DISSERTATION

DATA-DRIVEN METHODS FOR COMPACT MODELING OF STOCHASTIC PROCESSES

Submitted by

Mats S. Johnson

Department of Mathematics

In partial fulfillment of the requirements

For the Degree of Doctor of Philosophy

Colorado State University

Fort Collins, Colorado

Summer 2024

Doctoral Committee:

Advisor: David Aristoff

Margaret Cheney
Olivier Pinaud
Diego Krapf

ABSTRACT

DATA-DRIVEN METHODS FOR COMPACT MODELING OF STOCHASTIC PROCESSES

Stochastic dynamics are prevalent throughout many scientific disciplines where finding useful compact models is an ongoing pursuit. However, the simulations involved are often high-dimensional, complex problems necessitating vast amounts of data. This thesis addresses two approaches for handling such complications, coarse graining and neural networks. First, by combining Markov renewal processes with Mori-Zwanzig theory, coarse graining error can be eliminated when modeling the transition probabilities of the system. Second, instead of explicitly defining the low-dimensional approximation, using kernel approximations and a scaling matrix the appropriate subspace is uncovered through iteration. The algorithm, named the Fast Committor Machine, applies the recent Recursive Feature Machine of Radhakrishnan et al. to the committor problem using randomized numerical linear algebra. Both projects outline practical data-driven methods for estimating quantities of interest in stochastic processes that are tunable with only a few hyperparameters. The success of these methods is demonstrated numerically against standard methods on the biomolecule alanine dipeptide.

ACKNOWLEDGEMENTS

Throughout the years in pursuit of this thesis I've met many professors and teachers who have guided and inspired me to achieve what I have. While there are too many who have added enriching discussions or insights to my life, I'd particularly like to thank Tom Asaki, Bala Krishnamoorthy, and particularly my advisor David Aristoff. Together, they represent why I began, and continued, my pursuits in mathematics.

In my time at CSU, I've met many friends who have given me new perspectives on mathematics and on life. To name a few, Justin, Seth, George, Brian, Kelly, Amie, and Andre have left lasting positive imprints on who I am. I'd also like to single out Danny Long who went above and beyond helping me through one of the scarier times of my life.

To those outside CSU, Nish, Mollie, Adam, Amanda, Paige, and Carter, I owe a considerable deal and am always thankful for your friendship.

Finally, I'd like to thank my family, Mom, Dad, and CJ for their love and support through all the years. I wouldn't have the passion for most things in life if not for your involvement and encouragement.

DEDICATION

I would like to dedicate this thesis to my family who have always continued to support me and all the friends I've made along this journey.

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

Chapter 1

Introduction

Stochastic processes play a crucial role in understanding and modeling complex systems encountered in a breadth of scientific pursuits. For instance, they are endemic to problems in biology, computational chemistry, physics, and climate science [1–11]. These processes are characterized by their inherent unpredictability and the high-dimensional nature of the state space in which they evolve. Making accurate predictions of these complex processes involves producing and analyzing enormous datasets, a significant barrier in many fields. To generate this data, molecular dynamics (MD) is often used to simulate the evolution of collections of atoms or particles. However, metastability, the tendency for particles to reside in small regions of space for long times, is yet a further complication researchers often face [6]. In order to achieve adequate sampling of the phase space, more sophisticated techniques such as adaptive multilevel splitting [12], umbrella sampling [13], importance sampling techniques such as weighted ensemble [1], metadynamics [14], milestoning [15], transition path sampling [16], or Markov state models (MSMs) [17, 18] are often necessary.

More challenges await once the data has been collected and a myriad of approaches seeking to improve model accuracy, robustness, or cost have therefore been introduced. Such methods include neural networks [19–22], Markov state models, diffusion maps [23], the variational approach to conformational dynamics [24], or dimension reduction through clustering techniques such as PCA [25], t-SNE [26], or UMAP [27]. Broadly, these ideas seek to represent high-dimensional data in a reduced dimension or through a featurization of the high dimensional space. The underlying principal is that the dynamics exhibit some behavior of interest on a low-dimensional manifold. This observation suggests the use of coarse-graining, a common strategy to combat both the high dimensionality and metastability of stochastic systems. Coarse-graining works by partitioning the original high-dimensional space into discrete macrostates. The resulting dynamics can then be modeled as a continuous time Markov Chain (CTMC) as in Kinetic Monte Carlo models [28] or a

discrete time Markov Chain (DTMC) as MSMs. Coarse-graining offers improved model simplicity at the cost of potentially losing important dynamical information. Additionally, recent works [29] have shown it is possible to uncover the relevant low-dimensional manifold without a predefined mapping.

An alternative approach uses feature functions to reduce the complexity of the model. In place of explicitly defining a low-dimensional map, recent work (such as the recursive feature machine [29]) has shown it is possible to uncover the relevant low-dimensional manifold through iteration. Kernel machines seem to be in conflict with the goal of reducing system size. However, here, the motivation is that many non-linear problems may be made approximately linear by mapping the data to a higher dimensional space [30–32]. By mapping to an even higher dimensional space, the hope is that the original non-linear problem may be approximated linearly [30–32]. Surprisingly, kernel methods do not explicitly construct the higher dimensional space [33, 34]. Instead, a well defined kernel function leads to a reproducing kernel Hilbert space [35] and the associated norm can be defined through the distance of data pairs in the original space. Ultimately, through what is known as the kernel trick, the original data is implicitly mapped to a more useful configuration.

In this thesis, these methods will be used in two projects that address limitations in modeling stochastic processes. In particular, we introduce two, novel, data-driven approaches using Markov Renewal Processes (MRPs) with the Mori-Zwanzig formalism and an iterative scaling scheme using kernel feature machines. For both projects, numerical simulations against standard alternative approaches including MSMs and neural networks are used to demonstrate their effectiveness. These alternative approaches were chosen due to their similar application to reduced order modeling throughout these fields.

In the first project, we take advantage of the more flexible framework provided by MRPs over traditional MSMs by relaxing the assumption that waiting times between state transitions are exponentially distributed. Together with the projection map of coarse-graining, this naturally introduces memory effects to our model. To handle this added complexity, a discrete form of the Mori-Zwanzig formalism is used. The Mori-Zwanzig formulation gives an exact representation of

the high-dimensional dynamics in terms of only a few observable variables. These two techniques produce a model that is highly expressive and more data efficient than typical Markov state models. The improvement will be demonstrated numerically on a small peptide.

The second project explores the use of kernel machines to estimate the committor function, a common problem in computational biology and chemistry [11,36]. The iterative strategy we introduce is based on a combination of the recursive feature machine (RFM) [29] and randomly pivoted Cholesky (RPC) [37]. The RFM is a recent development that uses the estimated gradients of the target function to help generate the distance function in the kernel, as opposed to directly using Euclidean distance. To minimize the resulting least squares loss, RPC is used to efficiently compute the coefficients for the kernel approximation. In addition to our mathematical understanding of iterative feature learning, the application to the committor problem is novel. Fully-connected neural networks will be used to demonstrate the performance of our methodology.

The remainder of this thesis is organized as follows: In Chapter 2 some necessary background on stochastic processes, Markov models, and the challenges of high dimensional data analysis is outlined. Chapter 3 focuses on the first project, which first introduces a decorrelation time to model coarse-grained dynamics as a Markov renewal process. The decorrelation time naturally defines a projection and a discrete form of the Mori-Zwanzig equation is used to build an exact model. The associated Mori-Zwanzig equation is shown to be exact and this chapter finishes by outlining the methodology and discussing numerical performance. Chapter 4 details the RFM and RPC, explains how these emerging ideas are used together, and compares performance with feed-forward neural networks. Additionally, the iterative scheme is shown to produce isotropic gradients in the feature space. In Chapter 5, the main results of the thesis are summarized and potential future directions are discussed.

Chapter 2

Background

2.1 Introduction to Stochastic Processes

In this chapter, some of the foundational concepts at play in the two projects will be introduced. At the core of each project is the goal of modeling or predicting quantities associated with a stochastic process. A stochastic process is a mathematical model that describes the evolution of a system over time. These processes exist across a variety of disciplines including, chemistry, biology, physics, and engineering. Real world data may be obtained through direct observation or from molecular dynamics (MD) simulations. Often, the resulting models may have deterministic and nondeterministic parts, high dimension, metastability, or other complications.

Mathematically, a stochastic process is a time-indexed set of random variables [38]. We will denote such processes $(X(t); t \in \mathcal{T})$ or more simply, $X(t)$, to represent the state of the system at time t. $X(t)$ evolves over states ω in a state space Ω and for convenience, we sometimes write $X_n \doteq X(t_n)$. Throughout this thesis, we assume that the underlying process $X(t)$ has a unique stationary distribution π (2.2) and is a time reversible process. That is, at equilibrium,

$$\pi(i)\mathbb{P}(i,j) = \pi(j)\mathbb{P}(j,i).$$

The phase (or state) space Ω may be continuous (or discrete); for our purposes, we assume $\Omega = \mathbb{R}^d$ where d is the dimension. Similarly, the time set $\mathcal{T}$ maybe be continuous or discrete; our focus is the latter.

One desirable property a stochastic process can exhibit is the Markov property, which simply states that the future state of a stochastic process depends only on the current state [39], commonly referred to as being "memoryless." An everyday example of this property is the game of chess where, once the current state is known, any possible subsequent board position is also known.

Formally, the Markov property states that for a stochastic process $X(t)$ and $\{t_1 < t_2 < \cdots < t_n\}$

$$\begin{aligned} \mathbb{P}\left[X(t) \in d\mathcal{S} \,|\, X(t_1) = x_1, X(t_2) = x_2, \ldots X(t_n) = x_n\right), \, t \geq t_n] \\ = \mathbb{P}\left[X(t) \in d\mathcal{S} \,|\, X(t_n) = x_n, \, t \geq t_n\right]. \end{aligned} \tag{2.1}$$

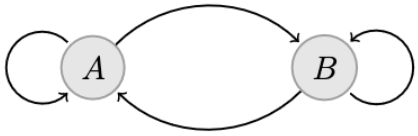
If a stochastic process satisfies the Markov property for all $t \in \mathscr{T}$, it is called a *Markov process* or *Markov chain* when the state space is discrete. Markov processes/chains are actively made use of throughout the finance sector [40–42] and the aforementioned areas of research.

For instance, hidden Markov models, are used in computational biology [43, 44] or machine learning tasks such as speech recognition [45]. A hidden Markov model extends the concept of Markov processes by considering observations that depend on an underlying, unobserved Markov process, allowing for the modeling of systems with hidden states. Hidden Markov models are extensively used beyond the two examples given here [46].

Markov Chain Monte Carlo [47, 48] (MCMC) is another well known application of Markov processes and is often used in high-dimensional problems. The Metropolis-Hastings algorithm [49] is perhaps the most famous MCMC algorithm. Monte Carlo methods rely on a number of trajectories to sample the state space and estimate statistical quantities of interest. The trajectories are constructed using Markov chains as the high-dimensional space makes direct sampling through Monte Carlo estimation infeasible.

Markov processes are desirable for their simplicity as the dynamics are governed by a single Markov kernel or Markov transition matrix. For a Markov transition matrix, $\boldsymbol{T}$, governing the evolution of a Markov chain, the entry $\boldsymbol{T}_{ij}$ is defined as the probability of transitioning from state i to state j. As such, Markov transition matrices satisfy

$$\begin{aligned} &(1)\, \boldsymbol{T}_{ij} \in [0, 1], \quad \forall i, j \in \Omega, \\ &(2)\, \sum_{j \in \Omega} \boldsymbol{T}_{ij} = 1, \quad \forall j \in \Omega. \end{aligned}$$



	A	B
A	0.3	0.7
B	0.8	0.2

Figure 2.1: (Left) A simple two state system which exhibits the Markov property. Possible state transitions are indicated by arrows. (Right) An example transition matrix, notice that the next transition probabilities (rows) solely depend on the current state.

Through $\boldsymbol{T}$, distributions, $\nu(t)$, can be evolved n steps simply by $\boldsymbol{T}^n\nu(t) = \nu(t+n)$. Markov kernels satisfy analogous definitions for continuous state spaces.

For a simple example of a Markov chain, consider the two state system described in Figure 2.1. The next state probabilities require only the current distribution to be given.

High-dimensional problems often require vast datasets to avoid sparsity. Though the original molecular dynamics simulations often satisfy the Markov property, coarse-graining (Section 2.2) often trades the Markov property for simpler representations of the data. This alone does not address all the challenges in modeling the stochastic processes of interest. Often, these processes contain metastable sets which require dynamics to be run for longer times, creating an additional computational burden. The first project addresses this challenge by constructing a model that requires much shorter trajectories than a standard Markov state model. This means that many, shorter trajectories could be run in parallel, reducing computational demand. The second project works directly with the larger dataset but uses randomized linear algebra to subsample useful data points. This improves the space and time complexity of the algorithm constructing the models. While the second project is not expressly hindered by metastability, the first project relies on it.

A stochastic process exhibits *metastability* when it tends to linger in regions of the state space for relatively long times. These regions of the phase space are known as metastable sets. The system may transition between such sets but these transitions are typically much faster than the time spent within the metastable set (see $X(t)$ in Figure 3.1 for a visual reference). Unfortunately, these transitions are usually directly linked to the quantity of interest, such as the folding or unfolding of protein in the committor problem discussed in Chapter 4. As such, these events are considered

rare or fast due to the difficulty in observation through experiments or simulations. Metastability is a common characteristic of problems of MD in climate models [50–53], hydrodynamics [54], and neural evolution [55–57] among many other stochastic systems.

Of particular relevance to Chapter 3, under mild assumptions, sets have what is known as a *quasi-stationary distribution* (QSD). For a set $\mathcal{A}$ to be considered metastable, there should be a large spectral gap. This is indicative of a long lived process in the set $\mathcal{A}$. To understand what a QSD is, we first consider a simpler concept of a stationary distribution. In the context of Markov chains, a stationary distribution satisfies

$$\nu \boldsymbol{T} = \nu, \tag{2.2}$$

for a distribution on the the state space ν and transition matrix $\boldsymbol{T}$. Informally, a stationary distribution is one that does not change with time evolution. If a Markov chain in a countable, discrete space is irreducible and positive recurrent, then a stationary distribution exists and is unique. Irreducibility requires that for any state $i \in \Omega$, every state $j \in \Omega$ is reachable; that is, for each pair $i, j \in \Omega$, there is some time t such that j can be reached from i in exactly t steps with positive probability. Positive recurrence additionally requires that the expected return time for any state i, starting from i, is finite.

To change the language to quasi-stationary distributions, consider a subset $\mathcal{A}$ of the state space Ω. Denote the hitting time of the set $\mathcal{A}^c$, $\mathcal{T}$ and assume $\mathbb{P}_x(\mathcal{T} < \infty) = 1$ for any $x \in \mathcal{A}$; i.e., the process almost surely 'escapes' $\mathcal{A}$. Similar to above, let ν denote a distribution on $\mathcal{A}$ and $\mathcal{B} \subset \mathcal{A}$ be any measurable set. Then ν is a quasi-stationary distribution if for $t \geq 0$

$$\mathbb{P}_\nu(X_t \in \mathcal{B} \,|\, t < \mathcal{T}) = \nu(\mathcal{B}).$$

In other words, the distribution ν remains constant after evolving the system conditional on remaining in the set $\mathcal{A}$. For an in depth review of quasi-stationary distributions, refer to [58, 59].

2.2 Modeling Through Markov Processes

As mentioned above, coarse-graining is often used to reduce the dimensionality of a problem (and combat metastability) but it often sacrifices the Markov property of the original process. Coarse-graining is the process of dividing the underlying state space into discrete sets of *macrostates*. This can be done naively or through principled approaches including PCA [25], TICA [60], PCCA+ [61], or k-means [62, 63]. Ideally, the division of space would group similar states or, understood another way, the macrostates would exhibit metastability. This simplifies the problem while retaining relevant details. The dimension of the system is reduced from the dimension of the underlying space to the number of macrostates. Markov state models (MSMs) are discrete time Markov chains on these macrostates.

A MSM is used to model a dynamical system of N states. Simply, the MSM is a Markov transition matrix as described in Section 2.1. Therefore, there is an inherent assumption that the system satisfies the Markov property. While this is not generally the case, the assumption can approximately hold with the introduction of a lag time, τ. The lag time corresponds to observations of the stochastic process at time increments of τ. The new process $(X(n\tau); n \in \mathbb{N})$, is assumed to have the Markov property if the lag is chosen appropriately.

To choose an appropriate lag time, the transition matrix $\boldsymbol{T}$ is constructed through simple counts for various lags, τ. The eigenvalues of $\boldsymbol{T}_\tau$ (the transition matrix at lag τ) can then be used to get the implied timescales of the system [64] by

$$t_m = -\tau / \ln(\lambda_m(\tau)),$$

where $\lambda_m(\tau)$ is the m^{th} eigenvalue of $\boldsymbol{T}_\tau$ constructed at lag τ and t_m is the implied timescale. To better understand the idea of implied time scales, consider the eigenvalues $\lambda_n(\tau)$ of $\boldsymbol{T}_\tau$. If Markovianity holds, then $\lambda_n(\tau) = e^{-\sigma_n \tau}$, which motivates the definition of implied timescale as $1/\sigma_n$ (note that σ_n represents a rate, so its inverse $1/\sigma_n$ gives the associated time). In practice, the approach is to plot the implied timescales for various lag times. The value of τ is chosen as the

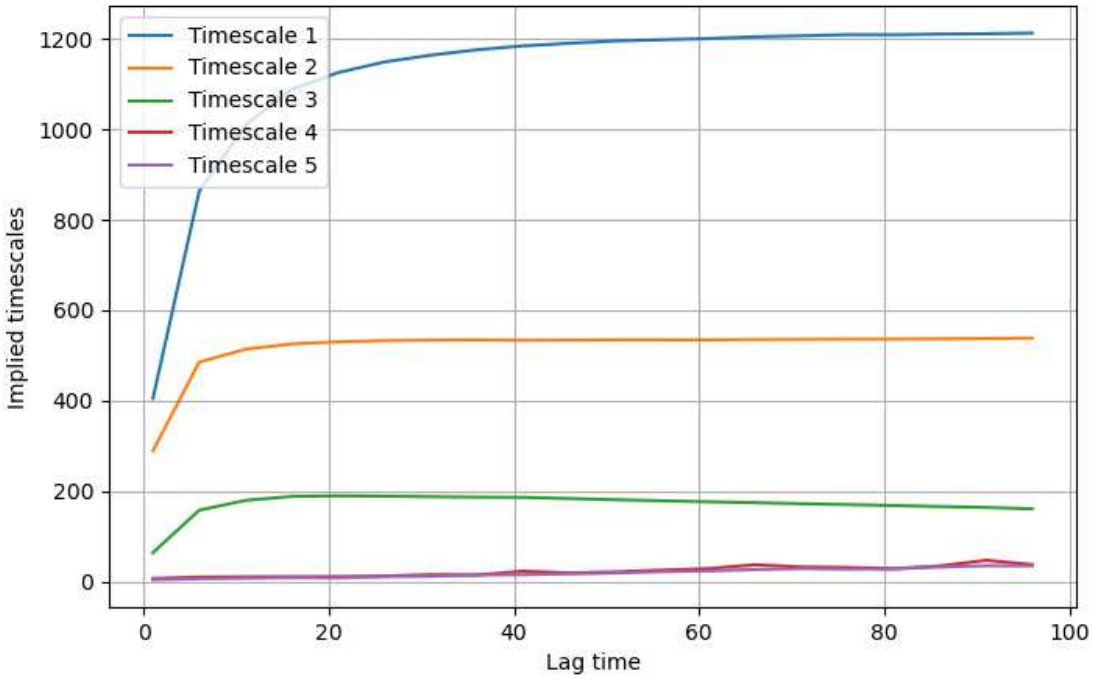


Figure 2.2: An illustration of a plot of implied timescales for a toy problem with the top 5 timescales plotted. Notice that after lag times of approximately 60, the curves appear to level off, indicating the system satisfies the Markov property. This makes 60 a suitable choice for τ.

smallest lag where the implied timescale plot becomes flat (as this implies the Markov property); Figure 2.2 illustrates idea.

In general, the eigenvalue $\lambda_1(\tau) = 1$ is ignored, but the other large eigenvalues communicate useful dynamical information about the system. In particular, the eigenvalues are related to the fast and slow dynamics of the system. This is precisely why the lag time should be chosen where the implied time scales have plateaued, but as small as possible to still capture the faster dynamics.

MSMs have several drawbacks of particular interest to the work in this thesis. They are sensitive to the initial state decomposition, are not immediately applicable to non-Markovian dynamics, and can struggle to capture rare events. In the last case, this can simply be from the lag time being too large to allow the fast transitions to be counted. While it is true that the counts are low for rare events and can lead to inaccuracy in $\boldsymbol{T}_\tau$, we will not attempt to address this shortcoming. Instead, we hope to construct a method which loses less dynamical information through Markov Renewal Processes (MRPs).

