{6.11}$$

In practice, the $\delta$-functions are approximated by bins of uniform width; for this section we use 100 bins. We compute the equilibrium averages using the BAD-NEUS data and (5.14). Figure 6.5 shows the results. We reproduce the PMF along with the conditional mean of the three RMSD coordinates for reference. The top center panel shows the conditional mean of the $\beta$-sheet RMSD. We see that at the transition state, the RMSD sharply drops, indicating

that crossing this barrier corresponds to full formation of the beta sheet. The bottom two panels show the two $\alpha$-helix RMSD coordinates. Folding of the first $\alpha$-helix coincides with barrier, while folding of the second follows it.

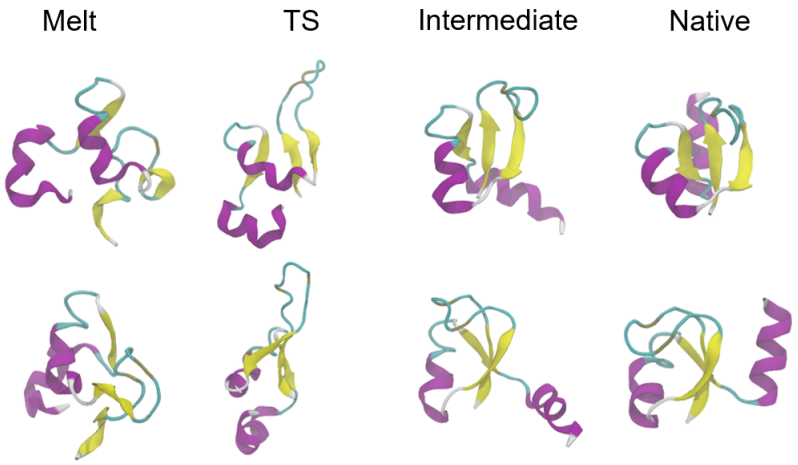


Figure 6.4: Sequence of structures from the BAD-NEUS simulation. Left to right: melted state, $-\vec{\beta}\cdot\vec{\theta} = 0.99$, intermediate at $\sum_i \beta_i\theta_i = 20.34$, and native state. In the transition state (i.e., the highest barrier located at $-\vec{\beta}\cdot\vec{\theta} = 0.99$, which corresponds to $q_- = 0.3$) that the $\beta$-sheet is partially formed, with the N-terminal $\beta$-hairpin partially formed. In the intermediate, the $\beta$-sheet is fully formed, while the C-terminal $\alpha$-helix is out of place and partially melted, and the central $\alpha$-helix is still partially melted. Top and bottom rows are the same structures rotated by $90^\circ$. to show the side view of the $\beta$-sheet.

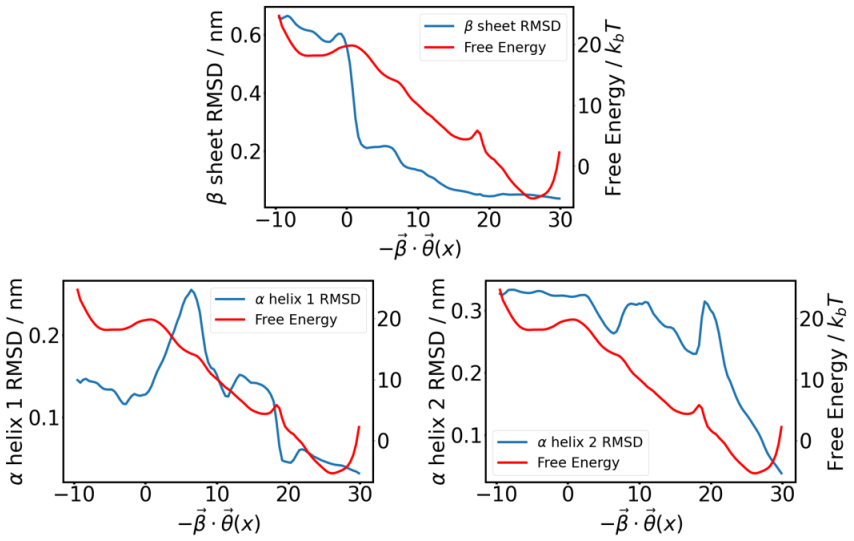


Figure 6.5: Average values of the $\beta$ sheet RMSD (top), central $\alpha$-helix RMSD (bottom left), and C-terminal $\alpha$-helix RMSD (bottom right) We see that the $\beta$ sheet RMSD drops sharply around the first local maximum in the logit-committor PMF, and the two helix RMSDs drop sharply around the second (and much lower) free energy barrier.

We can also compute PMFs and conditional averages of functions on other CV spaces. To illustrate, we here look at the 2 dimensional CV space of the full backbone RMSD and the fraction of native contacts. Figure 6.6 shows the results. The contour lines are spaced every $2k_BT$, and the conditional means of the backward committor, the logit of the conditional mean of the backward committor, the $\beta$ sheet RMSD, and the C terminal $\alpha$ helix RMSD are shown. The PMFs on the 2 dimensional CV space also show a single major barrier around a backbone RMSD of 0.75 nm and a fraction of native contacts $Q = .45$, as well as an intermediate state at backbone RMSD of around .55 nm and $Q = .75$. We also see that crossing the main barrier correlates with a sharp drop in the $\beta$ sheet RMSD, while the $\alpha$ helix RMSD only drops after leaving the intermediate state. These observations are in agreement with our analysis based on the backward committor extracted by logistic regression. This agreement gives us confidence that we have indeed resolved the backward committor accurately.

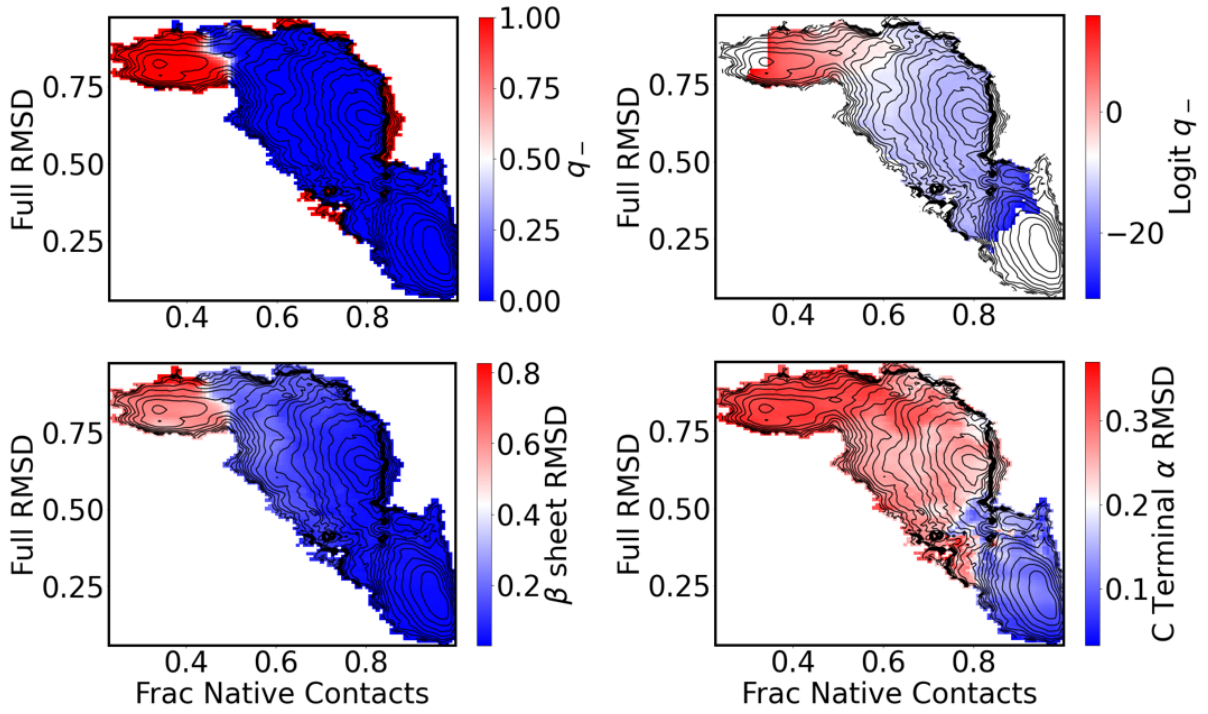


Figure 6.6: BAD-NEUS backward committor and PMF contours for NTL9 after 40 iterations. Top left: Backward committor projected onto the fraction of native contacts and the full RMSD. Top right: Logit of the projected backward committor. Bottom left: $\beta$-sheet RMSD. Bottom right: C-terminal $\alpha$-helix RMSD. We find that the formation of the $\beta$-sheet very closely correlates with a large decrease in the backward committor, suggesting that this step is rate-limiting, in agreement with the findings of Ref. [5]. We also see that the formation of the C-terminal $\alpha$-helix is the final step, and only occurs after the protein is nearly fully committed to folding.