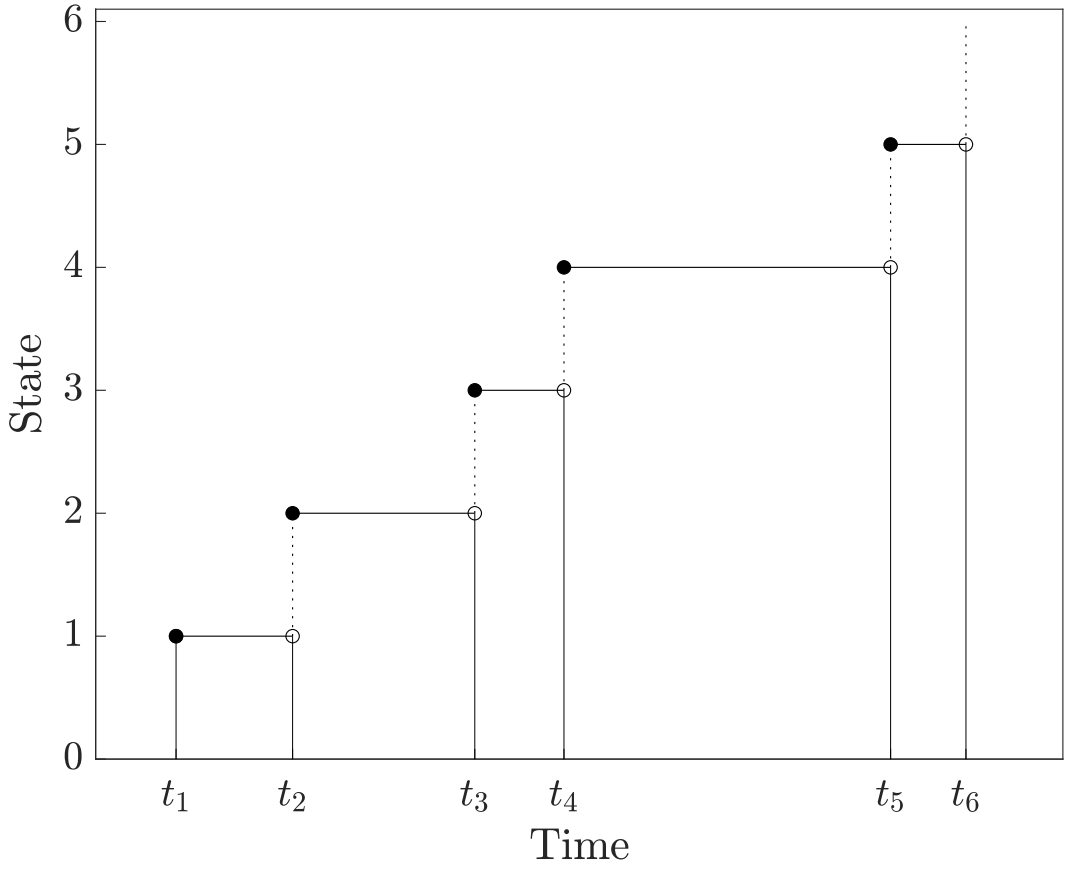


Figure 2.3: An example of a jump process on discrete states. The process remains in state n until time t_n where it jumps to the next state $n+1$. Here, the jump times are denoted by $t_1, t_2, \ldots$

MRPs satisfy a relaxed version of the Markov property. Instead of assuming the stochastic process satisfies (2.1) at all times, a MRP regains the Markov property at its jump times. A jump is simply an event where the process moves from one state to another and a jump time is when this jump occurs (see Figure 2.3). For a stochastic process $(X(t), t \in \mathcal{T})$ with jump times t_i, $i \in \{1, 2, \ldots\}$, $X(t)$ is a Markov renewal property if

$$\begin{aligned} &\mathbb{P}\left[t_{n+1} - t_n < t,\, X_{n+1} = j \,|\, (X_0, t_0), (X_1, t_1), \ldots (X_n, t_n)\right] \\ &\qquad = \mathbb{P}\left[t_{n+1} - t_n < t,\, X_{n+1} = j \,|\, (X_n, t_n)\right] \end{aligned} \tag{2.3}$$

This definition differs from typical CTMC and MSM constructions in that the cumulative distribution of residence times is now a general non-decreasing function in terms of the residence time,

rather than assumed to be exponentially distributed [65]. This more general framing allows the incorporation of non-Markovian dynamics, such as memory effects or time-dependent transition rates, which can be important in various application domains. By using Markov renewal processes, one can capture a wider range of temporal dynamics and potentially obtain more accurate models of the underlying system.

Chapter 3

Markov Renewal Processes and Mori-Zwanzig for Reduced Models

In this project, following a recent insight [65] which introduced Markov renewal processes as a solution to coarse-graining error, a new approach combining Markov renewal processes and the Mori-Zwanzig formalism is proposed. High-dimensional systems carry a large computational resource demand which complicates both data collection and utilization. Coarse graining is a popular technique [66–69] in reduced order modeling but inherently trades more compact models for dynamical information [70,71]. The setting of the problem encountered in this chapter is that of a discrete time Markov chain, or a Markov state model [17]. The loss in accuracy associated with coarse graining is a result of the Markov assumption not holding under coarse graining, improper sampling distributions [72], or if the time scales are not long enough to describe the system [64]. In MSMs, though much research attending to these problems has been carried out, in practice, some error due to the Markov assumption remains [73–75].

The approach carried out in this project does not directly use the coarse-grained data to construct an MSM. Instead, to address the coarse-graining error, the data is reconstituted using a decorrelation time and the Markov assumption is relaxed through modeling the dynamics as a Markov renewal process (MRP). By increasing this decorrelation time, the model's accuracy can be improved in a similar vein to increasing the lag time of a MSM. To contrast the efficacy of the two approaches, our numerical tests demonstrate that the MRP approach retains more dynamical information and can be made arbitrarily accurate without changing the coarse-graining itself.

The increased accuracy of MRPs does carry a computational cost. Through the Markov assumption underpinning MSMs, a transition matrix is estimated from simple counts and future predictions are made by simply taking powers of this matrix. In the MRP construction, transition times between the coarse-grained macrostates are time dependent and the resulting trajectory can

be dependent on its history, i.e. it is no longer "memoryless." To account for memory effects, the Mori-Zwanzig formalism is used to write an exact representation of the model. For the projection described within, a discrete form of the Mori-Zwanzig equation is shown to be exact. This equation leads to a data-driven, non-parameteric method for computing the jump times of the renewal process and the the transition probabilities of the model, whose performance is dictated by only three parameters. The result is an accurate, compact model with a modest increase, over MSMs, in the number of parameters that are learned through least squares.

A justification for this trade-off is the following. One avenue for MSMs to improve their accuracy is to increase the number of states used in the coarse graining. The context this project operates in however, is that the state definitions are fixed. The goal is therefore to introduce a framework that improves modeling performance for a given dataset. Hence, the modest increased cost in complexity is worth the additional accuracy from the MRP approach.

This chapter is organized as follows: The projection defining the Markov renewal process is introduced in Section 3.1 and convergence to this MRP is proven. Section 3.2 introduces the Mori-Zwanzig formalism and associated discrete formulas. Finally, the method which combines these ideas is laid out and performance relative to a Markov state model is demonstrated on a toy problem and alanine dipeptide in Section 3.4.

3.1 The Color Process as a Renewal Process

Throughout this chapter, assume the underlying stochastic process $X(t)$ is a Markov process evolving in a space of *microstates*, which may be continuous or discrete in both time and space. A predetermined coarse-graining then partitions these microstates into a finite set of *macrostates*, $\{I, J, \dots\}$. Note that it is not required for a microstate to be in any macrostate, however, for simplicity, the assumption that such an assignment exists for every microstate will be used throughout this project. The macrostates are therefore assumed to be given and may have been obtained through a number of methods such as PCA [25], PCCA+ [61], TICA [60], or simple rectangular divisions of the state space. The result is two processes: $X(t)$ on the underlying microstates, and

the color process $R(t)$ on the macrostates. $R(t)$ will depend on a macroscopic time step, Δ, and a projection from the $X(t)$ to the macrostates defined by the decorrelation times.

Perhaps the most natural projection would be

$$\pi_{\text{naive}}(X(t)) = I \quad \text{if } X(t) \in I, \tag{3.1}$$

simply defining the macrostate of $R(t)$ by the current mircostate of the underlying process at time t. However, this introduces some complications to the model. First, this definition makes $R(t)$ very sensitive to noise. In interesting problems, there are often metastable regions where $X(t)$ will linger for relatively long periods of time. Between these regions, $X(t)$ may move more erratically across the boundary of neighboring macrostates resulting in many, quick transitions in $R(t)$. Ideally, the macrostates would encompass these parts of the state space to avoid spurious fast transitions as much as possible.

A second concern is that $R(t)$ is entirely non-Markovian due to the coarse-graining. This affects the accuracy of Markov State Models (MSMs) as this is a fundamental assumption for such methods. In turn, this would require longer lag times in building the MSM to avoid these faster dynamics. With longer lag times comes increased variance in state-to-state transitions and the loss of relevant, faster dynamics.

Seeking the aforementioned improvements, using a *decorrelation time* τ_I, a delayed projection for macrostate I is defined as follows,

$$\pi_{\text{delay}}(X(t)) = I \quad \text{if } \forall s \in [t - \tau_I, t],\ X(s) \in I. \tag{3.2}$$

In other words, $X(t)$ must spend time τ_J consecutively in a macrostate J for $R(t)$ to jump from I to J. These jumps occur between distinct macrostates and at integer multiples of the time step Δ, as illustrated in Figure 3.1. Mathematically, after a jump into I at time s, the probability for $R(t)$

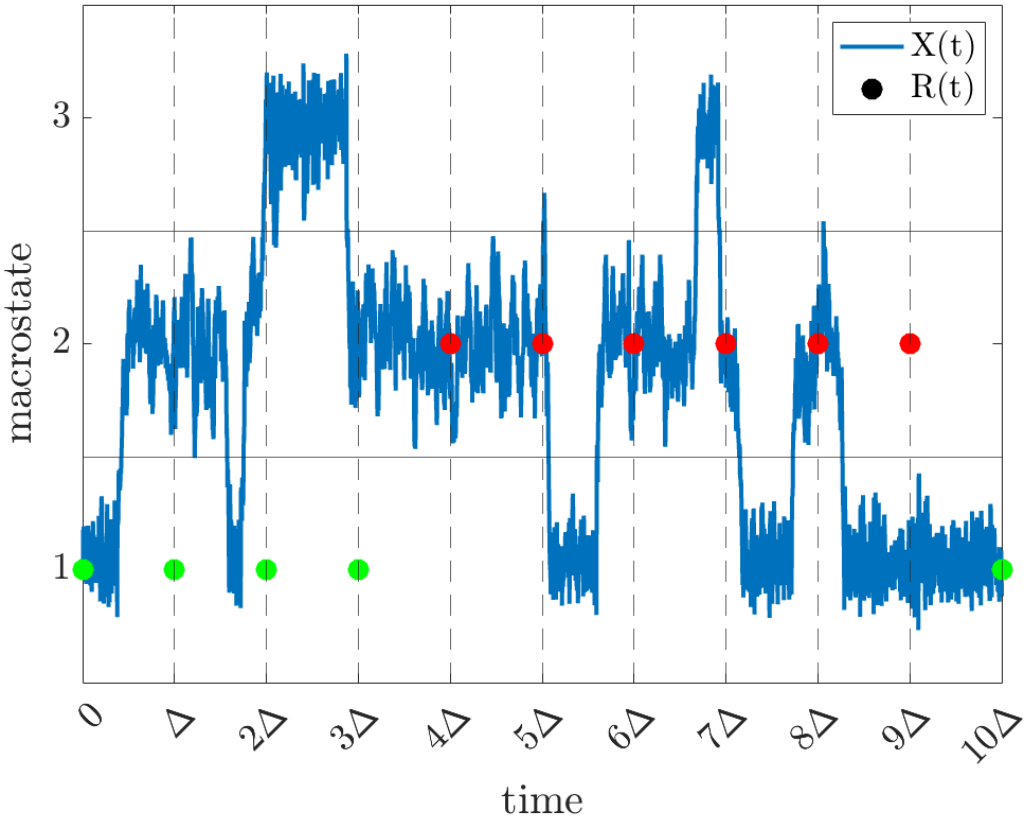


Figure 3.1: Illustration of the underlying process $X(t)$ and the color process $R(t)$, with three macrostates $\{1, 2, 3\}$, and decorrelation time $\Delta = \tau_1 = \tau_2 = \tau_3$. Note that $X(t)$ makes several transitions between macrostates that the color process does not observe. $R(t)$ only jumps to I after $X(t)$ has decorrelated in macrostate I.

to be in J at time $s + t$ can be defined by

$$\mathcal{T}_{IJ}(s, t) = \mathbb{P}(R(s + t) = J \,|\, R(s_-) \neq I,\ R(s) = I), \tag{3.3}$$

where s_- denotes time $s - \Delta$.

With sufficiently large decorrelation times τ_I, the underlying Markov process should reach a local equilibrium in I before $R(t)$ registers a transition. Conceptually, the decorrelation time allows $X(t)$ to lose its memory of how the process entered any particular macrostate. Consequently, $R(t)$ regains the Markov property at jump times making it a Markov Renewal Process (MRP). Theorem 3.1.1 shows that the discretized dynamics $R(t)$ converge to a MRP and in Figure 3.4 this property is (approximately) illustrated on alanine dipeptide.

It is important to note that $R(t)$ will lose any dynamical information on timescales smaller than Δ or decorrelation times τ_I, just as an MSM loses information about dynamics faster than its chosen lag time. This observation stresses the importance of both a good choice of macrostates and associated decorrelation times. Though macrostate definitions are beyond the jurisdiction of the method presented in this thesis, sufficiently troublesome state definitions will negatively impact the modeling capability just as with any other method. The argument is therefore that the methodology presented in Section 3.3 is a better utilization of given data.

A proper choice of the macrostates mitigates the loss of information if the typical dwell time in macrostate I is significantly longer than both τ_I and Δ. Note that this is perhaps not the case in Figure 3.1, and $R(t)$ misses many potentially significant transitions. The color process does converge to a MRP as proven in Theorem 3.1.1 but the choice of decorrelation time will ultimately dictate the utility of the model. Similar to MSM lag time, as $\tau \to \infty$ the method describes perfectly accurate, but ultimately uninformative, models of $R(t)$.

Now, if $R(t)$ is indeed a MRP, the notation can be simplified and a standard transition matrix for each time t can be written as $\mathcal{T}(s,t) = \mathcal{T}(t)$. Furthermore, with a jump probability matrix $\mathcal{P}(t)$ describing the probability of $R(t)$ to jump from I to J in time t as

$$\mathcal{P}_{IJ}(t) = \mathbb{P}(R(s+t) = J \,|\, R(s_-) \neq I,\, R(\hat{s}) = I,\, s \leq \hat{s} < s+t), \tag{3.4}$$

the transition matrices satisfy a renewal equation [76]

$$\mathcal{T}(t) = \sum_{0<s\leq t} \mathcal{P}(s)\mathcal{T}(t-s) + \mathcal{F}(t), \tag{3.5}$$

where $\mathcal{F}(t) \doteq \delta_{I=J} \sum_L \sum_{s>t} \mathcal{P}_{IL}(s)$. Here, the time arguments are integer multiples of Δ and this convention continues for equations associated with $R(t)$ unless otherwise noted. (3.5) is a result observable from total probability where $\mathcal{T}_{IJ}(t)$ denotes the probability of the process jumping from state I to state J at t time units in the future. The first term of the sum describes all possible intermediary paths ($I \to K \to J$) the process could take, assuming a jump to $K \notin \{I, J\}$ occurs at

some time s before jumping to J at time t. The second term represents the probability of remaining in the same state I before until time t.

As alluded to earlier, this definition becomes precise in the limit of large decorrelation times. Informally, a large decorrelation time allows $X(t)$ to forget its initial condition and converge to the quasi-stationary distribution (QSD) of any particular macrostate. More exactly, the QSD of $X(t)$ in I, η_I, describes the distribution of the evolution of $X(t)$ conditioned on the requirement that $X(t)$ remains in I. Mathematically, the QSD is described by

$$\eta_I(\cdot) = \lim_{t\to\infty} \mathbb{P}(X(t) \in \cdot \,|\, X(s) \in I,\, 0 \leq s \leq t), \tag{3.6}$$

or

$$\eta_I(\cdot) = \int \eta_I(dx)\mathbb{P}(X(t) \in \cdot | X(0) = x,\, X(s) \in I,\, s \leq t), \tag{3.7}$$

where the variable x represents the microstates of $X(t)$. With mild assumptions [58, 59], the convergence of the underlying process to the QSD in any macrostate I follows

$$\|\eta_I - \mathbb{P}(X(t) \in \cdot \,|\, X(s) \in I,\, s \leq t)\| \leq c_I \delta_I^t, \tag{3.8}$$

for constants c_I and $\delta_I < 1$ using the total variation norm. The result is that, conditional on remaining in macrostate I, $X(t)$ converges to the η_I at a geometric rate.

Convergence depends on the spectral gap [77] of the matrix describing intra-macrostate transitions T_I. The largest eigenvalue of T_I for a metastable state will be $\lambda_1 \sim 1$ and the next largest eigenvalue λ_2 dictates convergence speed by

$$\delta_I = \frac{\lambda_2}{\lambda_1}.$$

Above follows from the assumption that $X(t)$ is a Markov process on the microstates. Therefore, if $\lambda_2 \sim \lambda_1$ convergence will be slow and a longer decorrelation time will be needed. This might be the case if a macrostate enclosed two metastable regions of the state space. In this case both $\lambda_1 \sim 1$

and $\lambda_2 \sim 1$. On the other hand, if a metastable region were split by the macrostate definitions, it would induce many noisy transitions across the boundary. Here, the assumption that $\lambda_1 \sim 1$ is flawed and the spectral gap would again be small.

With (3.8) in hand, Theorem 3.1.1 shows that

$$\mathcal{T}(s,t) = O(t\delta^\sigma) + \sum_{0<r\leq t} \mathcal{P}(r)\mathcal{T}(s,t-r) + \mathcal{F}(t), \tag{3.9}$$

where $\mathcal{F}_{IJ}(t) = \delta_{I=J} \sum_L \sum_{s>t} \mathcal{P}_{IL}(s)$, $\mathcal{P}(t)$ is the jump probability matrix of a Markov renewal process, and $\delta = \max_I \delta_I$, $\sigma = \min_I \tau_I$, the shortest decorrelation time. It follows that the transition matrices $\mathcal{T}(s,t)$ also converge to $\mathcal{T}(t)$ at a geometric rate in terms of the decorrelation times. From the earlier discussion, the best convergence rates (smallest decorrelation times) occur when the macrostates are chosen to be maximally metastable.

Before the proof, to simplify the notation involved, let

$$E_{IJ}(s,t) = \{R(s+t) = J,\ R(s') = I,\ s \leq s' < s+t\}$$

be the event of switching to from I to J after a time t, starting from time s and

$$E_J(t) = \{R(t) = J\}, \qquad E_J^c(t) = \{R(t) \neq J\}$$

be the events that $R(t) = J$ and $R(t) \neq J$, respectively.

Theorem 3.1.1 (Exactness of the renewal equation)**.** *Assume that each macrostate I has a QSD η_I, and assume that* (3.8) *holds. Define*

$$\mathcal{T}_{IJ}(t) = \mathbb{P}(E_J(s+t) \,|\, E_I(s),\ X(s) \sim \eta_I). \tag{3.10}$$

Then $\mathcal{T}(s,t)$ *defined by* (3.3) *converges to* $\mathcal{T}(t)$ *defined by* (3.10) *as* $\min_I \tau_I \to \infty$. *Moreover, the limit* $\mathcal{T}(t)$ *is the unique solution to the renewal equation* (3.5) *when* $\mathcal{P}$ *is defined by*

$$\mathcal{P}_{IJ}(t) = \delta_{I\neq J}\mathbb{P}(E_{IJ}(s,t) \,|\, E_I(s),\, X(s) \sim \eta_I). \tag{3.11}$$

Proof. Using the law of total probability,

$$\begin{aligned}\mathcal{T}_{IJ}(s,t) &= \mathbb{P}(E_J(s+t) \,|\, E_I^c(s_-),\, E_I(s)) \\ &= \sum_{K\neq I}\sum_{0<r\leq t} \mathbb{P}(E_J(s+t)|E_I^c(s_-), E_{IK}(s,r)) \cdot \mathbb{P}(E_{IK}(s,r)|E_I^c(s_-),\, E_I(s)) \\ &\quad + \delta_{I=J}\mathbb{P}(E_{II}(s,t)|E_I^c(s_-),\, E_I(s)).\end{aligned} \tag{3.12}$$

Let $\sigma = \min_I \tau_I$, and in the notation of (3.8), define

$$c = \max_I c_I, \quad \delta = \max_I \delta_I.$$

Using (3.8), (3.10), and the Markov property of $X(t)$,

$$\begin{aligned}\mathcal{T}_{IJ}(s,t) &= \mathbb{P}(E_J(s+t) \,|\, E_I^c(s_-),\, E_I(s)) \\ &= \int \mathbb{P}(E_J(s+t) \,|\, E_I^c(s_-),\, E_I(s),\, X(s)=x) \cdot \mathbb{P}(X(s) \in dx \,|\, E_I^c(s_-),\, E_I(s)) \\ &= \int \mathbb{P}(E_J(s+t) \,|\, E_I(s),\, X(s)=x)\eta_I(dx) + \epsilon_I \\ &= \mathcal{T}_{IJ}(t) + \epsilon,\end{aligned} \tag{3.13}$$

where $|\epsilon| \leq c\delta^\sigma$. Similar calculations show that

$$\begin{aligned}&\delta_{I\neq K}\mathbb{P}(E_J(s+t) \,|\, E_I^c(s_-), E_{IK}(s,r)) = \mathcal{T}_{KJ}(t-r) + \epsilon \\ &\delta_{I\neq K}\mathbb{P}(E_{IK}(s,r) \,|\, E_I^c(s_-),\, E_I(s)) = \mathcal{P}_{IK}(r) + \epsilon \\ &\mathbb{P}(E_{II}(s,t)|E_I^c(s_-),\, E_I(s)) = \mathcal{F}_{II}(t) + \epsilon,\end{aligned}$$

where each ϵ is different but $|\epsilon| \leq c\delta^\sigma$, and

$$\begin{aligned}\mathcal{F}_{IJ}(t) &= \delta_{I=J}\mathbb{P}(E_{II}(s,t) \,|\, E_I(s),\, X(s) \sim \eta_I) \\ &= \delta_{I=J} \sum_L \sum_{s>t} \mathcal{P}_{IL}(s).\end{aligned}$$

Finally, from the above and (3.12),

$$\mathcal{T}(s,t) = O(t\delta^\sigma) + \sum_{0<r\leq t} \mathcal{P}(r)\mathcal{T}(s+r, t-r) + \mathcal{F}(t). \tag{3.14}$$

Now from (3.13), $\mathcal{T}(s,t)$ converges to $\mathcal{T}(t)$ as $\sigma \to \infty$. Meanwhile, using (3.14), it is readily shown from a standard renewal equation representation (Proposition 4.2 of [76]), that $\mathcal{T}(t)$ is the unique solution to equation (3.5). □

Moving forward, to further simplify the notation, let $\tau = \tau_I$ for every macrostate I and $\tau = \Delta$. This is not required, and the decorrelation times may need to be chosen to be state-dependent in other problems. However, this assumption makes the following analysis and examples easier to discuss and digest.

In summary, the usual Markov assumption in Markov state models can be relaxed to form a Markov renewal process. The renewal process is determined by defining a (global) decorrelation time τ and the representation of the dynamics as this renewal time can be made more accurate by increasing τ. There is an associated trade-off between this accuracy and the information carried in the model. Therefore, τ should be chosen as small as possible while still leveraging the metastability of the macrostates of the system.

3.2 Mori-Zwanzig Formalism

Mori-Zwanzig theory [78,79] is used throughout statistical mechanics [80–85] as well as many other fields [86–89]. The central idea is that the some system $Y(t)$ evolves deterministically in some high-dimensional space but, in practice, the evolution must be described without full knowl-

edge of the system. Immediately, this evokes the idea of coarse-grained dynamics where the evolution of underlying process is described through some set of macroscopic observables. In the context of the framework of this chapter, the macroscopic or resolved variables are the dynamics of the color process and the microscopic or underresolved variables are the dynamics of the underlying process $X(t)$. A common theme in Mori-Zwanzig theory is readily apparent here as well, the underresolved variables may meaningfully influence the system despite no direct method of measuring this impact. Recall, $R(t)$ simply describes the most recent macrostate $X(t)$ has decorrelated within, not necessarily the current macrostate in which $X(t)$ resides. To address these challenges, Mori [78] and Zwanzig [79] developed projection based methods which seek to describe the evolution of the system using only the observable variables.

To first gain insight into the Mori-Zwanzig formalism, consider the following two-dimensional ODE system

$$x' = ax + by$$
$$y' = px + qy.$$

Using the integrating factor $y_0 e^{qt} + p\int_0^t xe^{-q(t-s)}\,ds$ a solution for x' is obtained as

$$x' = ax + bp\int_0^t xe^{-q(t-s)}\,ds + by_0e^{qt}.$$

Notice that the dimension of the system has been reduced to only depend on the variable x'. In the context of Mori-Zwanzig theory, x' would be a *resolved* or *observable* variable and y' would be an *underresolved* or *unobservable* variable. By reducing the dimensionality, an integral term (commonly, the 'memory' term) is introduced to capture the effects of the underresolved variables and the final term represents the noise of the system. These observations are nicely paralleled by the traditional Mori-Zwanzig equation (3.15) introduced below.

In traditional Mori-Zwanzig theory, the dimension reduction is a result of some projection P. Recall a mapping P is a projection if it satisfies $P^2 = P$. The operator P projects the full system

dynamics onto the resolved variables and implicitly defines the orthogonal dynamics $Q = I - P$. An abbreviated description of the Mori-Zwanzig equation is given here, for a full discussion see [78, 79, 82, 90].

Consider some nonlinear dynamical system

$$\frac{d}{dt}X(t) = G(X(t)), \qquad X(0) = x_0,$$

evolving in $\mathbb{R}^d$. Commonly, the Louiville operator $\mathcal{L}$ is invoked here to describe how the evolution of the full system acts on an observable $g : \mathbb{R}^d \to \mathbb{R}$ by

$$e^{t\mathcal{L}} = g(X(t)).$$

Let $A = P\mathcal{L}$ and $B = Q\mathcal{L}$, then the traditional Mori-Zwanzig equation is obtained through Dyson's identity [83, 91] to be

$$\frac{d}{dt}\left(e^{t\mathcal{L}}g\right) = e^{t\mathcal{L}}P\mathcal{L}g + \int_0^t e^{(t-s)\mathcal{L}}\left[P\mathcal{L}e^{eQ\mathcal{L}}Q\mathcal{L}g\right]\, ds + e^{tQ\mathcal{L}}Q\mathcal{L}g. \tag{3.15}$$

The first expression of (3.15) is known as the Markovian or streaming term, the integral is the memory term due to its dependence on previous values of the resolved variables, and the final expression is the system noise or the fluctuation term. The Mori-Zwanzig formalism above is then well suited to describing the evolution of $R(t)$. The memory effects of the system are exactly described by the memory kernels in the integral and the Markov property associated with MRPs suggests that truncating the integral is well-motivated. This last observation is important as it allows for accurate estimations using Mori-Zwanzig theory without requiring computationally prohibitive calculations.

Clearly, the definition of P is instrumental in defining the Mori-Zwanzig equation. Classical projections were introduced by Mori [78], Zwanzig [79], and Chorin et al. [82]. Mori described a linear, finite rank projection that simply seeks to describe the dynamics of the system through

directly projecting the full dynamics of a system onto a subspace of spanned by the resolved variables.

Zwanzig's projection operator instead acts on the probability distribution of the system. In taking the conditional expectation of some quantity of interest given the resolved variables, it minimizes the contributions from the underresolved variables.

Finally, the projector given by Chorin et al. [82] was introduced in the context of fluid dynamics, designed to separate the system into coarse-scale and fine-scale components. The coarse-scale components are those which are slowly varying through system evolution and are considered to be the resolved variables.

In this chapter, the projection used is described by (3.2) and most closely aligns with Zwanzig's projection as both minimize the mean squared error of the approximation, albeit in different ways. Furthermore, this project is concerned with discrete time evolution and does not directly invoke the heavy machine of (3.15). Thus, the Mori-Zwanzig equation used within is described in Lemma 3.3.1 in service of proving Theorem 3.3.2 of the following section. Furthermore, by multiplying on the right by the indicator functions χ_J the noise term vanishes. This leads to a simpler loss term (3.25) when solving for the memory kernels.

3.3 Nonparametric Estimation of Transition Probabilities

From the previous section, the Mori-Zwanzig formalism allows for dimension reduction by separating the system into observed and unobserved variables. These two sets of variables define the projected and orthogonal dynamics of the system using only the observed variables; here, roughly the time-delayed macrostate residencies. Mori-Zwanzig theory grants the nice equation

$$\mathcal{T}(t) = \sum_{0<s\leq t} \mathcal{K}(s)\mathcal{T}(t-s), \tag{3.16}$$

where $\mathcal{T}$ is the time-dependent transition matrix as above and $\mathcal{K}(s)$ are the memory kernels as introduced in (3.15). Efficient estimation of these memory kernels is key for the method detailed herein.

Remark 3.1. Equation (3.16) was originally derived as an approximation of a continuous-time Mori-Zwanzig equation [92] and is not the sole description of discrete-time Mori-Zwanzig equations [90, 93]. However, Theorem 3.3.2 shows that (3.16) is exact given the projector (3.2).

Remark 3.2. Before proving the exactness of the discrete-time Mori-Zwanzig equation, it is prudent to precisely define the Mori-Zwanzig operator involved. The form of (3.16) is a result of choosing a Mori-Zwanzig projector that has only one nonzero memory kernel if $R(t)$ is Markovian. If $R(t)$ is nearly Markovian, the projector lends itself to memory kernels that decay quickly in time. This is extraordinarily useful in how estimates of (3.16) are obtained from data (see the discussion surrounding (3.25)).

For now, consider the augmented Markov chain $(X(t), R(t), C(t))$ where $C(t)$ counts the consecutive time $X(t)$ has spent in the current macrostate. The augmented Markov chain operates on states (x, I, s) where $X(t) = x$, $R(t) = I$, and $0 \leq s < \tau$ (or more precisely τ_J if $X(t) \in J$). For the augmented Markov chain starting at $(X(0), R(0), C(0)) = (x, I, s)$, let $\mathbb{P}^{x,I,s}$ denote the probability for this chain and define the associated Markov kernel T by

$$T(x, I, s; dy, J, t) = \mathbb{P}^{x,I,s}\left[(X(\tau), R(\tau), C(\tau)) = (dy, J, t).\right] \tag{3.17}$$

For functions $f = f(x, I, s)$, it makes sense to define how more general kernels and a projector operator works on these functions. Let $S(x, I, s; dy, J, t)$ be a kernel without the non-negativity and unit normalization requirements of T. Then a projector P satisfies

$$\begin{aligned} Sf(x, I, s) &\doteq \int \sum_{J,t} S(x, I, s; dy, J, t) f(y, J, t) \\ Pf(x, I, s) &\doteq \int \eta_I(dz) f(z, I, \tau_I). \end{aligned} \tag{3.18}$$

Though the lemma below applies more generally, the following discussion is framed in the context of the Markov kernel T of (3.17) and projector (3.18) above.

Lemma 3.3.1. *For any projector P and its complementary projector $Q = \mathrm{Id} - P$, where* Id *is the identity mapping, P acts on a transition kernel T by*

$$PT^n = \sum_{m=1}^{n} K(m) PT^{n-m} + F(n), \tag{3.19}$$

where $K(n) = PT(QT)^{n-1}$ and $F(n) = PT(QT)^{n-1}Q$.

Proof. From the self-evident equations

$$PT^{n+1} = PTPT^n + PTQT^n \tag{3.20}$$

$$QT^{n+1} = QTPT^n + QTQT^n. \tag{3.21}$$

Using induction in (3.21), it follows that

$$QT^n = \sum_{m=1}^{n} (QT)^m PT^{n-m} + (QT)^n Q.$$

The proof is completed by plugging the above into (3.20). □

All the pieces are now in place to prove the exactness of the discrete-time Mori-Zwanzig equation (3.16). Throughout the proof below, the indicator functions $\chi_J(x, I, s) = \delta_{I=J}$ are used.

Theorem 3.3.2 (Exactness of MZ equation)**.** *Let $K(n)$ be as in Lemma 3.3.1, where P is the projector from* (3.18) *and T is defined in* (3.17)*. Define*

$$\mathcal{K}_{IJ}(n\tau) := K(n)\chi_J(x, I, s). \tag{3.22}$$

Then, with $\mathcal{T}(t)$ as in (3.10)*,*

$$\mathcal{T}(n\tau) = \sum_{m=1}^{n} \mathcal{K}(m\tau)\mathcal{T}((n-m)\tau). \tag{3.23}$$

Proof. Multiply (3.19) on the right by $\chi_J(x, I, s)$. Note that $P\chi_J = \chi_J$, and $Q\chi_J = 0$ and $F(n)\chi_J(x, I, s) = 0$. Thus,

$$PT^n\chi_J(x, I, s) = \sum_{m=1}^{n} K(m)PT^{n-m}\chi_J(x, I, s). \tag{3.24}$$

Recalling $\mathcal{T}(t)$ defined in (3.10), it follows

$$\begin{aligned} PT^n\chi_J(x, I, s) &= \int \eta_I(dx)T^n\chi_J(x, I, \tau_I) \\ &= \int \eta_I(dx)\mathbb{E}^{x,I,\tau_I}[\chi_J(X(n\tau), R(n\tau), C(n\tau))] \\ &= \int \eta_I(dx)\mathbb{P}^{x,I,\tau_I}[R(n\tau) = J] \\ &= \mathcal{T}_{IJ}(n\tau). \end{aligned}$$

Below, write $S_m = T(QT)^{m-1}$, and note that $PT^{n-m}\chi_J(y, L, t)$ does not depend on y or t. Thus,

$$\begin{aligned} &\sum_L \mathcal{K}_{IL}(m\tau)\mathcal{T}_{LJ}((n-m)\tau) \\ &= \sum_L K(m)\chi_L(x, I, s)PT^{n-m}\chi_J(y, L, t) \\ &= \sum_L \int \eta_I(dz) \left[\int \sum_t S_m(z, I, \tau_I; dy, L, t)\right] PT^{n-m}\chi_J(y, L, t) \\ &= \int \eta_I(dz) \left[\int \sum_{L,t} S_m(z, I, \tau_I; dy, L, t)PT^{n-m}\chi_J(y, L, t)\right] \\ &= K(m)PT^{n-m}\chi_J(x, I, s). \end{aligned}$$

Finally, (3.23) is obtained by plugging the above into (3.24). □

With the precise definition of the memory kernels in Theorem 3.3.2, consider Remark 3.2 regarding the chosen operator's effect on memory kernel decay. When the color process is Markovian, the underlying process instantly decorrelates upon entering a new macrostate, i.e. $\tau = 0$. On the surface, this feels the same as the definition given by π_{naive} in (3.1). However, here, there is the additional assumption that $X(t)$ immediately follows the QSD upon entering a new macrostate. Therefore, $R(t)$ is Markovian as well.

In this case, $\mathcal{T}(n\tau) = \mathcal{T}(\tau)^n$ and from (3.16), $\mathcal{K}(\tau) = \mathcal{T}(\tau)$. From induction it follows that $\forall n > 1$, $\mathcal{K}(n\tau) = 0$. This idea is made precise below.

Theorem 3.3.3. *Suppose that $\tau_I = 0$ for all I and that*

$$T(x, I, s; dy, J, t) = \int \eta_I(dz) T(z, I, \tau_I; dy, J, t).$$

Then $\mathcal{K}(n\tau) = 0$ for $n > 1$.

Proof. The assumption on T implies that $PT = T$, so $QT = 0$ by definition and the result follows directly from the formula

$$\mathcal{K}_{IJ}(n\tau) = PT(QT)^{n-1}\chi_J(x, I, s).$$

□

While Mori-Zwanzig theory offers a compact, exact formula to precisely describe the evolution of $R(t)$ through (3.16), a more practical approach is to introduce cutoff times for the memory kernels t_{mem} (i.e., the number of memory kernels used) and transition matrices t_{max}. This approach nicely coordinates with the framework of the problem. The new parameter t_{max} can be thought of as a time horizon for the transition matrices $\mathcal{T}(t)$. While increasing t_{max} intuitively should increase the accuracy in estimating $\mathcal{T}(t)$, there is an important consideration. In the case of many short trajectories, choosing t_{max} too close to the length of such trajectories fundamentally lowers the transition counts. Practically, this observation can be seen in (3.25) where there is a balance

to be struck between the number of terms in the loss function and having sufficiently descriptive samples to construct $\mathcal{T}(t)$.

One interpretation of the memory kernels $\mathcal{K}(t)$, is they describe how $X(t)$ transitions between macrostates without fully decorrelating (Figure 3.1 provides a useful visual). Therefore, the memory kernels are not immediately available due to their dependence on the orthogonal dynamics of the unobserved variables. To obtain estimates of $\mathcal{K}(t)$, the transition matrices obtained from data $\mathcal{T}(t)$ are used with (3.16) in the loss function below.

The aforementioned t_{max} and t_{mem} are useful in easing the computational burden of the loss function as they control the number of parameters to be optimized. The least-squares loss function is then

$$\mathcal{L}(\mathcal{K}) = \sum_{t \leq t_{max}} \left\| \mathcal{T}(t) - \sum_{0 < s \leq \min\{t, t_{mem}\}} \mathcal{K}(s)\mathcal{T}(t-s) \right\|^2, \tag{3.25}$$

where $\| \cdot \|$ is the Frobenius norm. By setting the gradient of $\mathcal{L}(\mathcal{K})$ equal to zero, the following symmetric positive semidefinite linear system is observed, from which, the memory matrices may be directly solved for

$$\sum_{0 < s \leq t_{mem}} \mathcal{K}(s)\mathcal{A}(s,t) = \mathcal{B}(t), \quad 0 < t \leq t_{mem}. \tag{3.26}$$

Above $\mathcal{A}$ and $\mathcal{B}$ are the correlation matrices

$$\begin{aligned} \mathcal{A}(s,t) &= \sum_{r \leq t_{max}} \mathcal{T}(r-s)\mathcal{T}(r-t)^T, \\ \mathcal{B}(s) &= \sum_{r \leq t_{max}} \mathcal{T}(r)\mathcal{T}(r-s)^T, \end{aligned} \tag{3.27}$$

and $\mathcal{T}(s) = 0$ for $s < 0$, by convention. The result is a system that is both simple to build and work with and is generated entirely from data and only a few hyperparameter choices. It is also possible to include regularization terms, such as ridge regression, if desired. In the case of ridge regression penalizing the Frobenius norm of the memory kernels results in a slower decay of the memory kernels.

Once an estimate for the memory kernels is at hand, the transition probabilities for any time, including those greater than t_{max} can be estimated through

$$\mathcal{T}(t) \approx \sum_{0<s\leq \min\{t,t_{mem}\}} \mathcal{K}(s)\mathcal{T}(t-s). \tag{3.28}$$

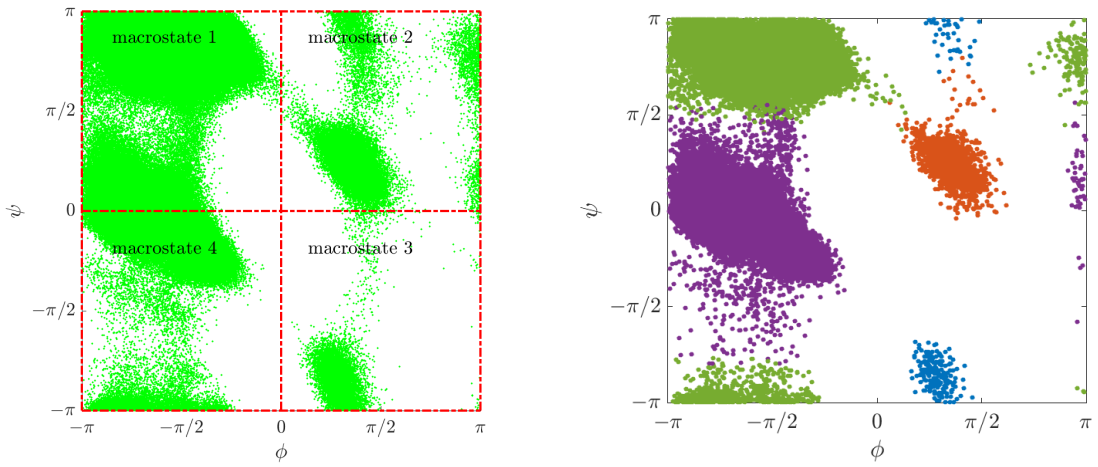
Furthermore, from these estimates of $\mathcal{T}(t)$, estimates of $\mathcal{P}(t)$ can be computed through the renewal equation (3.5). Unlike simple MSM approaches to the problem, this explicit reconstruction of the jump probabilities allows for additional analysis, such as running dynamics.

In summary, after choosing a decorrelation time τ, the number of memory kernels t_{mem}, and the length of trajectories t_{max}, entirely data-driven estimates of $\mathcal{T}(t)$ for $t \leq t_{max}$ are obtained. In turn, this defines the correlation matrices $\mathcal{A}$ and $\mathcal{B}$ as in (3.27) which allows the loss (3.25) to be minimized by solving the linear system (3.26). Finally, with estimates of $\mathcal{K}(t)$, (3.28) can be used to extrapolate time-dependent transition probabilities as far into the future as desired.

As a final remark, the results in the following section demonstrate that only a small number of memory kernels are needed to obtain promising results, see Figure 3.6. If the estimation involves k memory kernels and N macrostates, only N^2k parameters need to be optimized. For problems with small k, this is not computationally cumbersome. Nonetheless, recent randomized numerical linear algebra techniques such as Randomly Pivoted Cholesky [37] may allow this method to be applied to larger problems. Additionally, numerical tests showed that solving (3.16) directly was computationally inferior to solving a lower triangle version of the linear system.

3.4 Numerical Results and Discussion

To demonstrate the efficacy of our method, we performed numerical tests on alanine dipeptide, a common test problem in literature [20, 94–98]. Its structure is similar to more complicated proteins but is described simply by ten backbone atoms. Each atom exists in $\mathbb{R}^3$ and the resulting dynamics are generated in this $30-$dimensional space. However, the evolution of the biomolecule is well described by two dihedral angles ϕ and ψ. A molecular dynamics trajectory [65] of length



(a) Rectangular **(b)** PCCA

Figure 3.2: Representing a molecular dynamics trajectory of alanine dipeptide in ϕ-ψ space. Left: the macrostates (in red) evenly divide the state space into four rectangular regions. Right: The macrostates (green, purple, red, and blue) are defined using PCCA. The states defined through PCCA are highly metastable.

$\sim$ 70 ms sampled every 2 ps was used and subsequently mapped into $\phi - \psi$ space in $\mathbb{R}^2$. This longer trajectory was subsampled to simulate data in the form of many short trajectories.

Two sets of macrostates are used to illustrate the method's performance against a MSM. These sets help elucidate the earlier discussion regarding metastability and demonstrate how significant it is that the states are well defined. Nonetheless, in both cases, the approach laid out in this chapter significantly outperformed the MSM in accuracy and fidelity. One set of states is defined simply by dividing the space into four equal rectangles. The resulting states are entirely unphysical and do not exhibit metastability. In contrast, the second set of states is defined through PCCA and are highly metastable. Figure 3.2 depicts the sets choices of macrostates outlined above.