Our NEUS results may be compared to other computational and experimental studies of NTL9. As may be seen in Figure 6.4, our transition state has a native-like tertiary structure, with significant partially formed contacts among the three $\beta$-strands and among the one $\alpha$-helix and the $\beta$-strands. This observation is broadly in agreement with the experimental results of Ref. 226. Satoshi et al. measured $\phi$ values for a variety of mutants, and found that most residues have a small $\phi$ value, indicating that specific side chain interactions are not strongly formed in the transition state. They combine this observation with other experimental data to conclude that the transition state has significant hydrogen bond structure, and a native like topology. This is broadly in agreement with our BAD-NEUS results. In

addition, comparison of folding of the truncated peptide (residues 1-51) has shown that the C-terminal helix adopts a helical conformation in the melted state, but full consolidation of the helical structure does not occur until after the rate limiting step [225]. This observation agrees with our proposed mechanism, which has both helices fully adopting their native structures only after the peptide is fully committed to folding having crossed the highest barrier.

## 6.5 Conclusions

In this chapter, we have illustrated the BAD-NEUS method by computing the rate, backward committor, and PMF for NTL9 folding. Our calculated folding rate agrees well with prior experimental and computational work, but in addition to the rate, we are also able to compute the backward committor and stationary distribution. We showed how the backward committor may be extracted by logistic regression, and subsequently used as a collective variable. We used the function whose sigmoid is the regressed committor as a collective variable, and found that this CV enabled us to extract transition state and intermediate structures which suggest a folding mechanism that proceeds by first forming the beta sheet, followed by formation of the alpha helices only after the rate limiting step has occurred. Our results agree with prior experimental observations.

# CHAPTER 7

# CONCLUSIONS AND OUTLOOK

In this thesis, we have developed a number of new techniques for attacking the rare event problem. Our basic approach throughout has been to use short, unbiased molecular dynamics trajectories to compute kinetic statistics of interest. In Chapter 2, we refined the DGA [35] method. This method solves the Feynman-Kac equation for prediction statistics and the stationary distribution using a linear basis expansion. In addition, we propose estimators for the reactive current projected onto a CV space and reaction rate, and we apply the DGA method and our rate and current estimators to understanding the folding of the trp-cage miniprotein. To our knowledge, this is the first computation of the reactive current on a high-dimensional system using data; previous studies were limited to low-dimensional systems because they relied on direct solution of the partial-differential equation (e.g., Ref. 233). There are a number of extensions and refinements to our work that could be pursued. Principally, at present we do not have a way to estimate the magnitude of the sampling error. In Appendices 2.6.4 and 2.6.5, we propose asymptotic formulae for the sampling error, as well as a possible approach to drawing new samples in an optimal way, but we have not been able fully test and refine these proposed approaches.

In Chapters 3 and 4, we again tackle the prediction problem. A major limitation of the DGA method is that the quality of the approximation is limited by the expressivity of the basis set, even with infinite sampling. To address this limitation of the linear method, we developed two machine learning (ML) approaches to the prediction problem (that is, solving the Feynman-Kac equation). Our first ML approach required multiple independent short trajectories launched from each initial condition, which limited its utility. To relax this requirement we developed the inexact subspace iteration in Chapter 5. We illustrated our ML approaches on two-dimensional toy models, and also an atmospheric model, and a molecular model, and we found that our machine learning approaches agree with brute

force calculations in all cases. The main thrust of future work here is to extend our ML approaches to the adjoint problem. Our current methods only work for solving operator equations involving the forward in time transition operator, yet many quantities of interest, namely the backward committor and stationary distribution, satisfy equations involving the adjoint of the transition operator. More work will be needed to modify our subspace iteration to solve such adjoint equations.

Finally, in Chapters 5 and 6, we developed a modification to the weighted ensemble algorithm. Our approach uses the observation that it is possible to insert a short-trajectory based approximation to the stationary distribution into the WE algorithm in such a way that the resulting scheme converges to the true stationary distribution without approximation. In our work, we use a linear basis expansion approximation to the stationary distribution. There are a two main future directions for developing these methods. The first is to perform a rigorous error analysis of NEUS along the lines of previous error analyses of EMUS or MBAR [8, 234]. The similar structure of the algorithms suggests that a similar approach could be successful. This analysis would help us refine stratification schemes and assess convergence. The second future direction is to use more accurate stationary distribution approximations than our linear ansatz. This could involve modifying the approach in Ref. 173 to approximate the weights at each NEUS step. Finally, our NTL9 analysis is preliminary. Future work will analyze the folding mechanism in more detail, including examining the effect of truncating the peptide, and looking at the extent of side chain interactions as folding progresses in order to make more direct comparisons to experimental data.

# CHAPTER 8

# REFERENCES


[1] J. Juraszek and P. G. Bolhuis. Sampling the multiple folding mechanisms of Trp-cage in explicit solvent. *Proceedings of the National Academy of Sciences*, 103(43):15859–15864, October 2006. Publisher: National Academy of Sciences Section: Biological Sciences.

[2] Klaus Müller and Leo D. Brown. Location of saddle points and minimum energy paths by a constrained simplex optimization procedure. *Theoretica Chimica Acta*, 53(1):75–93, March 1979.

[3] Erik H Thiede, Dimitrios Giannakis, Aaron R Dinner, and Jonathan Weare. Galerkin approximation of dynamical quantities using trajectory data. *Journal of Chemical Physics*, 150(24):244111, 2019.

[4] Chatipat Lorpaiboon, Jonathan Weare, and Aaron R Dinner. Augmented transition path theory for sequences of events. *arXiv preprint arXiv:2205.05067*, 2022.

[5] Upendra Adhikari, Barmak Mostofian, Jeremy Copperman, Sundar Raman Subramanian, Andrew A. Petersen, and Daniel M. Zuckerman. Computational Estimation of Microsecond to Second Atomistic Folding Times. *Journal of the American Chemical Society*, 141(16):6519–6526, April 2019. Publisher: American Chemical Society.

[6] Alessandro Barducci, Giovanni Bussi, and Michele Parrinello. Well-Tempered Metadynamics: A Smoothly Converging and Tunable Free-Energy Method. *Physical Review Letters*, 100(2):020603, January 2008. Publisher: American Physical Society.

[7] Michael R. Shirts and John D. Chodera. Statistically optimal analysis of samples from multiple equilibrium states. *The Journal of Chemical Physics*, 129(12):124105, September 2008.

[8] Erik H Thiede, Brian Van Koten, Jonathan Weare, and Aaron R Dinner. Eigenvector method for umbrella sampling enables error analysis. *Journal of Chemical Physics*, 145(8):084115, 2016.

[9] Shankar Kumar, John M. Rosenberg, Djamal Bouzida, Robert H. Swendsen, and Peter A. Kollman. THE weighted histogram analysis method for free-energy calculations on biomolecules. I. The method. *Journal of Computational Chemistry*, 13(8):1011–1021, 1992. _eprint: https://onlinelibrary.wiley.com/doi/pdf/10.1002/jcc.540130812.

[10] Gary A Huber and Sangtae Kim. Weighted-ensemble Brownian dynamics simulations for protein association reactions. *Biophysical Journal*, 70(1):97, 1996.

[11] Alex Dickson and Aaron R Dinner. Enhanced sampling of nonequilibrium steady states. *Annual Review of Physical Chemistry*, 61(1):441–459, 2010.

[12] Nicholas Guttenberg, Aaron R. Dinner, and Jonathan Weare. Steered transition path sampling. *Journal of Chemical Physics*, 136(23):234103, June 2012. Publisher: American Institute of Physics.

[13] David Aristoff, Juan M. Bello-Rivas, and Ron Elber. A Mathematical Framework for Exact Milestoning. *Multiscale Modeling & Simulation*, 14(1):301–322, January 2016. Publisher: Society for Industrial and Applied Mathematics.

[14] Aaron R. Dinner, Jonathan C. Mattingly, Jeremy O. B. Tempkin, Brian Van Koten, and Jonathan Weare. Trajectory Stratification of Stochastic Dynamics. *SIAM Review*, 60(4):909–938, January 2018. Publisher: Society for Industrial and Applied Mathematics.

[15] Rosalind J Allen, Daan Frenkel, and Pieter Rein ten Wolde. Simulating rare events in equilibrium or nonequilibrium stochastic systems. *The Journal of chemical physics*, 124(2):024102, 2006.

[16] Cristian Giardinà, Jorge Kurchan, and Luca Peliti. Direct Evaluation of Large-Deviation Functions. *Physical Review Letters*, 96(12):120603, March 2006. Publisher: American Physical Society.

[17] Vivien Lecomte and Julien Tailleur. A numerical approach to large deviations in continuous time. *Journal of Statistical Mechanics: Theory and Experiment*, page P03004, March 2007. Publisher: IOP Publishing.

[18] Alexandra Lamtyugina, Yuqing Qiu, Étienne Fodor, Aaron R. Dinner, and Suriyanarayanan Vaikuntanathan. Thermodynamic Control of Activity Patterns in Cytoskeletal Networks. *Physical review letters*, 129(12):128002, September 2022.