As seen in Figure 3.3, the Markov models constructed using the "good" PCCA states leave room for improvement. There is significant departure from the reference using the shorter lag time and the longer lag time does not have the fidelity to capture the faster but still relevant transitions of the system.

Throughout, the same decorrelation time, τ ps, is used for each macrostate; though it is different between the two datasets. In choosing τ, there is a balance between having a long enough decorrelation time for for $R(t)$ to meaningfully approach a MRP while keeping τ smaller than

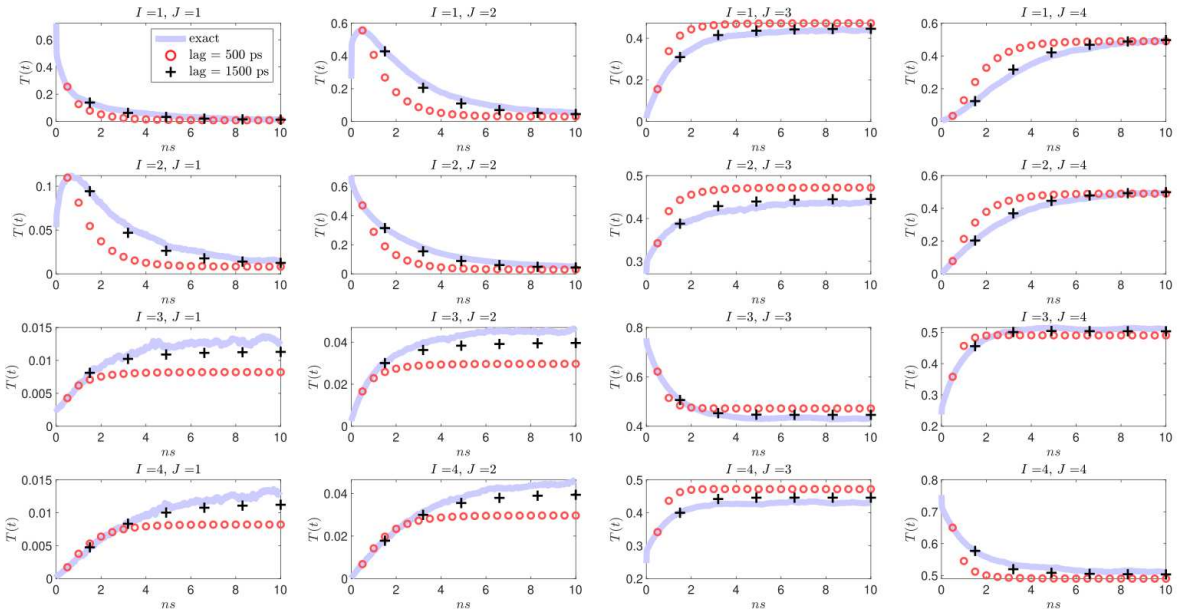


Figure 3.3: Building a Markov model estimate of $T_{IJ}(t)$ for alanine dipeptide, using states defined through PCCA (Figure 3.2.b). The Markov model is considerably less accurate than the Markov renewal approach (see Figure 3.5) at short lag times. While the Markov model estimates can be accurate, the lag time must be taken longer than the mean longest transitions of the system. This is because a simple coarse-graining of $X(t)$ into these states is not sufficiently Markovian. Each Markov model is based on a single transition matrix, computed from simple counts of transitions of $X(t)$ between macrostates at the specified lag time. Transistions are estimated at later times by taking powers of this transition matrix as described in Section 2.2.

the typical transition time between states. Both τ and t_{max} share this property where practical constraints naturally enforce a 'Goldilocks zone' in parameter choice. The practical concerns in choosing t_{mem} are largely computational contraints. Notice τ is unique among the hyperparameters. Where t_{max} and t_{mem} are used to obtain the memory kernels that generate an approximation of $R(t)$, τ, together with the macrostates, fundamentally defines the color process itself. To choose τ, plots similar to Figure 3.4 were generated, and the smallest value that visually approximated a MRP was selected. A more grounded approach to choosing τ is left to future considerations, though good results were attainable "by eye."

Once a decorrelation time was chosen, $R(t)$ was constructed as in 3.1 and the method as described in 3.3 was implemented. To validate the efficacy of our approach, the MD trajectory was split into training and test sets. Half of the data was used to create an estimate $\hat{\mathcal{T}}(t) \approx \mathcal{T}(t)$ and the other half was used as a reference. In the reference set, $\mathcal{T}(t)$ and $\mathcal{P}(t)$ were estimated through

simple counts of transitions and used as ground truth. Figure 3.5 shows how the estimation compares to the reference and Figure 3.6 shows the error in estimating $\hat{\mathcal{P}}(t) \approx \mathcal{P}(t)$. To quantify the estimation of the jump probabilities, a variation of the Cramer-von Mises criterion [99]

$$\text{Error} = \sum_{I,J} \int_0^\infty \left(\int_0^t \frac{[\mathcal{P}_{IJ}(s) - \hat{\mathcal{P}}_{IJ}(s)]}{Z_{IJ}} ds \right)^2 \frac{\mathcal{P}_{IJ}(t)}{Z_{IJ}} dt, \tag{3.29}$$

was used where $Z_{IJ} = \int_0^\infty \mathcal{P}_{IJ}(t)\, dt$. Figures 3.5 and 3.6 show that our method outlined in Section 3.3 gives good agreement with the reference using only a few memory kernels.

3.4.1 Simulation Details

Insofar, discussions surrounding how τ, t_{mem}, and t_{max} were or might be chosen have been largely avoided. Briefly, t_{mem} and t_{max} should be chosen as large as possible to accurately parameterize the model, subject to the constraints of the data itself and computational limits. As previously mentioned, τ should be just large enough that $R(t)$ is approximately a renewal process.

Expanding on the earlier discussions of τ, in theory, increasing τ should increase the accuracy of the model. However, due to the definition of $R(t)$, if τ is on the same order as the typical dwell time in macrostates, meaningful transitions will not be counted in building $R(t)$. It is for this reason that the ideal states should be highly metastable as this encourages the underlying Markov process to remain in place for long periods of time. This gives a larger buffer when choosing τ such that it approximates a MRP. As the macrostates are immutable, this places the emphasis on carefully choosing the decorrelation time. Figure 3.5 shows that it is possible to find such a τ that still contained meaningful data. If the decorrelation time is too high, it is possible that the model will be extremely accurate in its estimation of $\mathcal{T}(t)$ but unsubstantial in the information it carries. For instance, an infinite decorrelation time would be perfectly accurate $R(t) = R(0)$, $\forall t \geq 0$, but convey no dynamical statistics.

As previously mentioned, t_{max} requires some attention. If the data is many short trajectories of $X(t)$, this naturally imposes an upper bound on t_{max}. In these experiments, after determining the

number of memory kernels t_{mem}; the value $t_{max} = 2 \times t_{mem}$ was found to produce consistently good approximations. In practice, these parameters could instead be chosen in a more systematic way, such as a grid search or cross-validation.

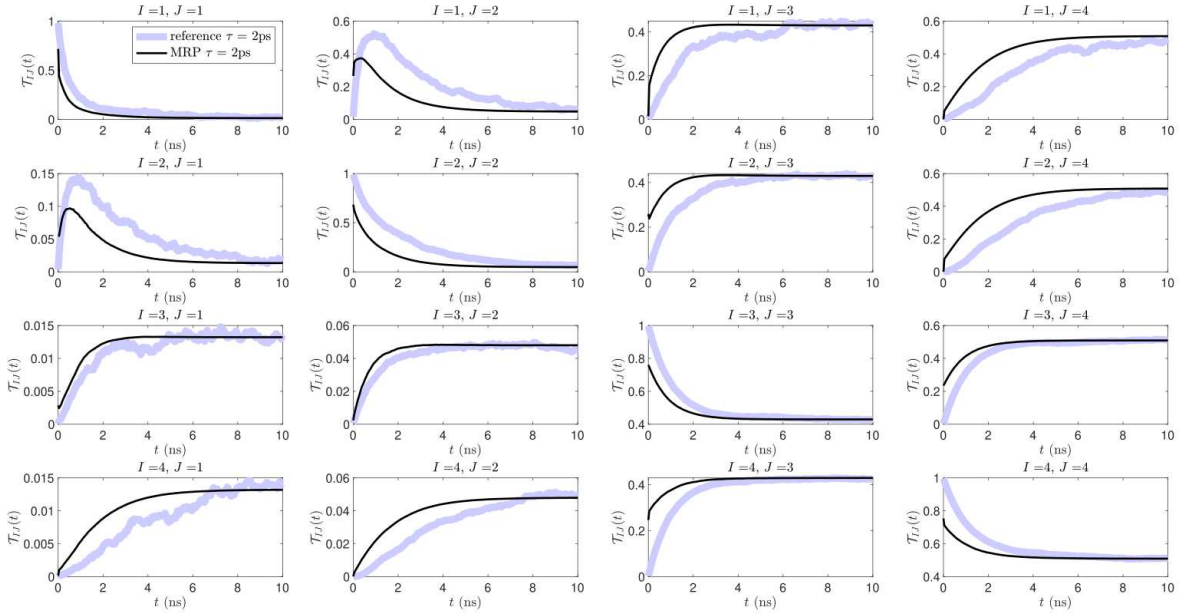
3.4.2 Discussion

This chapter presented a methodology that led to a compact model well suited to problems with metastability where the Markov assumption is costly or improper. Instead of relying on increasing the number of states to improve model accuracy, one avenue taken in Markov state models, a simple decorrelation parameter can be increased to yield arbitrarily accurate representations of the data. While it is true that large decorrelation times may produce uninformative models, the same is true for excessively large lag times of MSMs. In these cases, the data does not allow approaches that aim to model transition times or jump probabilities to be successful.

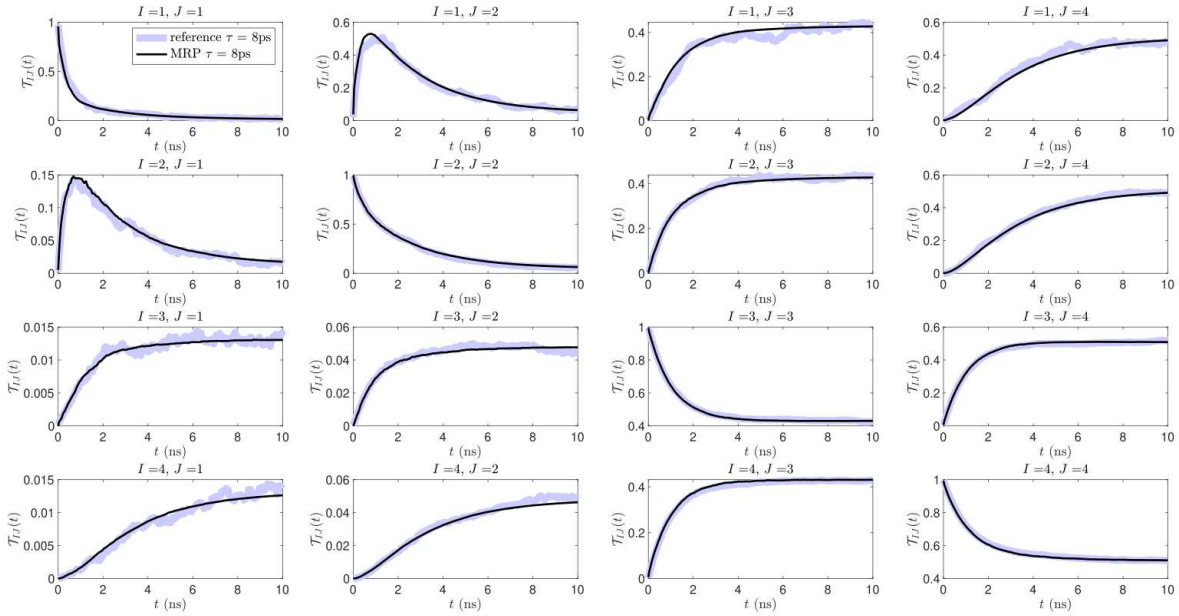
The decorrelation time, τ is highly motivated by the presence of metastability and defines the color process $R(t)$. Increasing τ, results in $R(t)$ approaching a Markov renewal process where Mori-Zwanzig theory is used to account for memory effects. Numerical tests showed that the time horizon the of memory kernels can be taken to be quite short, which led to good estimates below the order of the fastest transition times of the system. The Mori-Zwanzig piece introduced two new parameters that led to a modest increase in model complexity (the number of parameters) but was still easily handled by linear regression. With advances in randomized numerical linear algebra [37], larger approximations for more advanced problems may be tractable.

Finally, we showed that our approach is competitive with the standard approach through MSMs. The more complex MRP model we introduced was more accurate and contained more dynamical information on the test problem of alanine dipeptide. Though mathematically rigorous choices for the hyperparameters τ, t_{mem}, and t_{max} are not given, we discussed how they might be obtained in practice. With a couple considerations, our approach constructs compact and accurate models even with sub-optimal state definitions. Within this chapter, we assumed the dataset was unchangable

but it could also be the case that the optimal coarse-graining of high-dimensional data is not easily defined. Therefore, our method is both mathematically motivated and practically useful.

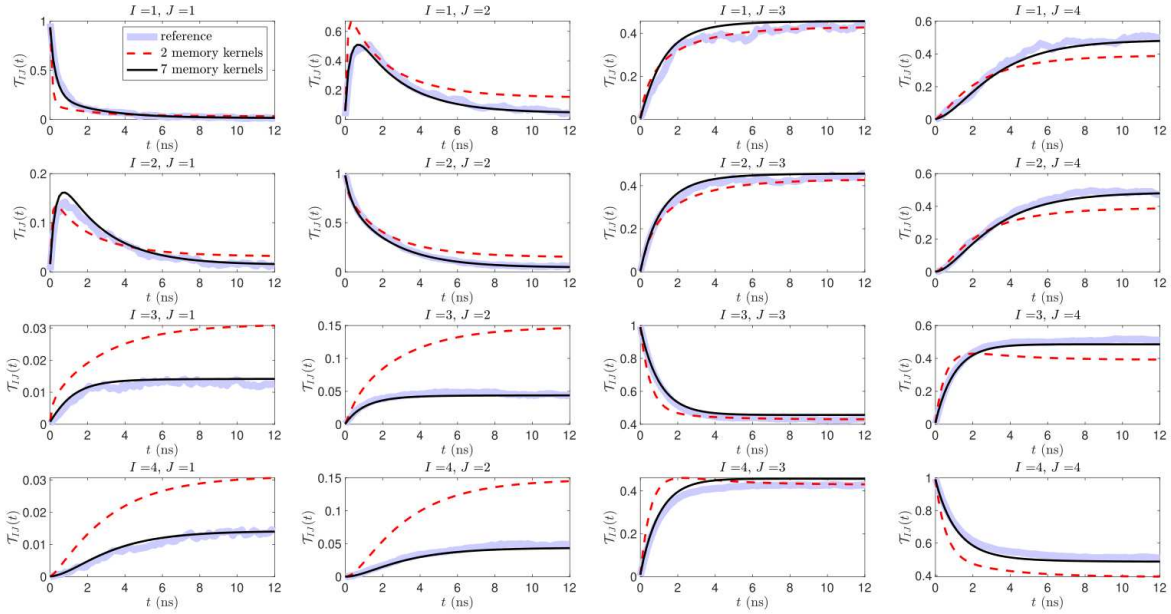


(a) $\tau = 2$ps

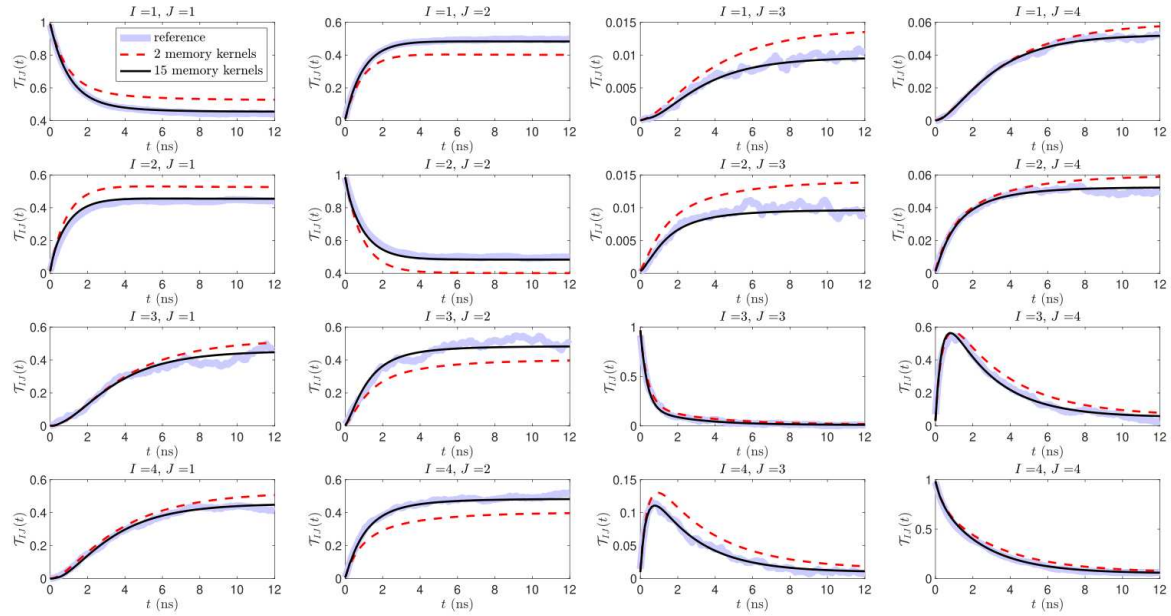


(b) $\tau = 8$ps

Figure 3.4: Verifying that $R(t)$ is approximately a MRP for sufficiently large decorrelation times, here, for PCCA states. The reference transition probabilities are computed from simple counts of $R(t)$, for $\tau = 2$ ps in (a) and $\tau = 8$ ps in (b), and compared to probabilities computed similarly using the renewal equation (3.5). The decorrelation time must be chosen long enough such that the underlying process relaxes in the macrostates before $R(t)$ jumps. While $\tau = 2$ ps in (a) does not produce a satisfactory representation of a MRP, $\tau = 8$ ps, in (b) gives good agreement without being unnecessarily large.

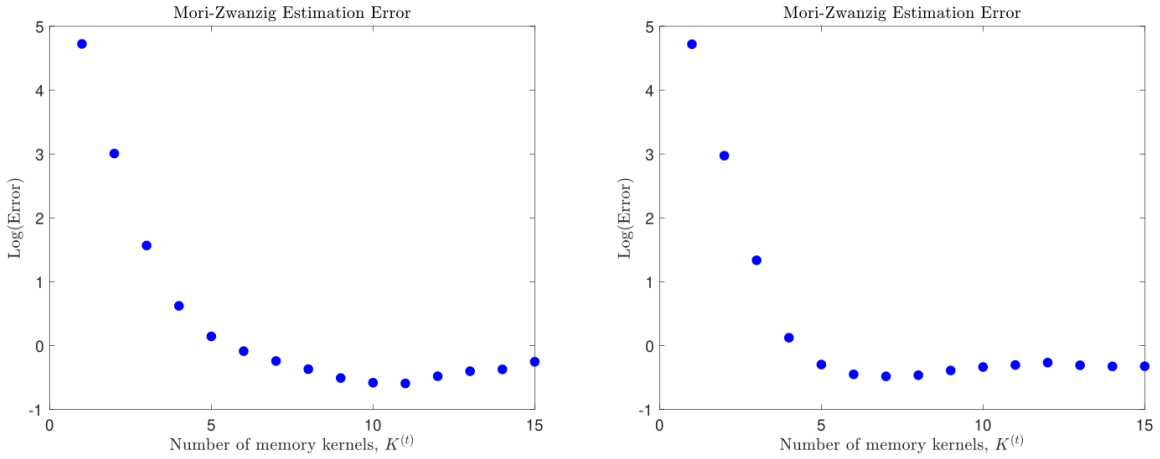


(a) PCCA states



(b) Rectangular states

Figure 3.5: Results for parametrizing the MRP using (a) states defined by PCCA and (b) equal rectangular states as described in Section 3.3. An accurate parameterization is achieved in each case with 7 and 15 memory kernels in (a) and (b) respectively. The transition probabilities shown in (a) and (b) are inferred using (3.28) with a smaller number of memory kernels (dashed line) and a larger number of memory kernels (solid line). The kernels are computed using (3.26)- (3.27) with cutoff time twice the memory length ($t_{max} = 2t_{mem}$). The macroscopic time steps, $\tau = 8$ ps (a) and $\tau = 30$ ps (b), are used after being found to define the "good" decorrelation times. (See Figure 3.4 for the choice of τ in (a)). The fit is clearly improved by increasing the number of memory kernels used in the parameterization.



(a) PCCA States, $\tau = 8$ ps

(b) Rectangular States, $\tau = 30$ ps

Figure 3.6: The error in the method described in Section 3.3 as a function of the number of memory kernels. The memory kernels are estimated at multiples of τ and the error is defined by (3.29). For each set of macrostates, a cutoff time, (a): $t_{max} = 120$ ps and (b): $t_{max} = 900$ ps, was chosen from $t_{max} = 2 \times t_{mem}$ using the largest value of t_{mem} considered (for instance in (b) $t_{mem} = 15\tau = 450$ ps, corresponding to 15 memory kernels). This is to avoid the advantage of larger t_{mem} values learning from longer trajectories.