[19] Nils E. Strand, Hadrien Vroylandt, and Todd R. Gingrich. Computing time-periodic steady-state currents via the time evolution of tensor network states. *The Journal of Chemical Physics*, 157(5):054104, August 2022.

[20] G A Huber and S Kim. Weighted-ensemble Brownian dynamics simulations for protein association reactions. *Biophysical Journal*, 70(1):97–110, January 1996.

[21] Peter J. Schmid. Dynamic mode decomposition of numerical and experimental data. *Journal of Fluid Mechanics*, 656:5–28, August 2010. Publisher: Cambridge University Press.

[22] Jan-Hendrik Prinz, Hao Wu, Marco Sarich, Bettina Keller, Martin Senne, Martin Held, John D Chodera, Christof Schütte, and Frank Noé. Markov models of molecular kinetics: Generation and validation. *The Journal of chemical physics*, 134(17):174105, 2011.

[23] Feliks Nüske, Bettina G. Keller, Guillermo Pérez-Hernández, Antonia S. J. S. Mey, and Frank Noé. Variational approach to molecular kinetics. *Journal of Chemical Theory and Computation*, 10(4):1739–1752, 2014.

[24] Matthew O. Williams, Ioannis G. Kevrekidis, and Clarence W. Rowley. A data–driven approximation of the Koopman operator: Extending dynamic mode decomposition. *Journal of Nonlinear Science*, 25(6):1307–1346, December 2015.

[25] Brooke E. Husic and Vijay S. Pande. Markov state models: From an art to a Science. *Journal of the American Chemical Society*, 140(7):2386–2396, February 2018.

[26] Stefan Klus, Feliks Nüske, Péter Koltai, Hao Wu, Ioannis Kevrekidis, Christof Schütte, and Frank Noé. Data-driven model reduction and transfer operator approximation. *Journal of Nonlinear Science*, 28:985–1010, 2018.

[27] Vijay S. Pande, Kyle Beauchamp, and Gregory R. Bowman. Everything you wanted to know about Markov State Models but were afraid to ask. *Methods (San Diego, Calif.)*, 52(1):99–105, September 2010.

[28] Tanja Eisner, Bálint Farkas, Markus Haase, and Rainer Nagel. *Operator Theoretic Aspects of Ergodic Theory*, volume 272. Springer, 2015.

[29] Hao Wu, Feliks Nüske, Fabian Paul, Stefan Klus, Péter Koltai, and Frank Noé. Variational Koopman models: Slow collective variables and molecular kinetics from short off-equilibrium simulations. *Journal of Chemical Physics*, 146(15):154104, April 2017.

[30] Andreas Mardt, Luca Pasquali, Hao Wu, and Frank Noé. VAMPnets for deep learning of molecular kinetics. *Nature Communications*, 9(1):1–11, January 2018. Number: 1 Publisher: Nature Publishing Group.

[31] Feliks Nüske, Reinhold Schneider, Francesca Vitalini, and Frank Noé. Variational tensor approach for approximating the rare-event kinetics of macromolecular systems. *The Journal of Chemical Physics*, 144(5):054105, 2016.

[32] Jan-Hendrik Prinz, John D Chodera, and Frank Noé. Spectral rate theory for two-state kinetics. *Physical Review X*, 4(1):011020, 2014.

[33] Pierre Del Moral. *Feynman-Kac Formulae*. Springer, 2004.

[34] Ioannis Karatzas and Steven Shreve. *Brownian Motion and Stochastic Calculus*, volume 113. Springer Science & Business Media, 2012.

[35] Erik H. Thiede, Dimitrios Giannakis, Aaron R. Dinner, and Jonathan Weare. Galerkin approximation of dynamical quantities using trajectory data. *Journal of Chemical Physics*, 150(24):244111, 2019.

[36] Hao Wu, Feliks Nüske, Fabian Paul, Stefan Klus, Péter Koltai, and Frank Noé. Variational Koopman models: slow collective variables and molecular kinetics from short off-equilibrium simulations. *The Journal of Chemical Physics*, 146(15):154104, 2017.

[37] Ronald R. Coifman and Stéphane Lafon. Diffusion maps. *Applied and Computational Harmonic Analysis*, 21(1):5–30, July 2006.

[38] David E. Shaw, Paul Maragakis, Kresten Lindorff-Larsen, Stefano Piana, Ron O. Dror, Michael P. Eastwood, Joseph A. Bank, John M. Jumper, John K. Salmon, Yibing Shan, and Willy Wriggers. Atomic-Level Characterization of the Structural Dynamics of Proteins. *Science*, 330(6002):341–346, October 2010. Publisher: American Association for the Advancement of Science Section: Research Article.

[39] Linlin Qiu, Suzette A. Pabit, Adrian E. Roitberg, and Stephen J. Hagen. Smaller and Faster: The 20-Residue Trp-Cage Protein Folds in 4 s. *Journal of the American Chemical Society*, 124(44):12952–12953, November 2002. Publisher: American Chemical Society.

[40] Jarek Juraszek and Peter G. Bolhuis. Rate Constant and Reaction Coordinate of Trp-Cage Folding in Explicit Water. *Biophysical Journal*, 95(9):4246–4257, November 2008. Publisher: Elsevier.

[41] Fabrizio Marinelli, Fabio Pietrucci, Alessandro Laio, and Stefano Piana. A kinetic model of trp-cage folding from multiple biased molecular dynamics simulations. *PLoS Comput Biol*, 5(8):e1000452, 2009.

[42] Kresten Lindorff-Larsen, Stefano Piana, Ron O. Dror, and David E. Shaw. How Fast-Folding Proteins Fold | Science. *Science*, 334(6055):517–520, October 2011.

[43] Nan-jie Deng, Wei Dai, and Ronald M. Levy. How Kinetics within the Unfolded State Affects Protein Folding: An Analysis Based on Markov State Models and an Ultra-Long MD Trajectory. *Journal of Physical Chemistry B*, 117(42):12787–12799, October 2013. Publisher: American Chemical Society.

[44] Hythem Sidky, Wei Chen, and Andrew L Ferguson. High-resolution markov state models for the dynamics of Trp-cage miniprotein constructed over slow folding modes identified by state-free reversible VAMPnets. *Journal of Physical Chemistry B*, 123(38):7999–8009, 2019.

[45] Lutz Molgedey and Heinz Georg Schuster. Separation of a mixture of independent signals using time delayed correlations. *Physical review letters*, 72(23):3634, 1994.

[46] Guillermo Pérez-Hernández, Fabian Paul, Toni Giorgino, Gianni De Fabritiis, and Frank Noé. Identification of slow molecular order parameters for markov model construction. *The Journal of chemical physics*, 139(1):07B604_1, 2013.

[47] Christian R Schwantes and Vijay S Pande. Improvements in markov state model construction reveal many non-native interactions in the folding of ntl9. *Journal of chemical theory and computation*, 9(4):2000–2009, 2013.

[48] Robert J. Webber, Erik H. Thiede, Douglas Dow, Aaron R. Dinner, and Jonathan Weare. Error bounds for dynamical spectral estimation. *arXiv:2005.02248 [physics]*, May 2020. arXiv: 2005.02248.

[49] Christof Schütte, Frank Noé, Jianfeng Lu, Marco Sarich, and Eric Vanden-Eijnden. Markov state models based on milestoning. *The Journal of chemical physics*, 134(20):05B609, 2011.

[50] F Vitalini, F Noé, and BG Keller. A basis set for peptides for the variational approach to conformational kinetics. *Journal of chemical theory and computation*, 11(9):3992–4004, 2015.

[51] Lorenzo Boninsegna, Gianpaolo Gobbo, Frank Noé, and Cecilia Clementi. Investigating molecular kinetics by variationally optimized diffusion maps. *Journal of chemical theory and computation*, 11(12):5947–5960, 2015.

[52] Marcus Weber, Konstantin Fackeldey, and Christof Schütte. Set-free markov state model building. *The Journal of Chemical Physics*, 146(12):124133, 2017.

[53] Ernesto Suárez, Joshua L. Adelman, and Daniel M. Zuckerman. Accurate Estimation of Protein Folding and Unfolding Times: Beyond Markov State Models. *Journal of Chemical Theory and Computation*, 12(8):3473–3481, August 2016. Publisher: American Chemical Society.

[54] Robert Zwanzig. *Nonequilibrium Statistical Mechanics*. Oxford University Press, 2001.

[55] Alexandre J Chorin, Ole H Hald, and Raz Kupferman. Optimal prediction with memory. *Physica D: Nonlinear Phenomena*, 166(3-4):239–257, 2002.

[56] Nicholas Guttenberg, James F Dama, Marissa G Saunders, Gregory A Voth, Jonathan Weare, and Aaron R Dinner. Minimizing memory as an objective for coarse-graining. *The Journal of Chemical Physics*, 138(9):094111, 2013.

[57] Eric Vanden-Eijnden. Transition path theory. In *Computer Simulations in Condensed Matter Systems: From Materials to Chemical Biology Volume 1*, pages 453–493. Springer, 2006.

[58] Philipp Metzner, Christof Schütte, and Eric Vanden-Eijnden. Transition Path Theory for Markov Jump Processes. *Multiscale Modeling & Simulation*, 7(3):1192–1219, January 2009. Publisher: Society for Industrial and Applied Mathematics.

[59] Mark James Abraham, Teemu Murtola, Roland Schulz, Szilárd Páll, Jeremy C. Smith, Berk Hess, and Erik Lindah. Gromacs: High performance molecular simulations through multi-level parallelism from laptops to supercomputers. *SoftwareX*, 1-2:19–25, 2015.

[60] Massimiliano Bonomi, Davide Branduardi, Giovanni Bussi, Carlo Camilloni, Davide Provasi, Paolo Raiteri, Davide Donadio, Fabrizio Marinelli, Fabio Pietrucci, Ricardo A. Broglia, and Michele Parrinello. PLUMED: A portable plugin for free-energy calculations with molecular dynamics. *Computer Physics Communications*, 180(10):1961–1972, 2009.

[61] Gareth A. Tribello, Massimiliano Bonomi, Davide Branduardi, Carlo Camilloni, and Giovanni Bussi. PLUMED 2: New feathers for an old bird. *Computer Physics Communications*, 185(2):604–613, 2014.

[62] PLUMED. Promoting transparency and reproducibility in enhanced molecular simulations. *Nature Methods*, 16(8):670–673, aug 2019.

[63] A. D. MacKerell, D. Bashford, M. Bellott, R. L. Dunbrack, J. D. Evanseck, M. J. Field, S. Fischer, J. Gao, H. Guo, S. Ha, D. Joseph-McCarthy, L. Kuchnir, K. Kuczera, F. T. K. Lau, C. Mattos, S. Michnick, T. Ngo, D. T. Nguyen, B. Prodhom, W. E. Reiher, B. Roux, M. Schlenkrich, J. C. Smith, R. Stote, J. Straub, M. Watanabe, J. Wiórkiewicz-Kuczera, D. Yin, and M. Karplus. All-Atom Empirical Potential for Molecular Modeling and Dynamics Studies of Proteins †. *The Journal of Physical Chemistry B*, 102(18):3586–3616, apr 1998.

[64] Robert B. Best, Xiao Zhu, Jihyun Shim, Pedro E. M. Lopes, Jeetain Mittal, Michael Feig, and Alexander D. MacKerell. Optimization of the Additive CHARMM All-Atom Protein Force Field Targeting Improved Sampling of the Backbone $\phi$, $\psi$ and Side-Chain $\chi$ 1 and $\chi$ 2 Dihedral Angles. *Journal of Chemical Theory and Computation*, 8(9):3257–3273, sep 2012.

[65] Jing Huang, Sarah Rauscher, Grzegorz Nawrocki, Ting Ran, Michael Feig, Bert L. De Groot, Helmut Grubmüller, and Alexander D. MacKerell. CHARMM36m: An improved force field for folded and intrinsically disordered proteins. *Nature Methods*, 14(1):71–73, 2016.

[66] Berk Hess, Henk Bekker, Herman J. C. Berendsen, and Johannes G. E. M. Fraaije. LINCS: A linear constraint solver for molecular simulations. *Journal of Computational Chemistry*, 18(12):1463–1472, 1997. _eprint: https://onlinelibrary.wiley.com/doi/pdf/10.1002/%28SICI%291096-987X%28199709%2918%3A12%3C1463%3A%3AAID-JCC4%3E3.0.CO%3B2-H.

[67] Jonathan W. Neidigh, R. Matthew Fesinmeyer, and Niels H. Andersen. Designing a 20-residue protein. *Nature Structural Biology*, 9(6):425–430, June 2002.

[68] William L Jorgensen, Jayaraman Chandrasekhar, Jeffry D Madura, Roger W Impey, and Michael L Klein. Comparison of simple potential functions for simulating liquid water. *The Journal of Chemical Physics*, 79(2):926–935, 1983.

[69] Sunhwan Jo, Taehoon Kim, Vidyashankara G. Iyer, and Wonpil Im. CHARMM-GUI: A web-based graphical user interface for CHARMM. *Journal of Computational Chemistry*, 29(11):1859–1865, aug 2008.

[70] Jumin Lee, Xi Cheng, Jason M. Swails, Min Sun Yeom, Peter K. Eastman, Justin A. Lemkul, Shuai Wei, Joshua Buckner, Jong Cheol Jeong, Yifei Qi, Sunhwan Jo, Vijay S.

Pande, David A. Case, Charles L. Brooks, Alexander D. MacKerell, Jeffery B. Klauda, and Wonpil Im. CHARMM-GUI Input Generator for NAMD, GROMACS, AMBER, OpenMM, and CHARMM/OpenMM Simulations Using the CHARMM36 Additive Force Field. *Journal of Chemical Theory and Computation*, 12(1):405–413, 2016.

[71] Massimo Marchi and Pietro Ballone. Adiabatic bias molecular dynamics: A method to navigate the conformational space of complex molecular systems. *Journal of Chemical Physics*, 110(8):3697–3702, 1999.

[72] Soohyung Park, Taehoon Kim, and Wonpil Im. Transmembrane Helix Assembly by Window Exchange Umbrella Sampling. *Physical Review Letters*, 108(10):108102, mar 2012.

[73] Yuji Sugita, Akio Kitao, and Yuko Okamoto. Multidimensional replica-exchange method for free-energy calculations. *Journal of Chemical Physics*, 113(15):6042–6051, 2000.

[74] Erik H. Thiede, Brian Van Koten, Jonathan Weare, and Aaron R. Dinner. Eigenvector method for umbrella sampling enables error analysis. *Journal of Chemical Physics*, 145(8), 2016.

[75] Adam Antoszewski, Chi-Jui Feng, Bodhi P Vani, Erik H Thiede, Lu Hong, Jonathan Weare, Andrei Tokmakoff, and Aaron R Dinner. Insulin Dissociates by Diverse Mechanisms of Coupled Unfolding and Unbinding. *The Journal of Physical Chemistry B*, 124(27):5571–5587, jul 2020.

[76] Aaron R. Dinner, Erik H. Thiede, Brian Van Koten, and Jonathan Weare. Stratification as a General Variance Reduction Method for Markov Chain Monte Carlo. *SIAM/ASA Journal on Uncertainty Quantification*, pages 1139–1188, January 2020. Publisher: Society for Industrial and Applied Mathematics.

[77] Chatipat Lorpaiboon, Erik Henning Thiede, Robert J. Webber, Jonathan Weare, and Aaron R. Dinner. Integrated VAC: A robust strategy for identifying eigenfunctions of dynamical operators. *arXiv:2007.08027 [physics]*, July 2020. arXiv: 2007.08027.

[78] Sang Beom Kim, Carmeline J. Dsilva, Ioannis G. Kevrekidis, and Pablo G. Debenedetti. Systematic characterization of protein folding pathways using diffusion maps: Application to Trp-cage miniprotein. *Journal of Chemical Physics*, 142(8):085101, February 2015. Publisher: American Institute of Physics.

[79] Erik Thiede, Brian Van Koten, and Jonathan Weare. Sharp entrywise perturbation bounds for markov chains. *SIAM Journal on Matrix Analysis and Applications*, 36(3):917–941, 2015.

[80] Ao Ma and Aaron R. Dinner. Automatic method for identifying reaction coordinates in complex systems. *Journal of Physical Chemistry B*, 109(14):6769–6779, April 2005.

[81] Jie Hu, Ao Ma, and Aaron R Dinner. A two-step nucleotide-flipping mechanism enables kinetic discrimination of DNA lesions by AGT. *Proceedings of the National Academy of Sciences*, 105(12):4615–4620, 2008.

[82] Nicolai Kozlowski and Helmut Grubmüller. Uncertainties in Markov State Models of Small Proteins. *Journal of Chemical Theory and Computation*, 19(16):5516–5524, August 2023. Publisher: American Chemical Society.

[83] G. W. Stewart. Stochastic Perturbation Theory. *SIAM Review*, 32(4):579–610, December 1990. Publisher: Society for Industrial and Applied Mathematics.

[84] David R. Easterling, Gerald A. Meehl, Camille Parmesan, Stanley A. Changnon, Thomas R. Karl, and Linda O. Mearns. Climate extremes: Observations, modeling, and impacts. *Science*, 289(5487):2068–2074, 2000.

[85] Amir AghaKouchak, Linyin Cheng, Omid Mazdiyasni, and Alireza Farahmand. Global warming and changes in risk of concurrent climate extremes: Insights from the 2014 California drought. *Geophysical Research Letters*, 41(24):8847–8852, 2014.

[86] Corey Lesk, Pedram Rowhani, and Navin Ramankutty. Influence of extreme weather disasters on global crop production. *Nature*, 529(7584):84–87, Jan 2016.