Chapter 4

Fast Kernel Machines for Committor Estimation

In this project, we combine recent developments in machine learning and randomized numerical linear algebra and apply these techniques to the forward committor problem from computational biology, among other disciplines [11, 19, 20, 23, 36, 100–102]. The key insight brought forward by Radhakrisnan et al. [29] (equation 3), called the *neural feature ansatz* (NFA), relates the weights of neural network to the gradient outer product at that layer. Using kernel learning, the equation for the NFA simplifies significantly so a full explanation is left to the source itself and we discuss the simpler form in Section 4.2. The observation of the NFA inspired the *recursive feature machine* (RFM), an iterative method that learns an optimal transformation of the state space combined with a kernel function. For large problems, obtaining estimates of the kernel function is computationally prohibitive, necessitating newly emerging randomized linear algebra techniques. *Randomly pivoted Cholesky* [37] (RPC) is used to optimize the coefficients in the kernel function and is more efficient in both time and space. The rest of this chapter is organized as follows: kernel functions and kernel learning are introduced, the aforementioned algorithms are described in more detail, and the committor problem is discussed. Finally, we detail our algorithm combining these ideas and give preliminary results of our method applied to a toy problem and alanine dipeptide.

4.1 Kernels and Kernel learning

There are many approaches for approximating the forward committor (Section 4.2) from trajectory data including neural networks [19, 20, 36, 103], diffusion maps [23], among others. The committor describes the probability of a system starting from state $\boldsymbol{x}$ reaching some set of interest B before another set A. Interesting systems are often high dimensional and exhibit metastability, making accurate estimates resource intensive. As a result, neural networks are often used but are not immediately interpretable and methods like diffusion maps can by slow to slow on large data sets. In a sense, we seek the best of both worlds with kernels, something which is easily trained

and interpretable. Kernels do not exist without their own drawbacks. Optimizing the coefficients involves $\mathcal{O}(N^2)$ storage and $\mathcal{O}(N^3)$ operations. Hence, large datasets are computationally inaccessible for kernel methods. Also, as will be seen shortly, selecting the correct kernel function also plays a critical role in crystallizing a model. The work on the RFM and our own are still recent enough that there are no directives for how best to work with larger problems.

Broadly, kernels k,

$$k : \mathcal{X} \times \mathcal{X} \to \mathbb{R}, \tag{4.1}$$

combine feature maps with linear regression to estimate a quantity of interest or function f. The general form allows for a variety of functions to be chosen [104–107]. The generality of kernels is a powerful tool made stronger by [108–111] who noticed that symmetric, positive definite kernels define a Hilbert space. Note that, in the context of kernels, positive definite is relaxed to mean for any coefficients $c_1, \ldots, c_n \in \mathbb{R}$, any $n \in \mathbb{N}$, and any sequence of points $\boldsymbol{x}_1 \ldots, \boldsymbol{x}_n \in \mathcal{X}$,

$$\sum_{i=1}^{n} \sum_{j=1}^{m} c_i c_j k(\boldsymbol{x}_i, \boldsymbol{x}_j) \geq 0. \tag{4.2}$$

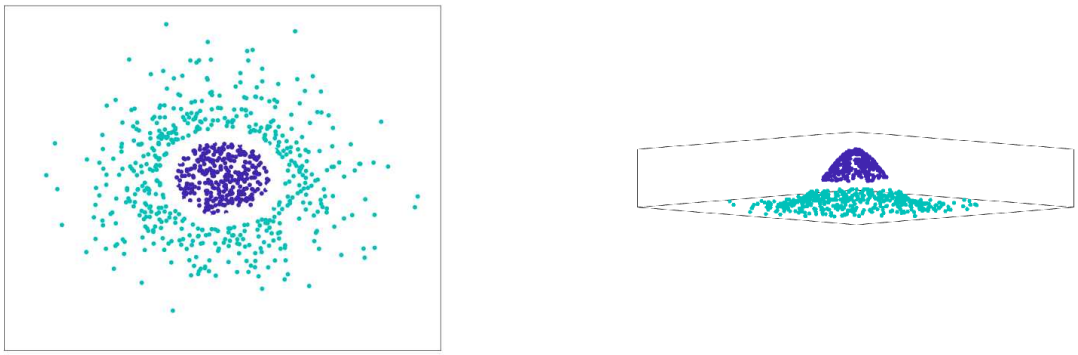
The Hilbert space, $\mathcal{H}_k$, is known as a reproducing kernel Hilbert space [35], together with an inner product $\langle \cdot, \cdot \rangle_{\mathcal{H}_k}$,

$$k(\boldsymbol{x}, \boldsymbol{y}) = \langle k(\cdot, \boldsymbol{x}), k(\cdot, \boldsymbol{y}) \rangle_{\mathcal{H}_k} \qquad \forall \boldsymbol{x}, \boldsymbol{y} \in \mathcal{X}. \tag{4.3}$$

Importantly, (4.3) shows that k may have an infinite dimensional representation, which allows for incredible flexibility.

This presents two immediate drawbacks, the higher dimensional space may not be known and the data demand may increase with the complexity of the representation of k. Discussion regarding the handling of the latter is left to 4.2 and randomly pivoted Cholesky.

As for the former concern above, consider the simple example in Figure 4.1 motivated by support vector machines [112] (SVMs). SVMs seek hyperplanes that divide or 'classify' data and are commonly used in machine learning [113–116]. A linear separation of the original data (Figure 4.1(a)) is not possible initally. However, after mapping the data to a featurized dimension



(a) Original data $\in \mathcal{X}$ **(b)** Transformed data $\in \mathcal{Z}$

Figure 4.1: An illustration of a kernel mapping. The original data in $\mathbb{R}^2$ is not linearly separable in (a) but is after mapping into $\mathbb{R}^3$.

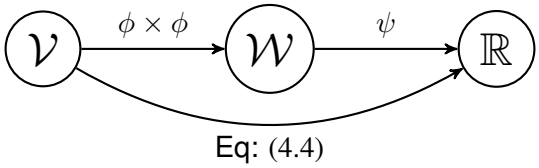


Figure 4.2: The implicit mapping performed by the kernel trick where $\mathcal{V} = \mathcal{X} \times \mathcal{X}$. In practice, the intermediary space $\mathcal{H}_k$ is never constructed or accessed.

(Figure 4.1(b)) it suddenly is possible. Such a feature map $\phi(x) : \mathcal{X} \to \mathcal{H}_k$ is not a simple matter to define and it introduces a further unknown function $\psi : \mathcal{H}_k \times \mathcal{H}_k \to \mathbb{R}$. It turns out that $\psi = \langle \cdot, \cdot \rangle_{\mathcal{H}_k}$ is the RKHS norm associated with $\mathcal{H}_k$ but the surprising result known as the kernel trick [33, 34] (Figures 4.1 and 4.2) avoids explicitly constructing such functions while retaining the benefits of kernelization. With such an inner product available in $\mathcal{H}_k$, the kernel k can be expressed as

$$k(\boldsymbol{x}, \boldsymbol{y}) = \langle \phi(\boldsymbol{x}), \phi(\boldsymbol{y}) \rangle_{\mathcal{H}_k}. \tag{4.4}$$

Estimating a function f using kernel functions is a simple expression. For input/output pairs $\boldsymbol{x}_n, \boldsymbol{y}_n = f(x_n), n = 1, \ldots, N$ f is just a superposition of the kernel functions

$$f(\boldsymbol{x}) \approx \sum_{n=1}^{N} \boldsymbol{\theta}_n k(\boldsymbol{x}_n, \boldsymbol{x}). \tag{4.5}$$

The weights, $\boldsymbol{\theta}_n$, are obtained by kernel ridge regression on

$$(\boldsymbol{K} - \gamma N \boldsymbol{I})\boldsymbol{\theta} = \boldsymbol{y}, \tag{4.6}$$

where $\boldsymbol{K} \in \mathbb{R}^{N \times N}$ and the entry $\boldsymbol{K}_{nm} = k(\boldsymbol{x}_n, \boldsymbol{x}_m)$.

Designing the kernel $\boldsymbol{K}$ is clearly important to the performance of the interpolator. Common choices include the radial basis function (RBF) kernels or Gaussian kernels [33, 117–121] which have the form

$$k(\boldsymbol{x}, \boldsymbol{x}') = \exp\left[-\gamma \|\boldsymbol{x} - \boldsymbol{x}'\|^2\right], \tag{4.7}$$

where γ is a bandwidth representing the width of the Gaussians at each data point. For this kernel, and similarly for other kernels, the approximation of f is a superposition of Gaussians at each point $\boldsymbol{x}_n$ with height $\boldsymbol{\theta}_n$. The key point in (4.7) is that the output depends only on the Euclidean distance between points. Other kernels, such as Laplace kernels [122], similarly depend solely on Euclidean distance. However, this metric is not well suited to high dimensions as the average distance approaches a constant value [123]. When the variation of f occurs on a low dimensional manifold, these kernels struggle to find a good featurization of the data. Following the work of Radhakrishnan et al. [29], this observation inspired an iterative featurization scheme to optimally extract and scale these low dimensional variations.

4.2 Background

In [29], Radhakrishnan et al. introduce the *recursive feature machine* (RFM) which leverages their NFA to define a scaling matrix that mimics the feature learning of trained neural networks. The simplified equation for the NFA with one hidden layer and is

$$W_1^{\mathbf{T}} W_1 \propto \frac{1}{N} \sum_{n=1}^{N} \nabla \hat{f}\left(\boldsymbol{x}_n\right) \nabla \hat{f}\left(\boldsymbol{x}_n\right)^{\mathbf{T}}, \tag{4.8}$$

where W_1 represents the weights of the first hidden layer, $\{\boldsymbol{x}_n\}_{p=1}^n$ are the inputs of the dataset, and $\hat{f}$ is the neural network prediction for the quantity of interest f. The average gradient outer product (AGOP) of (4.8) attempts to represent the optimal rescaling of the dataset and can therefore be used to define new coordinates for the kernel machine. Define the scaling matrix $\boldsymbol{M}$ by

$$\boldsymbol{M} = \frac{1}{N}\sum_{p=1}^{N} \nabla \hat{f}\left(\boldsymbol{x}_n\right) \nabla \hat{f}\left(\boldsymbol{x}_n\right)^{\mathbf{T}}, \tag{4.9}$$

where $\hat{f}$ is now the kernel estimation of f. The RFM algorithm is summarized in Algorithm 1 and our algorithm (Algorithm 3) follows a very similar form. The new metric uses the feature matrix $\boldsymbol{M}$ together with a kernel reminiscent of the Mahalanobis kernel [124, 125]

$$\langle \boldsymbol{x}, \boldsymbol{y} \rangle = (\boldsymbol{x} - \boldsymbol{y})^{\mathbf{T}} \boldsymbol{M} (\boldsymbol{x} - \boldsymbol{y}), \tag{4.10}$$

where $\boldsymbol{M}$ may be thought of as the squared Mahalanobis matrix [126].

Algorithm 1 Recursive Feature Machine [29]

Input: data $\{\boldsymbol{x}_n, y_n\}$, kernel k, bandwidth h, number iterations T
Output: kernel weights α, feature matrix $\boldsymbol{M}$
Initialize: $\boldsymbol{M} = I$
for $t \in T$ **do**
  $k(\boldsymbol{x}, \boldsymbol{x}') = \exp\left[-(\boldsymbol{x} - \boldsymbol{x}')^{\mathbf{T}} \boldsymbol{M} (\boldsymbol{x} - \boldsymbol{x}')/h\right]$
    Define kernel function
  $f(\boldsymbol{x}) \leftarrow$ `KernelRidgeRegression`$(\{\boldsymbol{x}_n, \boldsymbol{y}_n\}, k)$
    Update the coefficients
  $\boldsymbol{M} = \frac{1}{n}\sum_{p=1}^{n} \nabla \hat{f}\left(\boldsymbol{x}_p\right) \nabla \hat{f}\left(\boldsymbol{x}_p\right)^{\mathbf{T}}$
    Update the scaling matrix
end for

This adaptive distance function and RFM differ from other approaches to modeling the committor in that they involve both the knowledge of parameters of the problem and iteratively defines the mapping. For example, diffusion maps [127] and the variational approach to conformational dynamics [24] have both been used to estimate the committor [23, 128, 129] but do not directly

address the committor problem itself. Other methods [124, 125], remap the input data through a similar linear transformation but do not perform any iteration. The RFM adapts itself to the particular problem at hand and updates the transformation using the AGOP. In their experiments, Radhakrishnan et al. found the RFM suppressed irrelevant directions in high dimension feature learning tasks and the scaling matrix $\boldsymbol{M}$ experimentally confirmed the NFA.

The performance of the RFM, in part, depends on obtaining estimates of f quickly through kernel ridge regression. However, obtaining the coefficients for such an approximation can be computational prohibitive [130]. As previously mentioned, elementary regression requires $\mathcal{O}(N^3)$ operations and $\mathcal{O}(N^2)$ storage. Consequently, obtaining kernel estimates is too expensive for datasets on the order of 10^5 or greater and historically other methods, such as neural networks, have been used in this data regime. However, recent advances in numerical linear algebra [131,132] may improve the applicability of kernel methods; in particular, we employ randomly pivoted Cholesky (RPC) [37] on data sets up to $N = 10^6$. We found (Section 4.5) that the kernel method described herein (Algorithm 3, Section 4.3) trains more quickly than simple feed-forward neural networks and achieves comparable results.

In the context of minimizing the error in kernel ridge regression, RPC identifies a set of "landmark" indicies $S = \{s_1, \ldots, s_r\}$ that will be used to sample columns of the kernel matrix $\boldsymbol{K}$. Here, r denotes the rank of the approximation and the resource demand is reduced to $\mathcal{O}(Nr^2)$ operations and storage requirements are therefore $\mathcal{O}(Nr)$. Experimentally, we found that $r \leq 1000$ produced good agreement with the reference committors for datasets up to size $N = 10^6$. In these experiments, a block version of RPC was implemented and is described in Algorithm 2.

Finally, we briefly introduce the committor problem for sets A and B of the phase space that denote physically relevant regions or, generally, areas of interest. One prototypical example defines A and B as the folded and unfolded states of a protein [133–135]. The (forward) committor then describes the probability that a system will reach some absorbing state B before A. Mathe-

Algorithm 2 RPCholesky [37]

Input: kernel function k (to get entries of $\boldsymbol{K}$, approximation rank r (multiple of 10)
Output: landmark index set S
Initialize: $\boldsymbol{F} \leftarrow 0$, $S \leftarrow$, $T \leftarrow r/10$, $\boldsymbol{d} \leftarrow \mathrm{diag}(\boldsymbol{K})$
for $i = 0 : 9$ **do**
  $s_{iT+1}, \ldots, s_{iT+T} \sim \boldsymbol{d} / \sum_{j=1} d_j$
    Randomly sample indices
  $S' \leftarrow \texttt{Unique}(\{s_{iT+1}, \ldots, s_{iT+T}\})$
    Identify unique indices
  $S \leftarrow S \cup S'$
    Add new indices to landmark set
  $\boldsymbol{G} \leftarrow \boldsymbol{K}(:, S') - \boldsymbol{F}\boldsymbol{F}(S', :)^*$
    Evaluate new columns
  $\boldsymbol{G}(S', :) \leftarrow \boldsymbol{G}(S', :) + \varepsilon_{\mathrm{mach}} \mathrm{tr}(\boldsymbol{G}(S', :))\boldsymbol{I}$
    Regularize
  $\boldsymbol{R} \leftarrow \texttt{Cholesky}(\boldsymbol{G}(S', :))$
    Upper triangular Cholesky factor
  $\boldsymbol{F}(:, iT + 1 : iT + |S'|) \leftarrow \boldsymbol{G}\boldsymbol{R}^{-1}$
    Update approximation
  $\boldsymbol{d} \leftarrow \boldsymbol{d} - \texttt{SquaredRowNorms}(\boldsymbol{G}\boldsymbol{R}^{-1})$
    Update sampling probabilities
  $\boldsymbol{d} \leftarrow \max\{\boldsymbol{d}, 0\}$
    Ensure nonnegative probabilities
  $\boldsymbol{d}(S') \leftarrow 0$
    Prevent double sampling
end for

matically, the committor is often written

$$q(\boldsymbol{x}) = \mathbb{P}_{\boldsymbol{x}}(T_B < T_A) \tag{4.11}$$

where T_S is the first hitting time of the set S. The committor satisfies a variational principle involving a Dirichlet energy, which has been used in machine learning in its continuous form [19,20] and discrete form [128]. Our algorithm is motivated by a variational principle which states the committor q is the unique minimizer of (4.12) with boundary conditions $q|_A = 0$, $q|_B = 1$. The expectation is taken as the average over trajectories $\boldsymbol{x}(t)$ initialized in the equilibrium distribution $\boldsymbol{x}(0) \sim \mu$ and terminated at time τ.

$$\mathcal{L}(q) = \mathbb{E}_\mu \left| q(\boldsymbol{x}(0)) - q(\boldsymbol{x}(\tau)) \right|^2 . \tag{4.12}$$

We discuss our kernel ansatz for the committor that motivated in part by (4.12) in more detail in the following section.

4.3 Fast Committor Machine

We seek a data-driven approximation of the committor given in Equation (4.12). Let ρ denote the sampling density, input data $\boldsymbol{x}_n(0) \sim \rho$, and output data from τ time units of evolution $\boldsymbol{y}_n = \boldsymbol{x}_n(\tau)$. Using the input/output pairs $\{\boldsymbol{x}_n, \boldsymbol{y}_n\}_{n=1}^N$, $\mathcal{L}(q)$ can be approximated by

$$L(q) = \frac{1}{N} \sum_{n=1}^{N} w_n \left| q(\boldsymbol{x}_n) - q(\boldsymbol{y}_n) \right|^2 , \tag{4.13}$$

where the weights w_n are the likelihood ratios between the sampling ρ and target density μ

$$w_n = \frac{\mu(\boldsymbol{x}_n)}{\rho(\boldsymbol{x}_n)}. \tag{4.14}$$

In practice, only the relative weights may be known but this is still sufficient to minimize (4.13). The sampling and target densities may differ as a result of sampling the molecular dynamics (MD) at a higher temperature. Relevant to our work, this is how samples for our experiments in Section 4.5 were obtained. The MD simulation was run at a higher temperature to sample the phase space more effectively while the input/output pairs were generated at the temperature of interest; more detail is given in Section 4.5.

One quick note regarding the weights in (4.13), if the input/output pairs $(\boldsymbol{x}_n, \boldsymbol{y}_n)$ are ergodic and μ is absolutely continuous with respect to ρ, that is

$$\rho > 0 \quad \text{whenever} \quad \mu > 0,$$

then the estimator is unbiased and

$$\mathbb{E}\left[L(q)\right] = \mathcal{L}(q).$$

From the law of large numbers, as $N \to \infty$ the estimation converges almost surely $L(q) \to \mathcal{L}(q)$.

To estimate the committor, first we define an extension of the Laplace kernel function $k_{\boldsymbol{M}}$ by

$$k_M(\boldsymbol{x}, \boldsymbol{x}') = \begin{cases} \exp\left(-\frac{1}{\epsilon}\|\boldsymbol{M}^{1/2}(\boldsymbol{x} - \boldsymbol{x}')\|\right) & \boldsymbol{x}, \boldsymbol{x}' \in (A \cup B)^{\mathbf{C}} \\ 0 & \text{otherwise} \end{cases}, \tag{4.15}$$

where $\boldsymbol{M}$ is a square positive definite scaling matrix and $\epsilon > 0$ is a bandwidth parameter. To minimize (4.13), let $\boldsymbol{\theta} \in \mathbb{R}^N$ and $\boldsymbol{x} \in (A \cup B)^{\mathbf{C}}$. We then introduce the new committor approximation

$$q_{\boldsymbol{\theta}}(\boldsymbol{x}) = \sum_{n=1}^{N} \boldsymbol{\theta}_n \left[k_{\boldsymbol{M}}(\boldsymbol{x}_n, \boldsymbol{x}) - k_{\boldsymbol{M}}(\boldsymbol{y}_n, x)\right], \tag{4.16}$$

where $q_{\boldsymbol{\theta}}|_A = 0$ and $q_{\boldsymbol{\theta}}|_B = 1$. This form of the committor is motivated by the desire to obtain a symmetric positive definite system and a more rigorous mathematical justification may be found in the future, though we can offer partial motivation here. The general form can be described by a linear combination of kernel functions $k_{\boldsymbol{M}}$ at the data points $\boldsymbol{x}_n, \boldsymbol{y}_n$ as

$$q_{\boldsymbol{c},\boldsymbol{d}}(\boldsymbol{x}) = \sum_{n=1}^{N} c_n k_{\boldsymbol{M}}(\boldsymbol{x}_n, \boldsymbol{x}) + \sum_{n=1}^{N} d_n k_{\boldsymbol{M}}(\boldsymbol{y}_n, \boldsymbol{x}), \tag{4.17}$$

where $\boldsymbol{c}, \boldsymbol{d} \in \mathbb{R}^N$ are coefficients to be optimized. Theorem 4.3.1 shows that the optimal choices of $\boldsymbol{c}$ and $\boldsymbol{d}$ yields 4.15. Note, in Section 4.4, the assumption of positive semi-definiteness is justified.