[87] Michael E. Mann, Stefan Rahmstorf, Kai Kornhuber, Byron A. Steinman, Sonya K. Miller, and Dim Coumou. Influence of anthropogenic climate change on planetary wave resonance and extreme weather events. *Scientific Reports*, 7(1):45242, Mar 2017.

[88] David J. Frame, Suzanne M. Rosier, Ilan Noy, Luke J. Harrington, Trevor Carey-Smith, Sarah N. Sparrow, Dáithí A. Stone, and Samuel M. Dean. Climate change attribution and the economic costs of extreme weather events: a study on damages from extreme rainfall and drought. *Climatic Change*, 162(2):781–797, Sep 2020.

[89] Themistoklis P. Sapsis. Statistics of extreme events in fluid flows and waves. *Annual Review of Fluid Mechanics*, 53(1):85–111, 2021.

[90] C. L. Brooks, III, M. Karplus, and B. M. Pettitt. *Proteins*. John Wiley & Sons, New York, 1988.

[91] Matthew C Zwier and Lillian T Chong. Reaching biological timescales with all-atom molecular dynamics simulations. *Current opinion in pharmacology*, 10(6):745–752, 2010.

[92] Muhammad Saqib Sohail, Raymond HY Louie, Matthew R McKay, and John P Barton. Mpl resolves genetic linkage in fitness inference from complex evolutionary histories. *Nature Biotechnology*, 39(4):472–479, 2021.

[93] Gary R Mirams, Mark R Davies, Yi Cui, Peter Kohl, and Denis Noble. Application of

cardiac electrophysiology simulations to pro-arrhythmic safety testing. *British Journal of Pharmacology*, 167(5):932–945, 2012.

[94] Weiqing Liu, Tae Yun Kim, Xiaodong Huang, Michael B Liu, Gideon Koren, Bum-Rak Choi, and Zhilin Qu. Mechanisms linking t-wave alternans to spontaneous initiation of ventricular arrhythmias in rabbit models of long qt syndrome. *Journal of Physiology*, 596(8):1341–1355, 2018.

[95] Siewert J Marrink, H Jelger Risselada, Serge Yefimov, D Peter Tieleman, and Alex H De Vries. The martini force field: coarse grained model for biomolecular simulations. *Journal of Physical Chemistry B*, 111(27):7812–7824, 2007.

[96] Marissa G Saunders and Gregory A Voth. Coarse-graining methods for computational biology. *Annual Review of Biophysics*, 42:73–93, 2013.

[97] John M. Jumper, Nabil F. Faruk, Karl F. Freed, and Tobin R. Sosnick. Trajectory-based training enables protein simulations with accurate folding and Boltzmann ensembles in cpu-hours. *PLoS Computational Biology*, 14(12):e1006578, 2018.

[98] Robert Zwanzig. From classical dynamics to continuous time random walks. *Journal of Statistical Physics*, 30(2):255–262, February 1983.

[99] Frank Noé, Christof Schütte, Eric Vanden-Eijnden, Lothar Reich, and Thomas R Weikl. Constructing the equilibrium ensemble of folding pathways from short off-equilibrium simulations. *Proceedings of the National Academy of Sciences*, 106(45):19011–19016, 2009.

[100] Gregory R Bowman, Vijay S Pande, and Frank Noé. *An introduction to Markov state models and their application to long timescale molecular simulation*, volume 797. Springer Science & Business Media, 2013.

[101] J Nathan Kutz. Deep learning in fluid dynamics. *Journal of Fluid Mechanics*, 814:1–4, 2017.

[102] Samuel H Rudy, Steven L Brunton, Joshua L Proctor, and J Nathan Kutz. Data-driven discovery of partial differential equations. *Science Advances*, 3(4):e1602614, 2017.

[103] Paul A O'Gorman and John G Dwyer. Using machine learning to parameterize moist convection: Potential for modeling of climate, climate change, and extreme events. *Journal of Advances in Modeling Earth Systems*, 10(10):2548–2563, 2018.

[104] Laure Zanna and Thomas Bolton. Data-driven equation discovery of ocean mesoscale closures. *Geophysical Research Letters*, 47(17):e2020GL088376, 2020.

[105] A. Chattopadhyay, P. Hassanzadeh, and D. Subramanian. Data-driven predictions of a multiscale Lorenz 96 chaotic system using machine-learning methods: reservoir computing, artificial neural network, and long short-term memory network. *Nonlinear Processes in Geophysics*, 27(3):373–389, 2020.

[106] K. Kashinath, M. Mustafa, A. Albert, J-L. Wu, C. Jiang, S. Esmaeilzadeh, K. Azizzadenesheli, R. Wang, A. Chattopadhyay, A. Singh, A. Manepalli, D. Chirila, R. Yu, R. Walters, B. White, H. Xiao, H. A. Tchelepi, P. Marcus, A. Anandkumar, P. Hassanzadeh, and Prabhat. Physics-informed machine learning: case studies for weather and climate modelling. *Philosophical Transactions of the Royal Society A: Mathematical, Physical and Engineering Sciences*, 379(2194):20200093, 2021.

[107] Rose Du, Vijay S Pande, Alexander Yu Grosberg, Toyoichi Tanaka, and Eugene S Shakhnovich. On the transition coordinate for protein folding. *Journal of Chemical Physics*, 108(1):334–350, 1998.

[108] Peter G Bolhuis, David Chandler, Christoph Dellago, and Phillip L Geissler. Transition

path sampling: Throwing ropes over mountain passes in the dark. *Annual Review of Physical Chemistry*, 53:291–318, 2002.

[109] John Strahan, Adam Antoszewski, Chatipat Lorpaiboon, Bodhi P Vani, Jonathan Weare, and Aaron R Dinner. Long-time-scale predictions from short-trajectory data: A benchmark analysis of the trp-cage miniprotein. *Journal of Chemical Theory and Computation*, 17(5):2948–2963, 2021.

[110] Adam Antoszewski, Chatipat Lorpaiboon, John Strahan, and Aaron R Dinner. Kinetics of phenol escape from the insulin $R_6$ hexamer. *Journal of Physical Chemistry B*, 125(42):11637–11649, October 2021.

[111] H. C. Bloomfield, D. J. Brayshaw, P. L. M. Gonzalez, and A. Charlton-Perez. Subseasonal forecasts of demand and wind power and solar power generation for 28 European countries. *Earth System Science Data*, 13(5):2259–2274, 2021.

[112] Alexis Tantet, Fiona R. van der Burgt, and Henk A. Dijkstra. An early warning indicator for atmospheric blocking events using transfer operators. *Chaos: An Interdisciplinary Journal of Nonlinear Science*, 25(3):036406, 2015.

[113] Dario Lucente, Joran Rolland, Corentin Herbert, and Freddy Bouchet. Coupling rare event algorithms with data-based learned committor functions using the analogue Markov chain. *arXiv preprint arXiv:2110.05050*, 2021.

[114] Dario Lucente, Corentin Herbert, and Freddy Bouchet. Committor functions for climate phenomena at the predictability margin: The example of el niño southern oscillation in the jin and timmermann model. *Journal of the Atmospheric Sciences*, 2022.

[115] Justin Finkel, Edwin P Gerber, Dorian S Abbot, and Jonathan Weare. Revealing

the statistics of extreme events hidden in short weather forecast data. *arXiv preprint arXiv:2206.05363*, 2022.

[116] Justin Finkel, Robert J Webber, Edwin P Gerber, Dorian S Abbot, and Jonathan Weare. Learning forecasts of rare stratospheric transitions from short simulations. *Monthly Weather Review*, 149(11):3647–3669, November 2021.

[117] Justin Finkel, Robert J Webber, Edwin P Gerber, Dorian S Abbot, and Jonathan Weare. Exploring stratospheric rare events with transition path theory and short simulations. *arXiv preprint arXiv:2108.12727*, 2021.

[118] Grigorios A. Pavliotis. *Stochastic Processes and Applications: Diffusion Processes, the Fokker-Planck and Langevin Equations*, volume 60 of *Texts in Applied Mathematics*. Springer New York, New York, NY, 2014.

[119] Qianxiao Li, Bo Lin, and Weiqing Ren. Computing committor functions for the study of rare events using deep learning. *Journal of Chemical Physics*, 151(5):054112, August 2019.

[120] Yuehaw Khoo, Jianfeng Lu, and Lexing Ying. Solving for high-dimensional committor functions using artificial neural networks. *Research in the Mathematical Sciences*, 6(1):1–13, October 2019.

[121] Haoya Li, Yuehaw Khoo, Yinuo Ren, and Lexing Ying. A semigroup method for high dimensional committor functions based on neural network. In *Mathematical and Scientific Machine Learning*, pages 598–618. PMLR, 2022.

[122] Grant M Rotskoff, Andrew R Mitchell, and Eric Vanden-Eijnden. Active importance sampling for variational objectives dominated by rare events: Consequences for optimization and generalization. In *Mathematical and Scientific Machine Learning*, pages 757–780. PMLR, 2022.

[123] Yian Chen, Jeremy Hoskins, Yuehaw Khoo, and Michael Lindsey. Committor functions via tensor networks. *arXiv preprint arXiv:2106.12515*, 2021.

[124] Jiequn Han, Arnulf Jentzen, and Weinan E. Solving high-dimensional partial differential equations using deep learning. *Proceedings of the National Academy of Sciences*, 115(34):8505–8510, 2018.

[125] Jiequn Han, Jianfeng Lu, and Mo Zhou. Solving high-dimensional eigenvalue problems using deep neural networks: A diffusion Monte Carlo like approach. *Journal of Computational Physics*, 423:109792, 2020.

[126] George Em Karniadakis, Ioannis G Kevrekidis, Lu Lu, Paris Perdikaris, Sifan Wang, and Liu Yang. Physics-informed machine learning. *Nature Reviews Physics*, 3(6):422–440, 2021.

[127] Maziar Raissi, Paris Perdikaris, and George E Karniadakis. Physics-informed neural networks: A deep learning framework for solving forward and inverse problems involving nonlinear partial differential equations. *Journal of Computational Physics*, 378:686–707, 2019.

[128] Fan Chen, Jianguo Huang, Chunmei Wang, and Haizhao Yang. Friedrichs learning: Weak solutions of partial differential equations via deep learning. *arXiv preprint arXiv:2012.08023*, 2020.

[129] Qi Zeng, Spencer H Bryngelson, and Florian Schäfer. Competitive physics informed networks. *arXiv preprint arXiv:2204.11144*, 2022.

[130] Polina V Banushkina and Sergei V Krivov. Nonparametric variational optimization of reaction coordinates. *The Journal of chemical physics*, 143(18):11B607_1, 2015.

[131] Sergei V Krivov. Blind analysis of molecular dynamics. *Journal of Chemical Theory and Computation*, 17(5):2725–2736, 2021.

[132] Benoît Roux. String method with swarms-of-trajectories, mean drifts, lag time, and committor. *Journal of Physical Chemistry A*, 125(34):7558–7571, September 2021.

[133] Benoît Roux. Transition rate theory, spectral analysis, and reactive paths. *The Journal of Chemical Physics*, 156(13):134111, 2022.

[134] Weinan E, Weiqing Ren, and Eric Vanden-Eijnden. Transition pathways in complex systems: Reaction coordinates, isocommittor surfaces, and transition tubes. *Chemical Physics Letters*, 413(1-3):242–247, 2005.

[135] Richard S Sutton and Andrew G Barto. *Reinforcement learning: An introduction*. MIT press, 2018.

[136] Diederik P. Kingma and Jimmy Ba. Adam: A method for stochastic optimization, 2017. arXiv:1412.6980.

[137] Benedict Leimkuhler and Charles Matthews. Rational Construction of Stochastic Numerical Methods for Molecular Sampling. *Applied Mathematics Research eXpress*, 2013(1):34–56, January 2013.

[138] Christian R. Schwantes and Vijay S. Pande. Modeling Molecular Kinetics with tICA and the Kernel Trick. *Journal of Chemical Theory and Computation*, 11(2):600–608, February 2015.

[139] Andreas Bittracher, Stefan Klus, Boumediene Hamzi, Péter Koltai, and Christof Schütte. Dimensionality reduction of complex metastable systems via kernel embeddings of transition manifolds. *Journal of Nonlinear Science*, 31(1):3, December 2020.

[140] Diederik P. Kingma and Max Welling. Auto-encoding variational Bayes. *arXiv preprint arXiv:1312.6114*, May 2014.

[141] Christoph Wehmeyer and Frank Noé. Time-lagged autoencoders: Deep learning of slow collective variables for molecular kinetics. *Journal of Chemical Physics*, 148(24):241703, June 2018.

[142] Aaron R Dinner, Erik H Thiede, Brian Van Koten, and Jonathan Weare. Stratification as a general variance reduction method for Markov chain Monte Carlo. *SIAM/ASA Journal on Uncertainty Quantification*, 8(3):1139–1188, 2020.

[143] Mark P. Baldwin, Blanca Ayarzagüena, Thomas Birner, Neal Butchart, Amy H. Butler, Andrew J. Charlton-Perez, Daniela I. V. Domeisen, Chaim I. Garfinkel, Hella Garny, Edwin P. Gerber, Michaela I. Hegglin, Ulrike Langematz, and Nicholas M. Pedatella. Sudden stratospheric warmings. *Reviews of Geophysics*, 59(1):e2020RG000708, 2021.

[144] James R. Holton and Clifford Mass. Stratospheric vacillation cycles. *Journal of Atmospheric Sciences*, 33(11):2218–2225, 1976.

[145] Jeremiah P. Sjoberg and Thomas Birner. Stratospheric wave–mean flow feedbacks and sudden stratospheric warmings in a simple model forced by upward wave activity flux. *Journal of the Atmospheric Sciences*, 71(11):4055 – 4071, 2014.

[146] Penelope Maher, Edwin P. Gerber, Brian Medeiros, Timothy M. Merlis, Steven Sherwood, Aditi Sheshadri, Adam H. Sobel, Geoffrey K. Vallis, Aiko Voigt, and Pablo Zurita-Gotor. Model hierarchies for understanding atmospheric circulation. *Reviews of Geophysics*, 57(2):250–280, 2019.

[147] Justin Finkel, Dorian S Abbot, and Jonathan Weare. Path properties of atmospheric transitions: illustration with a low-order sudden stratospheric warming model. *Journal of the Atmospheric Sciences*, 77(7):2327–2347, June 2020.

[148] Bo Christiansen. Chaos, quasiperiodicity, and interannual variability: Studies of a

stratospheric vacillation model. *Journal of the Atmospheric Sciences*, 57(18):3161–3173, 2000.

[149] Shigeo Yoden. Dynamical Aspects of Stratospheric Vacillations in a Highly Truncated Model. *Journal of the Atmospheric Sciences*, 44(24):3683–3695, 12 1987.

[150] Frank Noé and Feliks Nüske. A Variational Approach to Modeling Slow Processes in Stochastic Dynamical Systems. *Multiscale Modeling & Simulation*, 11(2):635–655, January 2013. Publisher: Society for Industrial and Applied Mathematics.

[151] Robert J Webber, Erik H Thiede, Douglas Dow, Aaron R Dinner, and Jonathan Weare. Error bounds for dynamical spectral estimation. *SIAM journal on mathematics of data science*, 3(1):225–252, 2021.

[152] Robert T. McGibbon, Brooke E. Husic, and Vijay S. Pande. Identification of simple reaction coordinates from complex dynamics. *Journal of Chemical Physics*, 146(4):044109, 2017.

[153] Luis Busto-Moner, Chi-Jui Feng, Adam Antoszewski, Andrei Tokmakoff, and Aaron R Dinner. Structural ensemble of the insulin monomer. *Biochemistry*, 60(42):3125–3136, 2021.

[154] Guillermo Pérez-Hernández, Fabian Paul, Toni Giorgino, Gianni De Fabritiis, and Frank Noé. Identification of slow molecular order parameters for Markov model construction. *Journal of Chemical Physics*, 139(1):015102, 2013.

[155] Christian R. Schwantes and Vijay S. Pande. Improvements in Markov State Model construction reveal many non-native interactions in the folding of NTL9. *Journal of Chemical Theory and Computation*, 9(4):2000–2009, 2013.

[156] John Strahan, Adam Antoszewski, Chatipat Lorpaiboon, Bodhi P. Vani, Jonathan Weare, and Aaron R. Dinner. Long-time-scale predictions from short-trajectory data:

A benchmark analysis of the trp-cage miniprotein. *Journal of Chemical Theory and Computation*, 17(5):2948–2963, 2021.

[157] William C. Swope, Jed W. Pitera, and Frank Suits. Describing protein folding kinetics by molecular dynamics simulations. 1. Theory. *Journal of Physical Chemistry B*, 108(21):6571–6581, 2004.

[158] Frank Noé, Christof Schütte, Eric Vanden-Eijnden, Lothar Reich, and Thomas R. Weikl. Constructing the equilibrium ensemble of folding pathways from short off-equilibrium simulations. *Proceedings of the National Academy of Sciences*, 106(45):19011–19016, 2009.

[159] Gregory R. Bowman, Vijay S. Pande, and Frank Noé, editors. *An Introduction to Markov State Models and Their Application to Long Timescale Molecular Simulation*, volume 797 of *Advances in Experimental Medicine and Biology*. Springer Netherlands, Dordrecht, 2014.

[160] Justin Finkel, Robert J. Webber, Dorian S. Abbot, Edwin P. Gerber, and Jonathan Weare. Learning forecasts of rare stratospheric transitions from short simulations. *Monthly Weather Review*, 149(11):3647–3669, 2021.

[161] Adam Antoszewski, Chatipat Lorpaiboon, John Strahan, and Aaron R. Dinner. Kinetics of phenol escape from the insulin $R_6$ hexamer. *Journal of Physical Chemistry B*, 125(42):11637–11649, 2021.

[162] Spencer C. Guo, Rong Shen, Benoît Roux, and Aaron R. Dinner. Dynamics of activation in the voltage-sensing domain of Ci-VSP. *bioRxiv*, 2022.