Theorem 4.3.1. *Define the least-squares loss function*

$$L_\gamma(\boldsymbol{c}, \boldsymbol{d}) = \frac{1}{N}\left\| \begin{bmatrix} \mathbf{I} \\ -\mathbf{I} \end{bmatrix}^T \boldsymbol{K} \begin{bmatrix} \boldsymbol{c} \\ \boldsymbol{d} \end{bmatrix} - \boldsymbol{b} \right\|^2 + \gamma \begin{bmatrix} \boldsymbol{c} \\ \boldsymbol{d} \end{bmatrix}^T \boldsymbol{K} \begin{bmatrix} \boldsymbol{c} \\ \boldsymbol{d} \end{bmatrix},$$

where the positive semidefinite kernel matrix

$$\boldsymbol{K} = \begin{bmatrix} \boldsymbol{K}^{11} & \boldsymbol{K}^{12} \\ \boldsymbol{K}^{21} & \boldsymbol{K}^{22} \end{bmatrix},$$

consists of four blocks with entries

$$\begin{aligned} k_{mn}^{11} &= \sqrt{w_m}\sqrt{w_n}\, k_{\boldsymbol{M}}(\boldsymbol{x}_m, \boldsymbol{x}_n), \\ k_{mn}^{12} &= \sqrt{w_m}\sqrt{w_n}\, k_{\boldsymbol{M}}(\boldsymbol{x}_m, \boldsymbol{y}_n), \\ k_{mn}^{21} &= \sqrt{w_m}\sqrt{w_n}\, k_{\boldsymbol{M}}(\boldsymbol{y}_m, \boldsymbol{x}_n), \\ k_{mn}^{22} &= \sqrt{w_m}\sqrt{w_n}\, k_{\boldsymbol{M}}(\boldsymbol{y}_m, \boldsymbol{y}_n). \end{aligned}$$

Then, the loss function $\boldsymbol{L}_\gamma(\boldsymbol{c}, \boldsymbol{d})$ has a minimizer that satisfies $\boldsymbol{c} + \boldsymbol{d} = \boldsymbol{0}$.

Proof. The first term of the loss function is clearly convex and the second is as well due to $\gamma > 0$ and $\boldsymbol{K}$ being PSD. Therefore, a minimizer can be found by setting the gradient equal to 0,

$$\nabla_{\boldsymbol{c},\boldsymbol{d}} L_\gamma(\boldsymbol{c}, \boldsymbol{d}) = \frac{2}{N}\, \boldsymbol{K} \begin{bmatrix} \mathbf{I} \\ -\mathbf{I} \end{bmatrix} \left[\begin{bmatrix} \mathbf{I} \\ -\mathbf{I} \end{bmatrix}^T \boldsymbol{K} \begin{bmatrix} \boldsymbol{c} \\ \boldsymbol{d} \end{bmatrix} - \boldsymbol{b} \right] + 2\gamma\, \boldsymbol{K} \begin{bmatrix} \boldsymbol{c} \\ \boldsymbol{d} \end{bmatrix} = \boldsymbol{0}.$$

From rearranging terms and considering the positive definite system

$$\left[\begin{bmatrix} \mathbf{I} \\ -\mathbf{I} \end{bmatrix} \begin{bmatrix} \mathbf{I} \\ -\mathbf{I} \end{bmatrix}^T \boldsymbol{K} + \gamma N \begin{bmatrix} \boldsymbol{I} & \\ & \boldsymbol{I} \end{bmatrix} \right] \begin{bmatrix} \boldsymbol{c} \\ \boldsymbol{d} \end{bmatrix} = \begin{bmatrix} \mathbf{I} \\ -\mathbf{I} \end{bmatrix} \boldsymbol{b},$$

we can multiply by $\begin{bmatrix} \boldsymbol{I} & \boldsymbol{I} \end{bmatrix}$ on the left to obtain

$$\boldsymbol{0} + \gamma N \begin{bmatrix} \boldsymbol{I} & \\ & \boldsymbol{I} \end{bmatrix} \begin{bmatrix} \boldsymbol{c} \\ \boldsymbol{d} \end{bmatrix} = \boldsymbol{0}.$$

Algorithm 3 Fast Committor Machine

Input: Training points $\{\boldsymbol{x}_n, \boldsymbol{y}_n, w_n\}_{n=1}^N$, solution vector $\boldsymbol{b}$, kernel function k
Parameters: bandwidth ϵ, regularization γ, approximation rank r (multiple of 10)
Output: Scaling matrix $\boldsymbol{M}$, committor estimate $q_{\boldsymbol{\theta}}$
Initialize: $\boldsymbol{M} \leftarrow \boldsymbol{I}$
for $t = 1 : 5$ **do**
$\boldsymbol{M} \leftarrow \boldsymbol{M} / \operatorname{tr} \operatorname{cov}(\{\boldsymbol{M}^{1/2}\boldsymbol{x}_1, \ldots, \boldsymbol{M}^{1/2}\boldsymbol{x}_N\})$
Rescale $\boldsymbol{M}$
$k_{mn} \leftarrow k_{\boldsymbol{M}}(\boldsymbol{x}_m, \boldsymbol{x}_n) - k_{\boldsymbol{M}}(\boldsymbol{x}_m, \boldsymbol{y}_n) - k_{\boldsymbol{M}}(\boldsymbol{y}_m, \boldsymbol{x}_n) + k_{\boldsymbol{M}}(\boldsymbol{y}_m, \boldsymbol{x}_n)$
Repopulate $\boldsymbol{K}$ with kernel $k_{\boldsymbol{M}}$
$k_{mn} \leftarrow \sqrt{w_m, w_n} k_{mn}$;
Reweight $\boldsymbol{K}$
$S = \{s_1, \ldots, s_N\} \leftarrow \texttt{RPC}(\boldsymbol{K}, r)$
Apply RPC
$\boldsymbol{K}(S, S) \leftarrow \boldsymbol{K}(S, S) + \varepsilon_{\texttt{mach}} \text{tr}(\boldsymbol{K}(S, S))\boldsymbol{I}$
Regularize
$\boldsymbol{\eta} \leftarrow$ Solution to $\left[\boldsymbol{K}(:, S)^{\mathbf{T}} \boldsymbol{K}(:, S) + \gamma N \boldsymbol{K}(S, S)\right] \boldsymbol{\eta} = \boldsymbol{K}(:, S)^{\mathbf{T}} \boldsymbol{b}$
Optimize coefficients
$q_{\boldsymbol{\theta}}(\boldsymbol{x}) \leftarrow \sum_{i=1}^r \eta_i \sqrt{w_{s_i}} \left[k_{\boldsymbol{M}}(\boldsymbol{x}_{s_i}, \boldsymbol{x}) - k_{\boldsymbol{M}}(\boldsymbol{y}_{s_i}, \boldsymbol{x})\right]$
Update $q_{\boldsymbol{\theta}}$
$\boldsymbol{M} \leftarrow \sum_{n=1}^N \nabla q_{\boldsymbol{\theta}}(\boldsymbol{x}_n) \nabla q_{\boldsymbol{\theta}}(\boldsymbol{x}_n)^{\mathbf{T}}$
Update $\boldsymbol{M}$
end for

The minimizer of this system is found by taking $c_n = -d_n$ for $n = 1, \ldots, N$. □

The scaling matrix $\boldsymbol{M}$ is similar to the one used in the RFM outlined in Algorithm 1. In practice, we found that choosing a bandwidth of $\epsilon = 1$ was intuitively reasonable and performed well in numerical tests. In the transformed space $\boldsymbol{M}^{1/2}\boldsymbol{x}_1, \ldots, \boldsymbol{M}^{1/2}\boldsymbol{x}_n$, $\epsilon = 1$ corresponds to one standard deviation unit. Furthermore, ϵ is updated through rescaling $\boldsymbol{M}$ through each loop of our algorithm, the Fast Committor Machine (FCM) described in Algorithm 3. The bandwidth updates are a consequence of normalizing the scaling matrix $\boldsymbol{M}$ by the covariance matrix.

By rescaling the bandwidth, we are taking advantage of $\boldsymbol{M}$ rescaling the input data. In particular, the change-of-basis $\boldsymbol{z} = \boldsymbol{M}^{1/2}\boldsymbol{x}$ is advantageous in two regards. First, the gradients $\nabla_{\boldsymbol{z}} f_{\boldsymbol{\theta}}(\boldsymbol{z}_1), \ldots, \nabla_{\boldsymbol{z}} f_{\boldsymbol{\theta}}(\boldsymbol{z}_N)$ are isotropic, see Proposition 4.3.2. $\boldsymbol{M}$ may be understood in a couple ways, as an uncentered covariance matrix of the gradients $\nabla f_{\boldsymbol{\theta}}(\boldsymbol{x})$, summarizing the variance and

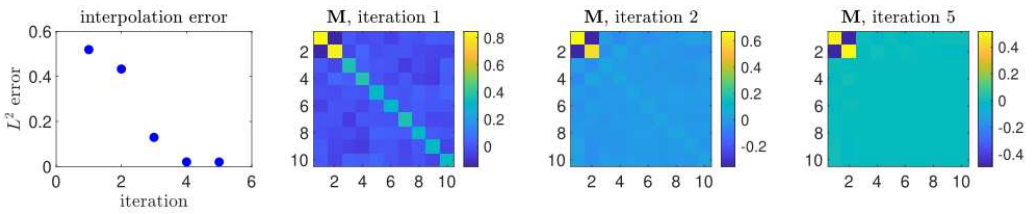


Figure 4.3: Algorithm 1 applied to the Gaussian function $f(\boldsymbol{x}) = \exp(-\boldsymbol{x}^{\mathbf{T}}\boldsymbol{A}\boldsymbol{x})$ with $\boldsymbol{A}_{ij} = 0$ for $i, j > 2$. $f(\boldsymbol{x})$ therefore only relies on two dimensions which the scaling matrix $\boldsymbol{M}$ identifies. From left to right: The error $\|\hat{f} - f\|_2$, and the Mahalanobis matrix $\boldsymbol{M}$ after iterations 1, 2 and 5.

correlations between dimensions, or as a whitening transformation [136] of the gradients. Whitening is a process that rescales, in this case, the gradients to have unit variance and no correlation. Therefore, $\boldsymbol{M}$ scales data along $\nabla f_{\boldsymbol{\theta}}(\boldsymbol{x})$ relative to the magnitude of the gradient; of particular note is that this would collapse irrelevant directions. The result is, as noted previously, isotropic gradients and a lower dimension representation of the data set which has clear benefits for learning an interpolator.

Second, as in the RFM, the linear transformation $\boldsymbol{M}$ is remapping the input data to the lower dimensional manifold where the variance of $f_{\boldsymbol{\theta}}$ occurs. This is a result of the data being stretched in directions of high variance and shrunk in directions of low variance. Through iteration, as $\boldsymbol{M}$ identifies the lower-dimensional space the samples are mapped closer together which, in turn, leads to a smaller bandwidth. The smaller bandwidth decreases the width of the kernel function helping to differentiate the samples. Figure 4.3 illustrates Algorithm 3 identifying the relevant underlying subspace.

Finally, consider $f : \mathbb{R}^d \to \mathbb{R}$ where $f = (g_{\boldsymbol{A}} \circ \boldsymbol{A})(\boldsymbol{x})$, $\boldsymbol{A} \in \mathbb{R}^{d \times d}$, and $\boldsymbol{x} \in \mathbb{R}^d$. The optimal transformation $\boldsymbol{A}$ is one such that the gradients

$$\nabla g_{\boldsymbol{A}}(\boldsymbol{z}_1), \ldots, \nabla g_{\boldsymbol{A}}(\boldsymbol{z}_1),$$

are isotropic, where

$$\boldsymbol{z}_i = \boldsymbol{A}\boldsymbol{x}_i, \qquad i = 1, \ldots, N.$$

The following proposition makes this idea rigorous.

Proposition 4.3.2. *The following are equivalent:*

(i) $\boldsymbol{A}$ *is invertible, and the sample gradients* $\nabla g_{\boldsymbol{A}}(\boldsymbol{z}_1), \dots, \nabla g_{\boldsymbol{A}}(\boldsymbol{z}_N)$ *are isotropic, that is,*

$$\frac{1}{N}\sum_{i=1}^{N} |\boldsymbol{u}^T \nabla g_{\boldsymbol{A}}(\boldsymbol{z}_i)|^2 = 1$$

for any unit vector $\boldsymbol{u} \in \mathbb{R}^d$.

(ii) The average gradient product

$$\boldsymbol{M} = \frac{1}{N}\sum_{i=1}^{N} \nabla f(\boldsymbol{x}_i) \nabla f(\boldsymbol{x}_i)^T.$$

is invertible, and $\boldsymbol{A} = \boldsymbol{Q}\boldsymbol{M}^{1/2}$ *for an orthogonal matrix* $\boldsymbol{Q} \in \mathbb{R}^{d\times d}$.

(iii) $\boldsymbol{M}$ *is invertible, and* $\boldsymbol{A}$ *transforms distances according to*

$$\|\boldsymbol{A}(\boldsymbol{x} - \boldsymbol{x}')\| = \|\boldsymbol{M}^{1/2}(\boldsymbol{x} - \boldsymbol{x}')\|,$$

for each $\boldsymbol{x}, \boldsymbol{x}' \in \mathbb{R}^d$.

Proof. Either of the conditions (i)-(ii) implies the linear map $\boldsymbol{A}$ is invertible. Therefore calculate

$$\begin{aligned}\frac{1}{N}\sum_{i=1}^{N} |\boldsymbol{u}^T \nabla g_{\boldsymbol{A}}(\boldsymbol{z}_i)|^2 &= \frac{1}{N}\sum_{i=1}^{N} |\boldsymbol{u}^T \boldsymbol{A}^{-T} \nabla f(\boldsymbol{x}_i)|^2 \\ &= \boldsymbol{u}^T \boldsymbol{A}^{-T} \boldsymbol{M} \boldsymbol{A}^{-1} \boldsymbol{u}.\end{aligned}$$

The above display is 1 for each unit vector $\boldsymbol{u} \in \mathbb{R}^d$ if and only if $\boldsymbol{A}^{-T}\boldsymbol{M}\boldsymbol{A}^{-1} = \mathbf{I}$ and $\boldsymbol{M}^{1/2}\boldsymbol{A}^{-1}$ is an orthogonal matrix. Thus, (i) and (ii) are equivalent.

Clearly, (ii) implies (iii). Conversely, if $\|\boldsymbol{A}\boldsymbol{x}\| = \|\boldsymbol{M}^{1/2}\boldsymbol{x}\|$ for each $\boldsymbol{x} \in \mathbb{R}^d$, it follows that

$$\boldsymbol{M} = \boldsymbol{A}^T \boldsymbol{A}.$$

Consequently, if $\boldsymbol{A} = \boldsymbol{U}\boldsymbol{\Sigma}\boldsymbol{V}^T$ is a singular value decomposition for $\boldsymbol{A}$, then $\boldsymbol{M} = \boldsymbol{V}\boldsymbol{\Sigma}^2\boldsymbol{V}^T$ is an eigenvalue decomposition for $\boldsymbol{M}$, and

$$\boldsymbol{A} = \boldsymbol{Q}\boldsymbol{M}^{1/2}, \quad \text{where } \boldsymbol{Q} = \boldsymbol{U}\boldsymbol{V}^T \text{ is orthogonal.} \tag{4.18}$$

This shows that (ii) and (iii) are equivalent. □

4.4 Methods

4.4.1 Least Squares Solution

In the FCM algorithm, RPC is used to minimize the loss associated with a least squares problem. This section will give details regarding the formation of the least squares problem and describe the testing methodology used to validate our algorithm's performance.

Recall, the data is given as input/output $(\boldsymbol{x}_n, \boldsymbol{y}_n)$ pairs with weights w_n and we seek a kernel $k_{\boldsymbol{M}}$ minimizing the loss $L(q)$ (Equation (4.13)). Define the kernel matrix $\boldsymbol{K} \in \mathbb{R}^{N\times N}$ and solution vector $\boldsymbol{b} \in \mathbb{R}^N$ as

$$\begin{aligned} k_{mn} &= \sqrt{w_m}\sqrt{w_n}\big[k_{\boldsymbol{M}}(\boldsymbol{x}_m, \boldsymbol{x}_n) - k_{\boldsymbol{M}}(\boldsymbol{x}_m, \boldsymbol{y}_n) \\ &\qquad - k_{\boldsymbol{M}}(\boldsymbol{y}_m, \boldsymbol{x}_n) + k_{\boldsymbol{M}}(\boldsymbol{y}_m, \boldsymbol{y}_n)\big], \\ b_n &= \sqrt{w_n}[1_B(\boldsymbol{y}_n) - 1_B(\boldsymbol{x}_n)]. \end{aligned}$$

As this is a weighted least squares problem, the coefficients are normalized according to

$$\overline{\boldsymbol{\theta}}_n = \boldsymbol{\theta}_n/\sqrt{w_n}, \qquad n = 1, \ldots, N.$$

Denote the reproducing kernel Hilbert space (RKHS) associated with the kernel approximation of $q_{\boldsymbol{\theta}}$ by $\mathcal{H}$. The form of $\boldsymbol{K}$ is then given by

$$\boldsymbol{K} = \langle \sqrt{w_m}[k_{\boldsymbol{M}}(x, x_m) - k_{\boldsymbol{M}}(x, y_m)], \sqrt{w_n}[k_{\boldsymbol{M}}(x, x_n) - k_{\boldsymbol{M}}(x, y_n)] \rangle_{\mathcal{H}}\,. \tag{4.19}$$

By this observation, $\boldsymbol{K}$ is a Gram matrix and therefore is symmetric positive semi-definite which, in the case of kernels, is sufficient to guarantee the existence of a solution to the least squares problem. However, the cubic dependence on N in simple least squares solutions requires the use of more sophisticated means to identify $\overline{\boldsymbol{\theta}} \in \mathbb{R}$ minimizing

$$L_\gamma(\overline{\boldsymbol{\theta}}) = \tfrac{1}{N}\|\boldsymbol{K}\overline{\boldsymbol{\theta}} - \boldsymbol{b}\|^2 + \gamma\, \overline{\boldsymbol{\theta}}^T \boldsymbol{K}\overline{\boldsymbol{\theta}}. \tag{4.20}$$

From RPC (Algorithm 2), the problem is restricted to a set of indicies S by

$$\boldsymbol{\theta}(i) = \begin{cases} \boldsymbol{\eta}_i & i \in S \\ 0 & \text{otherwise} \end{cases}, \tag{4.21}$$

where $\boldsymbol{\eta} \in \mathbb{R}^r$ for $r = |S|$, the rank of the approximation. In practice, for problem sizes up to $N = 10^6$, we found choosing $r = 1000$ was sufficient as the error in the FCM had mostly converged by this point, see Figure 4.4.a. The computational gains result from only needing to construct and store the columns identified by S leading to the linear system

$$\min_{\boldsymbol{\eta}\in\mathbb{R}^r} \tfrac{1}{N}\|\boldsymbol{K}(:,S)\boldsymbol{\eta} - \boldsymbol{b}\|^2 + \gamma\, \boldsymbol{\eta}^T \boldsymbol{K}(S,S)\boldsymbol{\eta},$$

where the coefficients η are found by minimizing

$$\left[\boldsymbol{K}(:,S)^T\boldsymbol{K}(:,S) + \gamma N \boldsymbol{K}(S,S)\right]\boldsymbol{\eta} = \boldsymbol{K}(:,S)^T\boldsymbol{b}.$$

The final hyperparameter to tune is the regularization γ which helps prevent overfitting by shrinking the norm of the coefficients. Notice that in Equation (4.20), the term $\left(\overline{\boldsymbol{\theta}}^{\mathbf{T}}\boldsymbol{K}\overline{\boldsymbol{\theta}}\right)^{1/2}$ is the RKHS norm of $\mathcal{H}$. For our experiments, $\gamma = 10^{-6}$ was chosen through a grid search, which could also be used to identify choices for the bandwidth ϵ and approximation rank r. There is no immediate intuition behind the choice of γ in the vein of our earlier discussion on the choice of ϵ.

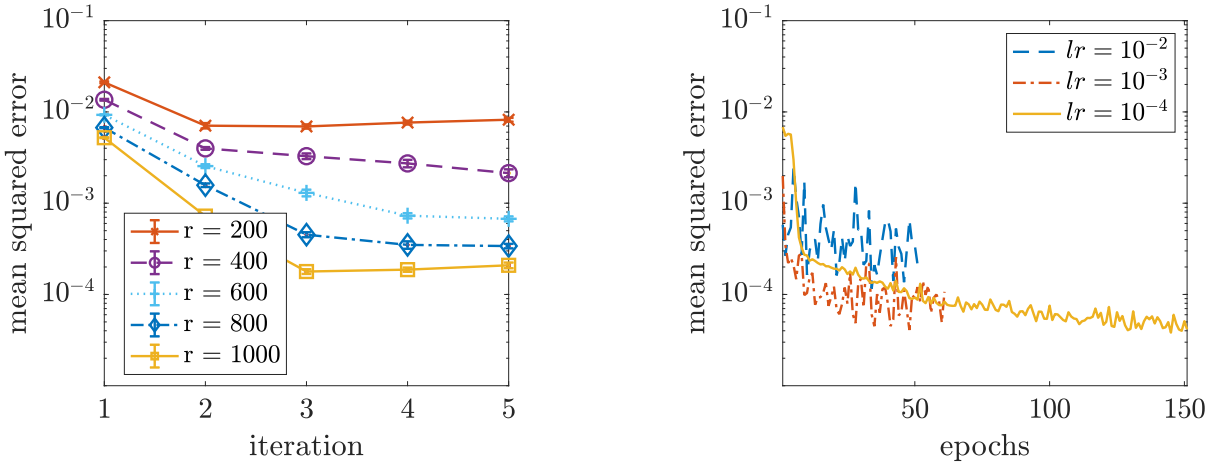


Figure 4.4: Training performance for the triple-well experiment (Section 4.5.1), demonstrating the advantages of training the FCM. Improving the model is intuitive and saturates for parameters such as r where neural networks famously require significant tuning of several parameters with unexpected interactions. The error is computed using the reference committor from Fig. 4.5. (a) The FCM with different approximation ranks r. (b) The NN with different learning rates lr.

As for the approximation rank, increasing r increases the number of columns selected from K. This has the benefit of improving the accuracy of the approximation at the cost of increased storage and run time. We chose r through a grid search but in practice could be chosen as large as computational resources allow. In theory, the minimum useful value of r depends on the spectral decay of $\boldsymbol{K}$ [37].