[163] Galen Andrew, Raman Arora, Jeff Bilmes, and Karen Livescu. Deep canonical correlation analysis. In *Proceedings of the 30th International Conference on Machine Learning*, pages 1247–1255. PMLR, 2013.

[164] Bethany Lusch, J. Nathan Kutz, and Steven L. Brunton. Deep learning for universal linear embeddings of nonlinear dynamics. *Nature Communications*, 9(1):4950, 2018.

[165] Wei Chen, Hythem Sidky, and Andrew L. Ferguson. Nonlinear discovery of slow molecular modes using state-free reversible VAMPnets. *Journal of Chemical Physics*, 150(21):214114, 2019.

[166] Aldo Glielmo, Brooke E. Husic, Alex Rodriguez, Cecilia Clementi, Frank Noé, and Alessandro Laio. Unsupervised learning methods for molecular simulation data. *Chemical Reviews*, 2021.

[167] John Strahan, Justin Finkel, Aaron R. Dinner, and Jonathan Weare. Predicting rare events using neural networks and short-trajectory data. *Journal of Computational Physics*, 488:112152, 2023.

[168] Yuehaw Khoo, Jianfeng Lu, and Lexing Ying. Solving for high-dimensional committor functions using artificial neural networks. *Research in the Mathematical Sciences*, 6(1):1, 2018.

[169] Qianxiao Li, Bo Lin, and Weiqing Ren. Computing committor functions for the study of rare events using deep learning. *Journal of Chemical Physics*, 151(5):054112, 2019.

[170] Benoît Roux. Transition rate theory, spectral analysis, and reactive paths. *Journal of Chemical Physics*, 156(13):134111, 2022.

[171] Grant M. Rotskoff, Andrew R. Mitchell, and Eric Vanden-Eijnden. Active Importance Sampling for Variational Objectives Dominated by Rare Events: Consequences for Optimization and Generalization. In *Proceedings of the 2nd Mathematical and Scientific Machine Learning Conference*, pages 757–780. PMLR, 2022.

[172] Richard S. Sutton and Andrew G. Barto. *Reinforcement Learning: An Introduction*. The MIT Press, Cambridge, Massachusetts, second edition edition, 2018.

[173] Junfeng Wen, Bo Dai, Lihong Li, and Dale Schuurmans. Batch stationary distribution estimation. In *Proceedings of the 37th International Conference on Machine Learning*, ICML'20, pages 10203–10213. JMLR.org, 2020.

[174] Gene H. Golub and Charles F. Van Loan. *Matrix Computations*. The Johns Hopkins University Press, third edition, 1996.

[175] Rose Du, Vijay S. Pande, Alexander Yu. Grosberg, Toyoichi Tanaka, and Eugene S. Shakhnovich. On the transition coordinate for protein folding. *Journal of Chemical Physics*, 108(1):334–350, 1998.

[176] Ao Ma and Aaron R. Dinner. Automatic method for identifying reaction coordinates in complex systems. *Journal of Physical Chemistry B*, 109(14):6769–6779, 2005.

[177] Sergei V. Krivov. On reaction coordinate optimality. *Journal of Chemical Theory and Computation*, 9(1):135–146, 2013.

[178] Weinan E and Eric Vanden-Eijnden. Transition-path theory and path-finding algorithms for the study of rare events. *Annual review of physical chemistry*, 61:391–420, 2010.

[179] Pierre Collett, Servet Martinez, and Jamie San Martin. *Quasi-Stationary Distributions*. Probability and its Applications. Springer Berlin Heidelberg, October 2012.

[180] XuanLong Nguyen, Martin J. Wainwright, and Michael I. Jordan. Estimating divergence functionals and the likelihood ratio by convex risk minimization. *IEEE Transactions on Information Theory*, 56(11):5847–5861, 2010.

[181] Baron Peters and Bernhardt L. Trout. Obtaining reaction coordinates by likelihood maximization. *Journal of Chemical Physics*, 125(5):054108, 2006.

[182] Baron Peters, Gregg T Beckham, and Bernhardt L Trout. Extensions to the likelihood maximization approach for finding reaction coordinates. *Journal of Chemical Physics*, 127(3):034109, 2007.

[183] Hendrik Jung, Roberto Covino, and Gerhard Hummer. Artificial intelligence assists discovery of reaction coordinates and mechanisms from molecular dynamics simulations. *arXiv preprint arXiv:1901.04595*, 2019.

[184] Ashesh Chattopadhyay, Ebrahim Nabizadeh, and Pedram Hassanzadeh. Analog forecasting of extreme-causing weather patterns using deep learning. *Journal of Advances in Modeling Earth Systems*, 12(2):e2019MS001958, 2020.

[185] Hendrik Jung, Roberto Covino, A. Arjun, Christian Leitold, Christoph Dellago, Peter G. Bolhuis, and Gerhard Hummer. Machine-guided path sampling to discover mechanisms of molecular self-organization. *Nature Computational Science*, 3(4):334–345, 2023.

[186] George Miloshevich, Bastien Cozian, Patrice Abry, Pierre Borgnat, and Freddy Bouchet. Probabilistic forecasts of extreme heatwaves using convolutional neural networks in a regime of lack of data. *Physical Review Fluids*, 8(4):040501, 2023.

[187] Chatipat Lorpaiboon, Jonathan Weare, and Aaron R. Dinner. Augmented transition path theory for sequences of events. *Journal of Chemical Physics*, 157(9):094115, 2022.

[188] Sebastian Buchenberg, Norbert Schaudinnus, and Gerhard Stock. Hierarchical Biomolecular Dynamics: Picosecond Hydrogen Bonding Regulates Microsecond Conformational Transitions. *Journal of Chemical Theory and Computation*, 11(3):1330–1336, 2015.

[189] Alberto Perez, Florian Sittel, Gerhard Stock, and Ken Dill. MELD-Path efficiently

computes conformational transitions, including multiple and diverse paths. *Journal of Chemical Theory and Computation*, 14(4):2109–2116, 2018.

[190] Florian Sittel, Thomas Filk, and Gerhard Stock. Principal component analysis on a torus: Theory and application to protein dynamics. *Journal of Chemical Physics*, 147(24):244101, 2017.

[191] Chad W. Hopkins, Scott Le Grand, Ross C. Walker, and Adrian E. Roitberg. Long-Time-Step Molecular Dynamics through Hydrogen Mass Repartitioning. *Journal of Chemical Theory and Computation*, 11(4):1864–1874, 2015.

[192] George A. Khoury, James Smadbeck, Phanourios Tamamis, Andrew C. Vandris, Chris A. Kieslich, and Christodoulos A. Floudas. Forcefield_NCAA: Ab Initio Charge Parameters to Aid in the Discovery and Design of Therapeutic Proteins and Peptides with Unnatural Amino Acids and Their Application to Complement Inhibitors of the Compstatin Family. *ACS Synthetic Biology*, 3(12):855–869, 2014.

[193] Hai Nguyen, Daniel R. Roe, and Carlos Simmerling. Improved generalized Born solvent model parameters for protein simulations. *Journal of Chemical Theory and Computation*, 9(4):2020–2034, 2013.

[194] Peter Eastman, Jason Swails, John D. Chodera, Robert T. McGibbon, Yutong Zhao, Kyle A. Beauchamp, Lee-Ping Wang, Andrew C. Simmonett, Matthew P. Harrigan, Chaya D. Stern, Rafal P. Wiewiora, Bernard R. Brooks, and Vijay S. Pande. OpenMM 7: Rapid development of high performance algorithms for molecular dynamics. *PLOS Computational Biology*, 13(7):e1005659, July 2017. Publisher: Public Library of Science.

[195] Philipp Metzner, Christof Schütte, and Eric Vanden-Eijnden. Illustration of transition

path theory on a collection of simple examples. *The Journal of chemical physics*, 125(8):084110, 2006.

[196] Diederik P Kingma and Jimmy Ba. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*, 2014.

[197] Eric Darve, Jose Solomon, and Amirali Kia. Computing generalized Langevin equations and generalized Fokker-Planck equations. *Proceedings of the National Academy of Sciences*, 106(27):10884–10889, July 2009.

[198] Siqin Cao, Andrés Montoya-Castillo, Wei Wang, Thomas E. Markland, and Xuhui Huang. On the advantages of exploiting memory in Markov state models for biomolecular dynamics. *Journal of Chemical Physics*, 153(1):014105, July 2020.

[199] Dario Lucente, Joran Rolland, Corentin Herbert, and Freddy Bouchet. Coupling rare event algorithms with data-based learned committor functions using the analogue Markov chain. *Journal of Statistical Mechanics: Theory and Experiment*, 2022(8):083201, 2022.

[200] Yilin Meng, Diwakar Shukla, Vijay S. Pande, and Benoît Roux. Transition path theory analysis of c-Src kinase activation. *Proceedings of the National Academy of Sciences*, 113(33):9193–9198, August 2016.

[201] Bodhi P Vani, Jonathan Weare, and Aaron R Dinner. Computing transition path theory quantities with trajectory stratification. *Journal of Chemical Physics*, 157(3):034106, 2022.

[202] Justin Finkel, Dorian S. Abbot, and Jonathan Weare. Path properties of atmospheric transitions: Illustration with a low-order sudden stratospheric warming model. *Journal of the Atmospheric Sciences*, 77(7):2327–2347, 2020.

[203] P. Miron, F. J. Beron-Vera, L. Helfmann, and P. Koltai. Transition paths of marine debris and the stability of the garbage patches. *Chaos: An Interdisciplinary Journal of Nonlinear Science*, 31(3):033101, 2021.

[204] Dario Lucente, Corentin Herbert, and Freddy Bouchet. Committor functions for climate phenomena at the predictability margin: The example of El Niño-Southern Oscillation in the Jin and Timmermann model. *Journal of the Atmospheric Sciences*, 79(9):2387–2400, 2022.

[205] Justin Finkel, Edwin P Gerber, Dorian S Abbot, and Jonathan Weare. Revealing the statistics of extreme events hidden in short weather forecast data. *AGU Advances*, 4(2):e2023AV000881, 2023.

[206] Justin Finkel, Robert J. Webber, Edwin P. Gerber, Dorian S. Abbot, and Jonathan Weare. Data-driven transition path analysis yields a statistical understanding of sudden stratospheric warming events in an idealized model. *Journal of the Atmospheric Sciences*, 80(2):519–534, 2023.

[207] Eric Xia and Martin Wainwright. Krylov-Bellman boosting: Super-linear policy evaluation in general state spaces. In Francisco Ruiz, Jennifer Dy, and Jan-Willem van de Meent, editors, *Proceedings of The 26th International Conference on Artificial Intelligence and Statistics*, volume 206 of *Proceedings of Machine Learning Research*, pages 9137–9166. PMLR, 25–27 Apr 2023.

[208] Rosalind J. Allen, Chantal Valeriani, and Pieter Rein ten Wolde. Forward flux sampling for rare event simulations. *Journal of Physics: Condensed Matter*, 21(46):463102, October 2009.

[209] John D. Russo, She Zhang, Jeremy M. G. Leung, Anthony T. Bogetti, Jeff P. Thompson, Alex J. DeGrave, Paul A. Torrillo, A. J. Pratt, Kim F. Wong, Junchao Xia, Jeremy

Copperman, Joshua L. Adelman, Matthew C. Zwier, David N. LeBard, Daniel M. Zuckerman, and Lillian T. Chong. WESTPA 2.0: High-Performance Upgrades for Weighted Ensemble Simulations and Analysis of Longer-Timescale Applications. *Journal of Chemical Theory and Computation*, 18(2):638–649, February 2022. Publisher: American Chemical Society.

[210] Gabriel Earle and Jonathan Mattingly. Convergence of Stratified MCMC Sampling of Non-Reversible Dynamics, February 2022. arXiv:2111.05838 [math].

[211] D. Aristoff, J. Copperman, G. Simpson, R. J. Webber, and D. M. Zuckerman. Weighted ensemble: Recent mathematical developments. *The Journal of Chemical Physics*, 158(1):014108, January 2023.

[212] Aryeh Warmflash, Prabhakar Bhimalapuram, and Aaron R. Dinner. Umbrella sampling for nonequilibrium processes. *The Journal of Chemical Physics*, 127(15):154112, October 2007.

[213] Alex Dickson, Aryeh Warmflash, and Aaron R Dinner. Nonequilibrium umbrella sampling in spaces of many order parameters. *Journal of chemical physics*, 130(7), 2009.

[214] Alex Dickson, Aryeh Warmflash, and Aaron R Dinner. Separating forward and backward pathways in nonequilibrium umbrella sampling. *Journal of chemical physics*, 131(15), 2009.

[215] Alex Dickson, Mark Maienschein-Cline, Allison Tovo-Dwyer, Jeff R Hammond, and Aaron R Dinner. Flow-dependent unfolding and refolding of an rna by nonequilibrium umbrella sampling. *Journal of Chemical Theory and Computation*, 7(9):2710–2720, 2011.

[216] Juan M. Bello-Rivas and Ron Elber. Exact milestoning. *The Journal of Chemical Physics*, 142(9):094102, March 2015.

[217] Divesh Bhatt, Bin W. Zhang, and Daniel M. Zuckerman. Steady-state simulations using weighted ensemble path sampling. *The Journal of Chemical Physics*, 133(1):014110, July 2010.

[218] Jeremy Copperman and Daniel M. Zuckerman. Accelerated Estimation of Long-Timescale Kinetics from Weighted Ensemble Simulation via Non-Markovian "Microbin" Analysis. *Journal of Chemical Theory and Computation*, 16(11):6763–6775, November 2020. Publisher: American Chemical Society.

[219] Bodhi P. Vani, Jonathan Weare, and Aaron R. Dinner. Computing transition path theory quantities with trajectory stratification. *Journal of Chemical Physics*, 157(3):034106, July 2022.

[220] John Strahan, Justin Finkel, Aaron R. Dinner, and Jonathan Weare. Predicting rare events using neural networks and short-trajectory data, March 2023. arXiv:2208.01717 [physics].

[221] John Strahan, Spencer C. Guo, Chatipat Lorpaiboon, Aaron R. Dinner, and Jonathan Weare. Inexact iterative numerical linear algebra for neural network-based spectral estimation and rare-event prediction, March 2023. arXiv:2303.12534 [physics, stat].

[222] Jérôme Hénin, Tony Lelièvre, Michael R Shirts, Omar Valsson, and Lucie Delemotte. Enhanced sampling methods for molecular dynamics simulations. *arXiv preprint arXiv:2202.04164*, 2022.

[223] Eric Vanden-Eijnden and Maddalena Venturoli. Exact rate calculations by trajectory parallelization and tilting. *The Journal of chemical physics*, 131(4):044120, 2009.

[224] George V. Moustakides. Extension of Wald's First Lemma to Markov Processes. *Journal of Applied Probability*, 36(1):48–59, 1999. Publisher: Applied Probability Trust.

[225] Donna L Luisi, Brian Kuhlman, Kostandinos Sideras, Philip A Evans, and Daniel P Raleigh. Effects of varying the local propensity to form secondary structure on the stability and folding kinetics of a rapid folding mixed / protein: characterization of a truncation mutant of the N-terminal domain of the ribosomal protein L911Edited by P. E. Wright. *Journal of Molecular Biology*, 289(1):167–174, May 1999.

[226] Satoshi Sato, Jae-Hyun Cho, Ivan Peran, Rengin G. Soydaner-Azeloglu, and Daniel P. Raleigh. The N-Terminal Domain of Ribosomal Protein L9 Folds via a Diffuse and Delocalized Transition State. *Biophysical Journal*, 112(9):1797–1806, May 2017.

[227] James A. Maier, Carmenza Martinez, Koushik Kasavajhala, Lauren Wickstrom, Kevin E. Hauser, and Carlos Simmerling. ff14SB: Improving the Accuracy of Protein Side Chain and Backbone Parameters from ff99SB. *Journal of Chemical Theory and Computation*, 11(8):3696–3713, August 2015. Publisher: American Chemical Society.

[228] Gregory D. Hawkins, Christopher J. Cramer, and Donald G. Truhlar. Parametrized Models of Aqueous Free Energies of Solvation Based on Pairwise Descreening of Solute Atomic Charges from a Dielectric Medium. *The Journal of Physical Chemistry*, 100(51):19824–19839, January 1996. Publisher: American Chemical Society.

[229] Gregory D. Hawkins, Christopher J. Cramer, and Donald G. Truhlar. Pairwise solute descreening of solute charges from a dielectric medium. *Chemical Physics Letters*, 246(1):122–129, November 1995.

[230] Robert B. Best, Gerhard Hummer, and William A. Eaton. Native contacts determine protein folding mechanisms in atomistic simulations. *Proceedings of the National Academy of Sciences*, 110(44):17874–17879, October 2013. Publisher: Proceedings of the National Academy of Sciences.

[231] Jae-Hyun Cho, Wenli Meng, Satoshi Sato, Eun Young Kim, Hermann Schindelin, and

Daniel P. Raleigh. Energetically significant networks of coupled interactions within an unfolded protein. *Proceedings of the National Academy of Sciences*, 111(33):12079–12084, August 2014. Publisher: Proceedings of the National Academy of Sciences.

[232] Stuart Lloyd. Least squares quantization in pcm. *IEEE transactions on information theory*, 28(2):129–137, 1982.

[233] Philipp Metzner, Christof Schütte, and Eric Vanden-Eijnden. Illustration of transition path theory on a collection of simple examples. *The Journal of Chemical Physics*, 125(8):084110, August 2006.

[234] Xiang Sherry Li, Brian Van Koten, Aaron R. Dinner, and Erik H. Thiede. Understanding the sources of error in MBAR through asymptotic analysis. *The Journal of Chemical Physics*, 158(21):214107, June 2023.