4.4.2 Neural Network

To demonstrate the performance of the FCM, a feedforward neural network (NN) was trained on the same dataset to estimate the committor function. The size of the neural network was chosen such that the number of parameters being optimized by the (largest) FCM and the NN were approximately the same (see Table 4.1). The number of parameters in the FCM is dictated by the dimension of the phase space d and the approximation rank r leading to $r + d(d+1)/2$ parameters. This results in 1055 parameters for the triple-well experiment ($r_{\texttt{max}} = 1000$, $d = 10$) and 1465 parameters for the for the alanine dipeptide experiment ($r_{\texttt{max}} = 1000$, $d = 30$).

The number of parameters in the neural network depends on the input and output dimension, and the sizes of the hidden layers. All the networks we considered were constructed with L hidden layers each with M neurons. The output dimension is always 1 as we are estimating the committor and the input dimension depends on the problem. The total number of parameters is then

$$M(d+1) + (L-1)M(M+1) + (M+1),$$

resulting in 1081 parameters for the triple well experiment ($L = 2$, $M = 27$, $d = 10$) and 1481 parameters for the triple well experiment ($L = 3$, $M = 20$, $d = 30$). The sizes of these networks were chosen through a grid search constrained to the number of parameters in the largest FCM approximation. Larger NN models may lead to improved accuracy [137] or models that are more suited to the committor problem. However, the larger models would prove more challenging to train and we believe the network is sufficent to showcase the promise of the FCM.

Table 4.1: A table summarizing the number of parameters used in the triple well and alanine dipeptide experiments by the FCM and NN.

Experiment	Dimension	FCM parameters	NN parameters
Triple Well	10	1055	1081
Alanine Dipeptide	30	1465	1481

The neural networks were written using the PyTorch package [138] and were trained using the AdamW optimizer [139]. Code for both the FCM and neural networks are available on GitHub [140]. The neural networks used a tanh nonlinearity between hidden layers and a sigmoid activation for the final layer as $q \in [0, 1]$. Before training either method, the data was split into train/validation sets and a separate test set was reserved to evaluate model performance. The FCM used a 10% of the data as a validation set and the NN used 20% of the data. A NN epoch is a single pass through all data points with updates to the model parameters after each mini-batch of size 500 points. To halt training, a maximum number of epochs was set but we also employed a

version early stopping to improve training speed and combat overfitting [33]. After every t epochs (in practice we let $t = 1$), the model parameters were saved and the NN was evaluated on the validation set using the loss function (4.13). If no improvement in the best loss was observed after 20 epochs, the model was saved and training was concluded. In comparison, when training the FCM, we found that stopping after the 5^{th} iteration was sufficient, and did not require a complicated stopping criteria.

Futhermore, compared to the FCM, the hyperparameter selection of neural networks is less straightforward. Though default values can be sufficient, as in AdamW's moment decay values, ϵ, and weight decay, neural networks can be very sensitive to the choice of the learning rate (Figure 4.4.b). Too large of a learning rate can decrease the network's accuracy and too small of a learning rate negatively impacts the convergence rate. Additionally, tuning the learning rate may require retuning the patience parameter to not stop the NN too soon in its training. The learning rates chosen were 10^{-4} in the triple-well experiment and 5×10^{-4} in the alanine dipeptide experiment. NN parameters are summarized in Table 4.2.

Table 4.2: A table summarizing the choices of hyperparameters used for the triple well (TW) and alanine dipeptide (AD) experiments in the neural network.

Experiment	Dimension	Layers	Neurons	Nonlinearity	Activation	Patience	Learning Rate
TW	10	2	27	tanh	sigmoid	20	10^{-4}
AD	30	3	20	tanh	sigmoid	20	5×10^{-4}

We conclude that the parameter choices for the FCM are much more straightforward and produce stable approximations more predictably. On the contrary, the choice of parameters in neural networks, and the network architecture itself, is difficult and without much guidance.

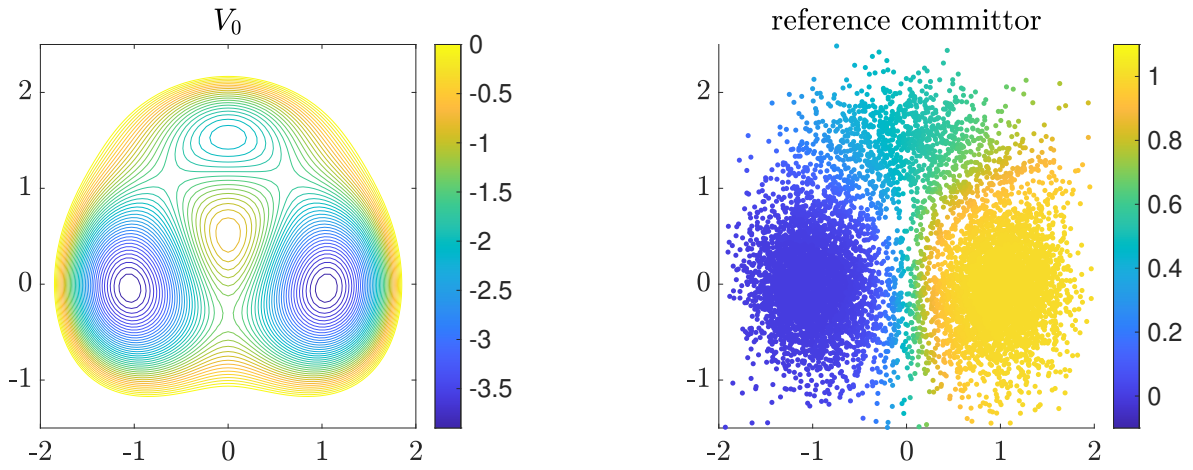


Figure 4.5: (a) The potential function V_0. (b) The reference committor evaluated on the reference data points.

4.5 Results

4.5.1 Triple Well

Our first experiment uses the three-well potential V_0 [12] to construct the potential function $V(x)$

$$V(\boldsymbol{x}) = V_0(\boldsymbol{e}_1^{\mathbf{T}}\boldsymbol{x}, \boldsymbol{e}_2^{\mathbf{T}}\boldsymbol{x}) + 2\sum_{i=3}^{1} 0\left(\boldsymbol{e}_i^{\mathbf{T}}\boldsymbol{x}\right)^2, \tag{4.22}$$

depicted in Figure 4.5.a, where $\boldsymbol{e}_i$ denotes the i^{th} standard basis vector. We consider a system governed by overdamped Langevin dynamics

$$d\boldsymbol{x}(t) = -\nabla V(\boldsymbol{x}(t))dt + \sqrt{2\beta^{-1}}d\omega(t),$$

using this potential, inverse temperature $\beta > 0$, and Weiner process $\omega(t)$. The associated equilibrium density is the Gibbs measure

$$\mu(\boldsymbol{x}) = \frac{e^{-\beta V(\boldsymbol{x})}}{\int e^{-\beta V(\boldsymbol{x})} d\boldsymbol{y}}.$$

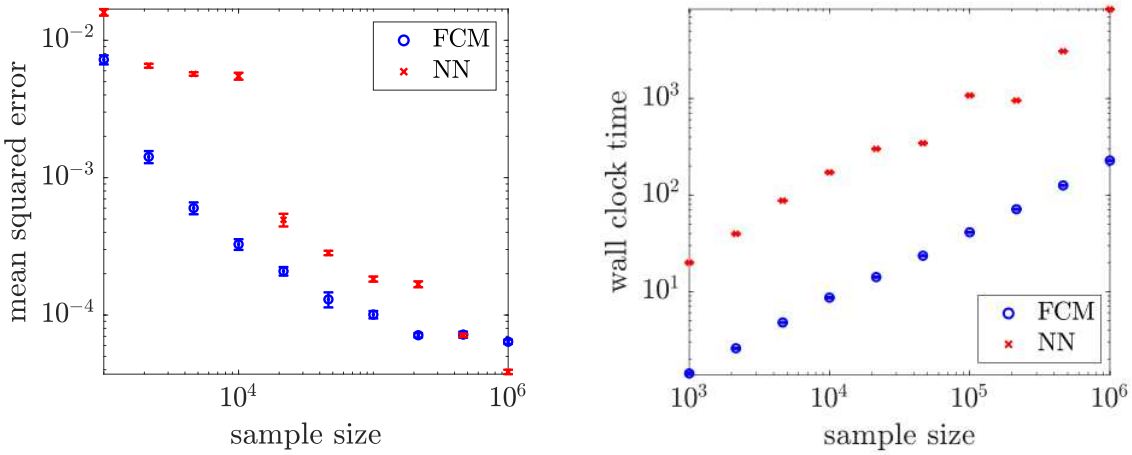


Figure 4.6: Comparison of neural net (NN) and FCM performance for the triple-well system, with standard error bars computed from 10 independent runs of the FCM. (a) Mean squared error computed using the reference committor. (b) Runtime in seconds.

This allows us to easily reweight data run at a higher temperature $\beta_S = 1$, to generate samples of the phase space. These points defined the sampling distribution once the phase space was deemed to be sufficiently well sampled (10^6 time steps of evolution). Since our target distribution was $\beta = 2$, the initial states were given weights

$$w_n = e^{(\beta_S - \beta)V(\boldsymbol{x}_n)}.$$

The output points were determined by advancing the system from $\{x_n\}_{n=1}^{N=10^6}$ forward $\tau = 10^{-2}$ time units at the lower temperature yielding points $\boldsymbol{y}_n = \boldsymbol{x}_n(\tau)$.

To estimate the committor for this problem, the states

$$A = \{\boldsymbol{x} \in \mathbb{R}^{10} \colon (\boldsymbol{e}_1^T \boldsymbol{x} + 1)^2 + (\boldsymbol{e}_2^T \boldsymbol{x})^2 \leq 0.3^2\},$$
$$B = \{\boldsymbol{x} \in \mathbb{R}^{10} \colon (\boldsymbol{e}_1^T \boldsymbol{x} - 1)^2 + (\boldsymbol{e}_2^T \boldsymbol{x})^2 \leq 0.3^2\},$$

are defined, depending only on the first two coordinates. As both the potential (4.22) and the set definitions of A and B depend only on the first two dimensions, finite element methods can be

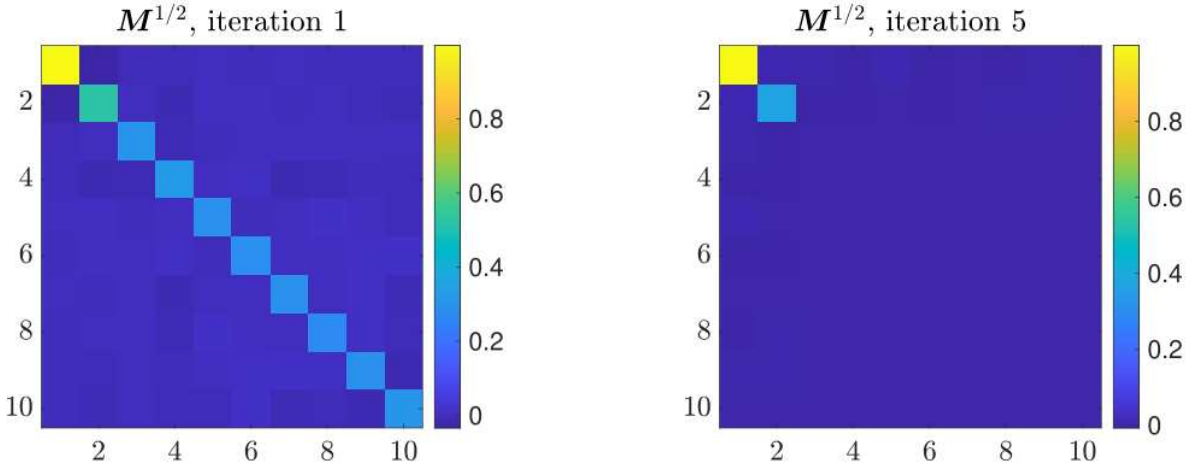
used to find a reference for the committor function Figure 4.5.b. The additional dimensions are noise, meant to increase the complexity of the committor approximation.

The FCM and NN were applied to 10 data sizes, logarithmically spaced between $N = 10^3$ and the full data set $N = 10^6$. The smaller data sets were obtained by uniformly sampling the full data set at random. Each data set was evaluated 10 times and the data was resampled for each repetition. The data was split according to the procedure described in Section 4.4 and the performance of each method was evaluated on the same, separate, reference set. Figure 4.6 evaluates the capability of the two models in estimating the committor function. For all sample sizes, the FCM trains more quickly than the NN and achieved better or comparable error. This observation held true for smaller nets though the difference in training speed was less pronounced. In Figure 4.6.a, the network size and small learning rate do not allow the NN to learn the committor well when the size of the data set is less than 10^4. Faster learning rates do help the convergence rate but are much noiser (see Figure 4.4), reminiscent of the discussion at the end of Section 4.4.

This robustness in training suggests the promise of the FCM. A more sophisticated neural network may provide more favorable results but would require added complexity. An additional benefit of running the FCM is the interpretability of the result. Naively inspecting the potential of Figure 4.5, one might conclude that the most important coordinate is the first dimension and the only other relevant coordinate is the second dimension; simply as a result of how the problem was constructed. In Figure 4.7, the square root of the scaling matrix uncovers exactly this information. The mapping $\boldsymbol{M}^{1/2}$ eliminates the unnecessary nuisance dimensions and the leading eigenvector closely aligns with $\boldsymbol{e}_1$.

4.5.2 Alanine

The preceding example is promising for the FCM but does not exhibit the complexity of real-world problems. While low dimensional evolution is not uncommon in the study of high-dimensional biomolecules [17], the performance on a toy problem may remain unconvincing. To this end, we again turn to alanine dipeptide. In Chapter 3, we discussed the low dimensional $\phi - \psi$



(a) **(b)**

Figure 4.7: Square root of scaling matrix $\boldsymbol{M}$ for the triple-well system when $N = 10^6$ and $r = 1000$. (a) After 1 iteration. (b) After 5 iterations, corresponding to convergence.

space that is often employed to describe the molecule. Figure 4.8.a uses this space to describe the free energy surface and indicates two metastable states.

In this experiment however, the full molecule was simulated and the resulting dataset was 30 dimensional, including the (x, y, z)-coordinates for the ten backbone atoms. The sets A and B are not as easy to describe in $\mathbb{R}^{30}$ but Figure 4.8.b offers a view of them in $\phi - \psi$ space. Similarly, computing a reference committor in such high dimensions is not feasible and we instead run the FCM on the largest data set ($N = 10^6$) with approximation rank $r = 2000$.

The story is largely the same as the triple-well potential as seen in Figure 4.9 where the performance of the FCM and NN are compared under the same methodology. However, in this experiment, it is more likely that the early plateau in NN performance ($N \lessapprox 10^4$) is a result of the complexity of the problem and the sample size not being large enough for sufficient learning to take place. Though different neural networks may perform differently, these initial results suggest that the FCM is a competitive method in modeling the alanine dipeptide committor.

Finally, Figure 4.10.a shows the square root of the scaling matrix $\boldsymbol{M}$. The mapping identifies a two-dimensional subspace where the variation of the committor takes place. The top 10 eigenvalues are given in Figure 4.10.b and Figure 4.10.c shows the reference committor data mapped

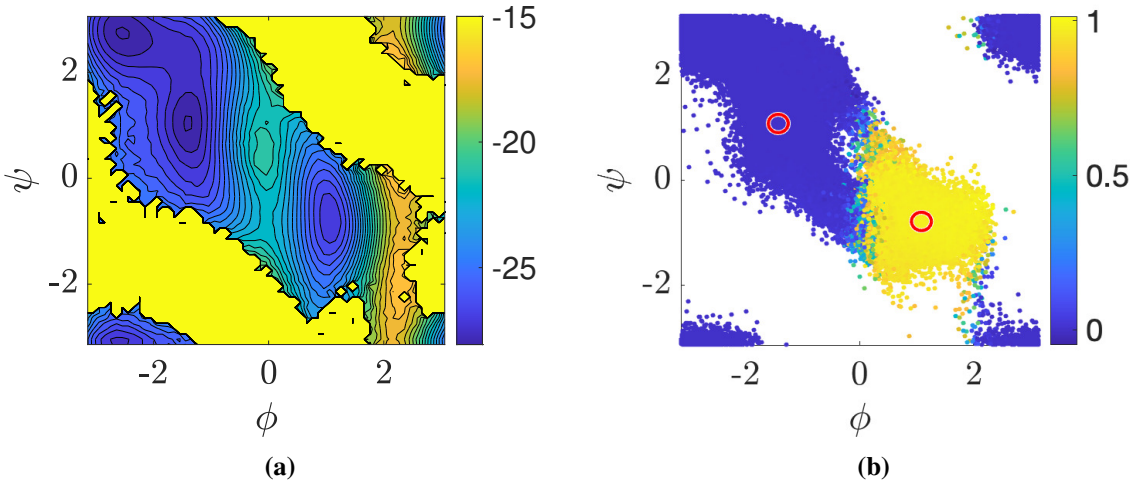


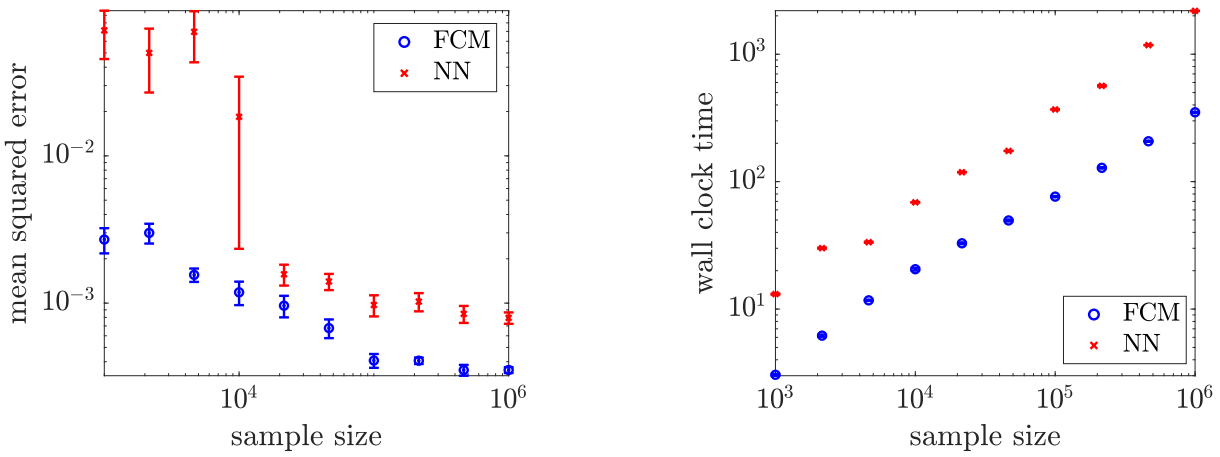
Figure 4.8: (a) $\phi - \psi$ depiction of the free energy surface for alanine dipeptide. (b) $\phi - \psi$ projection of the reference committor with states A and B circled.

onto the top two eigenvectors. This subspace proves to be remarkably well suited to describing the given committor problem. Linear regression shows that the two-dimensional subspace found by the FCM explains 95% of the variance of the committor compared to only 62% using $\phi - \psi$ coordinates. While the latter is historically easier to compute, it is a nonlinear mapping compared to the linear mapping identified in the FCM.

4.6 Discussion

The preliminary results presented in Section 4.5 indicate the FCM has the potential of an efficient, competitive solver for the committor problem. The highlight of the FCM is its ability to uncover low-dimensional subspaces to describe the variance in the committor. The true realization of this potential rests in testing against higher-dimensional problems of molecular dynamics. In part, the difficulty of benchmarking methods on the more complex problems of computational chemistry lies in obtaining reliable reference values and is beyond the scope of this thesis.

Additionally, there are fundamental mathematical questions introduced by this project. The convergence of the FCM is one confounding topic as well as if the exponential kernel presented in Section 4.3 is indeed optimal. Radhakrishnan et al. also suggest their methods can be adapted to train neural networks without backpropagation, which can be seen here by attempting to minimize



(a) **(b)**

Figure 4.9: Comparison of neural net (NN) and FCM performance for alanine dipeptide, with standard error bars computed from 10 independent simulations. (a) Mean squared error computed with respect to the reference. (b) Runtime in seconds.

the Dirichlet energy

$$\mathcal{R}(g) = \frac{1}{2}\int |\nabla g(\boldsymbol{z})|^2 d\boldsymbol{z}.$$

This is an intriguing direction and may aid in answering how the linear rescaling is so successful. In regards to $\boldsymbol{M}$, it remains to be seen if there exist nonlinear extensions that may improve upon the existing framework.

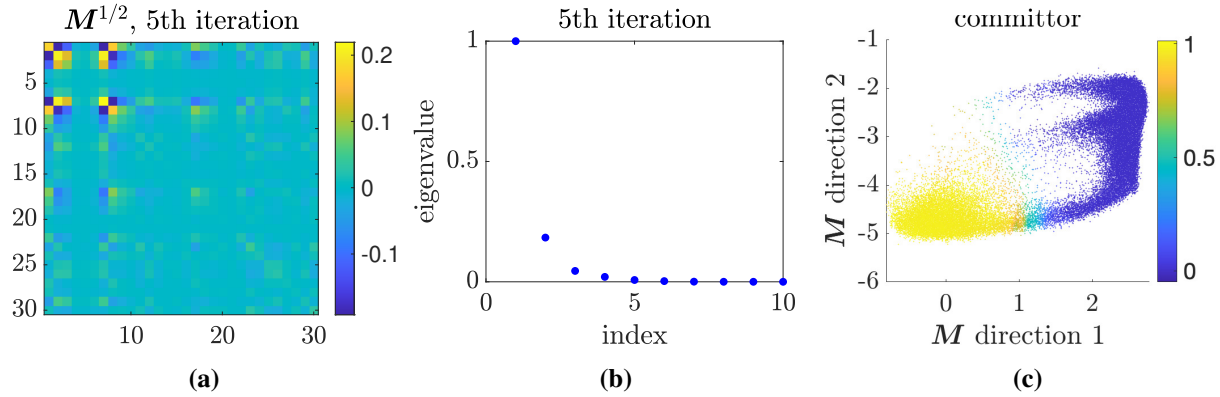


Figure 4.10: FCM applied to alanine dipeptide when $N = 10^6$ and $r = 1000$. (a) $\boldsymbol{M}^{1/2}$ after 5 iterations, corresponding to convergence. (b) Top 10 eigenvalues of $\boldsymbol{M}^{1/2}$, demonstrating low dimensional evolution. (c) Reference committor mapped onto top 2 eigenvectors of $\boldsymbol{M}$.

Chapter 5

Conclusion

This thesis presented two methods that lead to compact models of stochastic processes. Both projects extended earlier work [29, 65] and demonstrated practical data-driven methods to estimate quantities of interest. The two projects used unrelated approaches to achieve their models and both present compelling cases for real-world implementation.

The first project addressed the inherent coarse-graining error that is introduced when constructing Markov state models. By incorporating a decorrelation time, the coarse-grained trajectory could be approximated as a Markov renewal process. This approximation could be made arbitrarily accurate by simply increasing this one parameter as shown by Theorem 3.1.1. While excessively large decorrelation times may result in uninformative models, the numerical results presented in Section 3.4 show that modest decorrelation times still effectively approximate the theoretical renewal process. Additionally, the Mori-Zwanzig formalism was used to model the memory effects involved with modeling as a Markov renewal process. Previously used as an approximation, we showed that the discrete form of the Mori-Zwanzig equation used in Chapter 3 is exact (Theorem 3.3.2) for our projection. Moreover, if the renewal process is Markovian, the Mori-Zwanzig simplifies tremendously. In this case, there is only one non-zero memory kernel as was shown by Theorem 3.3.3.

To be computationally efficient, the Mori-Zwanzig equation needs to be truncated in practice. For this reason, the parameters t_{mem} and t_{max} were introduced. These parameters controlled the memory time horizon and number of terms in the loss function (3.25) and a principled choice for these parameters remains a future goal. Instead, a grid search was used to find sufficiently accurate parameters in the numerical tests. Compared to the Markov state models, the approach of Chapter 3 yielded models that were significantly more descriptive and required trajectories that were roughly an order of magnitude shorter than those needed to construct the Markov models.

In addition to the lack of formal direction for choosing the model parameters, modeling with renewal processes increased the number of parameters that needed to be optimized. Compared with more standard Markov models, these complications initially appear unappealing. However, the direction for improving the model performance is clear, as discussed in Chapter 3.4. Furthermore, we found that a relatively small number of memory kernels were needed to achieve significantly better results than Markov state models. Therefore, we argue that the increase in model complexity is worth the improved performance.

Moving forward, describing the memory kernel decay in general and the possible relation to the decorrelation time remains a future investigation. Additionally, more complicated problems may expose new considerations beyond what was discussed here. Looking beyond Markov models, connections between this work and extended dynamic mode decomposition [141] seem apparent yet unexplored.

The second project covered in this thesis approached model reduction in an orthogonal way to the first project. Instead of explicitly reducing the dimension, kernel functions were used to represent a high dimensional space to estimate the committor function. This is an extension of recent work that introduced the neural feature ansatz (and average gradient outer product) to define the recursive feature machine. These pieces iteratively uncovered a low-dimensional representation of the data. In this thesis, we propose motivation for why these iterative approaches are successful. In Proposition 4.3.2, the featurization defined by the recursive feature machine and the method introduced in Section 4.3 (the Fast Committor Machine) were shown to lead to isotropic gradients in the feature space. This result justified the Gaussian form of the kernel used in the approximation and even inspired future directions of research. The most immediate new direction considers using Dirichlet energy to define the optimal featurization. Moreover, with some assumptions, the neural feature ansatz of Radhakrishnan et al. can be recovered. Beyond hinting at deeper mathematical results may, it may lead to more general, non-linear, featurizations.

The other major question of this project was how to handle large datasets involved in high-dimensional molecular dynamics. Instead of computing all pair-wise entries $k(\boldsymbol{x}_m, \boldsymbol{x}_n)$, Randomly

Pivoted Cholesky [37] was used to construct a low rank approximation of the kernel matrix. This solved a problem left unanswered in the original work of Radhakrishnan et al. After combining these pieces, the Fast Committor Machine (Algorithm 3) was detailed and used in numerical tests to find the committor function. This new approach for iterative featurization resulted in performance competitive with fully-connected neural networks, though tt remains an open question which additional techniques will be necessary to achieve parity in general. Nonetheless, this early version of the Fast Committor Machine was faster and easier to train than a comparable neural network. An additional benefit of the Fast Committor Machine was the matrix M which captures the low-dimensional manifold explaining the variance in the committor function. These initial results suggest several exciting avenues beyond merely considering the best feature maps. Alternative directions could explore what uses there are for M or if the Dirichlet energy lens could unveil more of the black box enclosing machine learning.

Bibliography

[1] G.A. Huber and S. Kim. Weighted-ensemble brownian dynamics simulations for protein association reactions. *Biophysical Journal*, 70(1):97–110, January 1996.

[2] David R. Evans, Hyunwook S. Kwak, David J. Giesen, Alexander Goldberg, Mathew D. Halls, and Masahito Oh-e. Estimation of charge carrier mobility in amorphous organic materials using percolation corrected random-walk model. *Organic Electronics*, 29:50–56, February 2016.

[3] Bibhab Bandhu Majumdar, Vera Prytkova, Eric K. Wong, J. Alfredo Freites, Douglas J. Tobias, and Matthias Heyden. Role of conformational flexibility in monte carlo simulations of many-protein systems. *Journal of Chemical Theory and Computation*, 15(2):1399–1408, January 2019.

[4] Matthew C. Zwier, Adam J. Pratt, Joshua L. Adelman, Joseph W. Kaus, Daniel M. Zuckerman, and Lillian T. Chong. Efficient atomistic simulation of pathways and calculation of rate constants for a protein–peptide binding process: Application to the MDM2 protein and an intrinsically disordered p53 peptide. *The Journal of Physical Chemistry Letters*, 7(17):3440–3445, August 2016.

[5] Anu Nagarajan, Juan R. Perilla, and Thomas B. Woolf. Understanding the conformational changes in ca-APTase using coarse-grained and all-atom simulations with dynamic importance sampling. *Biophysical Journal*, 96(3):430a, February 2009.

[6] Lillian T Chong, Ali S Saglam, and Daniel M Zuckerman. Path-sampling strategies for simulating rare events in biomolecular systems. *Current Opinion in Structural Biology*, 43:88–94, April 2017.

[7] Martin Karplus and Gregory A. Petsko. Molecular dynamics simulations in biology. *Nature*, 347(6294):631–639, October 1990.

[8] Josep Gelpi, Adam Hospital, Ramón Goñi, and Modesto Orozco. Molecular dynamics simulations: advances and applications. *Advances and Applications in Bioinformatics and Chemistry*, page 37, November 2015.

[9] Scott A. Hollingsworth and Ron O. Dror. Molecular dynamics simulation for all. *Neuron*, 99(6):1129–1143, September 2018.

[10] Jacob D Durrant and J Andrew McCammon. Molecular dynamics simulations and drug discovery. *BMC Biology*, 9(1), October 2011.

[11] 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, September 2022.

[12] Frédéric Cérou, Arnaud Guyader, Tony Lelièvre, and David Pommier. A multiple replica approach to simulate reactive trajectories. *The Journal of chemical physics*, 134(5), 2011.

[13] G.M. Torrie and J.P. Valleau. Nonphysical sampling distributions in monte carlo free-energy estimation: Umbrella sampling. *Journal of Computational Physics*, 23(2):187–199, February 1977.

[14] Alessandro Laio and Michele Parrinello. Escaping free-energy minima. *Proceedings of the National Academy of Sciences*, 99(20):12562–12566, September 2002.

[15] Anton K. Faradjian and Ron Elber. Computing time scales from reaction coordinates by milestoning. *The Journal of Chemical Physics*, 120(23):10880–10889, June 2004.

[16] Christoph Dellago, Peter G. Bolhuis, Félix S. Csajka, and David Chandler. Transition path sampling and the calculation of rate constants. *The Journal of Chemical Physics*, 108(5):1964–1977, February 1998.

[17] 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, 2018.

[18] William C. Swope, Jed W. Pitera, and Frank Suits. Describing protein folding kinetics by molecular dynamics simulations. 1. theory. *The Journal of Physical Chemistry B*, 108(21):6571–6581, April 2004.

[19] 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, March 2019.

[20] Qianxiao Li, Bo Lin, and Weiqing Ren. Computing committor functions for the study of rare events using deep learning. *The Journal of Chemical Physics*, 151(5), August 2019.

[21] Jiuxiang Gu, Zhenhua Wang, Jason Kuen, Lianyang Ma, Amir Shahroudy, Bing Shuai, Ting Liu, Xingxing Wang, Gang Wang, Jianfei Cai, et al. Recent advances in convolutional neural networks. *Pattern recognition*, 77:354–377, 2018.

[22] Pin Wu, Junwu Sun, Xuting Chang, Wenjie Zhang, Rossella Arcucci, Yike Guo, and Christopher C Pain. Data-driven reduced order model with temporal convolutional neural network. *Computer Methods in Applied Mechanics and Engineering*, 360:112766, 2020.

[23] L. Evans, M. K. Cameron, and P. Tiwary. Computing committors via Mahalanobis diffusion maps with enhanced sampling data. *The Journal of Chemical Physics*, 157(21):214107, December 2022.

[24] Frank Noé and Feliks Nüske. A variational approach to modeling slow processes in stochastic dynamical systems. *Multiscale Modeling & Simulation*, 11(2):635–655, 2013.

[25] I. T. Jolliffe. *Principal Component Analysis*. Springer-Verlag, 2002.

[26] Laurens van der Maaten and Geoffrey E. Hinton. Visualizing data using t-sne. *Journal of Machine Learning Research*, 9:2579–2605, 2008.

[27] Leland McInnes, John Healy, Nathaniel Saul, and Lukas Großberger. Umap: Uniform manifold approximation and projection. *Journal of Open Source Software*, 3(29):861, September 2018.

[28] Arthur F. Voter. *Introduction to the Kinetic Monte Carlo Method*, page 1–23. Springer Netherlands, 2007.

[29] Adityanarayanan Radhakrishnan, Daniel Beaglehole, Parthe Pandit, and Mikhail Belkin. Mechanism for feature learning in neural networks and backpropagation-free machine learning models. *Science*, 383(6690):1461–1467, March 2024.

[30] B. O. Koopman. Hamiltonian systems and transformation in hilbert space. *Proceedings of the National Academy of Sciences*, 17(5):315–318, May 1931.

[31] G. D. Birkhoff and B. O. Koopman. Recent contributions to the ergodic theory. *Proceedings of the National Academy of Sciences*, 18(3):279–282, March 1932.

[32] Hassan Arbabi and Igor Mezić. Ergodic theory, dynamic mode decomposition, and computation of spectral properties of the koopman operator. *SIAM Journal on Applied Dynamical Systems*, 16(4):2096–2126, January 2017.

[33] Christopher M. Bishop. *Pattern Recognition and Machine Learning (Information Science and Statistics)*. Springer-Verlag, Berlin, Heidelberg, 2006.

[34] John Shawe-Taylor and Nello Cristianini. Preface. In *Kernel Methods for Pattern Analysis*, pages xi–xiv. Cambridge University Press, Cambridge, June 2004.

[35] Motonobu Kanagawa, Philipp Hennig, Dino Sejdinovic, and Bharath K Sriperumbudur. Gaussian processes and kernel methods: A review on connections and equivalences, 2018.

[36] Valérian Jacques-Dumas, René M. van Westen, Freddy Bouchet, and Henk A. Dijkstra. Data-driven methods to estimate the committor function in conceptual ocean models. *Nonlinear Processes in Geophysics*, 30(2):195–216, June 2023. Publisher: Copernicus GmbH.

[37] Yifan Chen, Ethan N. Epperly, Joel A. Tropp, and Robert J. Webber. Randomly pivoted Cholesky: Practical approximation of a kernel matrix with few entry evaluations, February 2023. arXiv:2207.06503 [cs, math, stat].

[38] Per Kragh Andersen, Ørnulf Borgan, Richard D. Gill, and Niels Keiding. *Statistical Models Based on Counting Processes*. Springer US, 1993.

[39] Howard M. Taylor Samuel Karlin. *A First Course in Stochastic Processes*. Elsevier, 1975.

[40] Wolfgang Paul and Jörg Baschnagel. Stochastic processes. *From Physics to finance*, 1999.

[41] Tomasz Rolski, Hanspeter Schmidli, Volker Schmidt, and Jozef L Teugels. *Stochastic processes for insurance and finance*. John Wiley & Sons, 2009.

[42] Etienne Pardoux. *Markov processes and applications: algorithms, networks, genome and finance*. John Wiley & Sons, 2008.

[43] Sean R Eddy. Hidden markov models. *Current opinion in structural biology*, 6(3):361–365, 1996.

[44] Sean R Eddy. What is a hidden markov model? *Nature biotechnology*, 22(10):1315–1316, 2004.

[45] Lawrence R Rabiner. A tutorial on hidden markov models and selected applications in speech recognition. *Proceedings of the IEEE*, 77(2):257–286, 1989.

[46] Bhavya Mor, Sunita Garhwal, and Ajay Kumar. A systematic review of hidden markov models and their applications. *Archives of computational methods in engineering*, 28:1429–1448, 2021.

[47] Stephen Brooks. Markov chain monte carlo method and its application. *Journal of the royal statistical society: series D (the Statistician)*, 47(1):69–100, 1998.

[48] Walter R Gilks, Sylvia Richardson, and David Spiegelhalter. *Markov chain Monte Carlo in practice*. CRC press, 1995.

[49] Siddhartha Chib and Edward Greenberg. Understanding the metropolis-hastings algorithm. *The american statistician*, 49(4):327–335, 1995.

[50] Jonathan Weare. Particle filtering with path sampling and an application to a bimodal ocean current model. *J. Comput. Phys.*, 228(12):4312–4331, July 2009.

[51] Robert J Webber, David A Plotkin, Morgan E O'Neill, Dorian S Abbot, and Jonathan Weare. Practical rare event sampling for extreme mesoscale weather. *Chaos*, 29(5):053109, May 2019.

[52] 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. *J. Atmos. Sci.*, 80(2):519–534, February 2023.

[53] 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), April 2023.

[54] Yves Pomeau. Front motion, metastability and subcritical bifurcations in hydrodynamics. *Physica D: Nonlinear Phenomena*, 23(1-3):3–11, 1986.

[55] Andrew A Fingelkurts and Alexander A Fingelkurts. Making complexity simpler: multivariability and metastability in the brain. *International Journal of Neuroscience*, 114(7):843–862, 2004.

[56] Clayton Haldeman and John M Beggs. Critical branching captures activity in living neural networks and maximizes the number of metastable states. *Physical review letters*, 94(5):058101, 2005.

[57] Federico Cavanna, Martina G Vilas, Matías Palmucci, and Enzo Tagliazucchi. Dynamic functional connectivity and brain metastability during altered states of consciousness. *Neuroimage*, 180:383–395, 2018.

[58] Pierre Collet, Servet Martínez, and Jaime San Martín. *Quasi-Stationary Distributions: Markov Chains, Diffusions and Dynamical Systems*. Springer Berlin Heidelberg, 2013.

[59] Nicolas Champagnat and Denis Villemonais. General criteria for the study of quasi-stationarity. *Electronic Journal of Probability*, 28(none), January 2023.

[60] 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), July 2013.

[61] Peter Deuflhard and Marcus Weber. Robust perron cluster analysis in conformation dynamics. *Linear Algebra and its Applications*, 398:161–184, March 2005.

[62] Stuart P. Lloyd. Least squares quantization in pcm. *IEEE Trans. Inf. Theory*, 28:129–136, 1982.

[63] James MacQueen et al. Some methods for classification and analysis of multivariate observations. In *Proceedings of the fifth Berkeley symposium on mathematical statistics and probability*, volume 1 (14), pages 281–297. Oakland, CA, USA, 1967.

[64] Feliks Nüske, Hao Wu, Jan-Hendrik Prinz, Christoph Wehmeyer, Cecilia Clementi, and Frank Noé. Markov state models from short non-equilibrium simulations—analysis and correction of estimation bias. *The Journal of Chemical Physics*, 146(9), March 2017.

[65] Animesh Agarwal, Sandrasegaram Gnanakaran, Nicholas Hengartner, Arthur F. Voter, and Danny Perez. Arbitrarily accurate representation of atomistic dynamics via markov renewal processes, 2020.

[66] Michael Levitt and Arieh Warshel. Computer simulation of protein folding. *Nature*, 253(5494):694–698, February 1975.

[67] Sergei Izvekov, Angela Violi, and Gregory A. Voth. Systematic coarse-graining of nanoparticle interactions in molecular dynamics simulation. *The Journal of Physical Chemistry B*, 109(36):17019–17024, August 2005.

[68] W. G. Noid, Jhih-Wei Chu, Gary S. Ayton, Vinod Krishna, Sergei Izvekov, Gregory A. Voth, Avisek Das, and Hans C. Andersen. The multiscale coarse-graining method. i. a rigorous bridge between atomistic and coarse-grained models. *The Journal of Chemical Physics*, 128(24), June 2008.

[69] Gregory R. Bowman, Vijay S. Pande, and Frank Noé. *An Introduction to Markov State Models and Their Application to Long Timescale Molecular Simulation*. Springer Netherlands, 2014.

[70] Daniel T. Gillespie. Exact stochastic simulation of coupled chemical reactions. *The Journal of Physical Chemistry*, 81(25):2340–2361, December 1977.

[71] M. G. Guenza, M. Dinpajooh, J. McCarty, and I. Y. Lyubimov. Accuracy, transferability, and efficiency of coarse-grained models of molecular liquids. *The Journal of Physical Chemistry B*, 122(45):10257–10278, August 2018.

[72] Giacomo Di Gesù, Tony Lelièvre, Dorian Le Peutrec, and Boris Nectoux. Jump markov models and transition state theory: the quasi-stationary distribution approach. *Faraday Discussions*, 195:469–495, 2016.

[73] Sheldon M Ross. *Stochastic Processes*. John Wiley & Sons, Nashville, TN, 2 edition, January 1995.

[74] T Lelièvre. Accelerated dynamics: Mathematical foundations and algorithmic improvements. *Eur. Phys. J. Spec. Top.*, 224(12):2429–2444, September 2015.

[75] T. Lelièvre. *Handbook of Materials Modeling: Methods: Theory and Modeling*. Springer, 2020.

[76] Erhan Cinlar. Exceptional paper—markov renewal theory: A survey. *Management Science*, 21(7):727–752, March 1975.

[77] David A Levin and Yuval Peres. *Markov chains and mixing times*, volume 107. American Mathematical Soc., 2017.

[78] Hazime Mori. Transport, collective motion, and brownian motion. *Progress of Theoretical Physics*, 33(3):423–455, March 1965.

[79] Robert Zwanzig. Nonlinear generalized langevin equations. *Journal of Statistical Physics*, 9(3):215–220, November 1973.

[80] Denis J. Evans and Gary P. Morris. *Statistical Mechanics of Nonequilibrium Liquids*. Academic Press, London, 1990.

[81] A J Chorin, O H Hald, and R Kupferman. Optimal prediction and the Mori-Zwanzig representation of irreversible processes. *Proc. Natl. Acad. Sci. U. S. A.*, 97(7):2968–2973, March 2000.

[82] Alexandre J. Chorin, Ole H. Hald, and Raz Kupferman. Optimal prediction with memory. *Physica D: Nonlinear Phenomena*, 166(3–4):239–257, June 2002.

[83] Denis J. Evans and Gary Morriss. *Statistical Mechanics of Nonequilibrium Liquids*. Cambridge University Press, May 2008.

[84] Sergei Izvekov. Mori-zwanzig theory for dissipative forces in coarse-grained dynamics in the markov limit. *Physical Review E*, 95(1), January 2017.

[85] Jeroen Wouters and Valerio Lucarini. Multi-level dynamical systems: Connecting the ruelle response theory and the mori-zwanzig approach. *Journal of Statistical Physics*, 151(5):850–860, March 2013.

[86] CA Marsh, G Backx, and MH Ernst. Static and dynamic properties of dissipative particle dynamics. *Physical Review E*, 56(2):1676, 1997.

[87] Eric J. Parish and Karthik Duraisamy. Non-markovian closure models for large eddy simulations using the mori-zwanzig formalism. *Physical Review Fluids*, 2(1), January 2017.

[88] Weiqi Chu and Xiantao Li. The mori–zwanzig formalism for the derivation of a fluctuating heat conduction model from molecular dynamics. *Communications in Mathematical Sciences*, 17(2):539–563, 2019.

[89] Michael te Vrugt, Sabine Hossenfelder, and Raphael Wittkowski. Mori-zwanzig formalism for general relativity: A new approach to the averaging problem. *Physical Review Letters*, 127(23), December 2021.

[90] Yen Ting Lin, Yifeng Tian, Daniel Livescu, and Marian Anghel. Data-driven learning for the mori–zwanzig formalism: A generalization of the koopman learning framework. *SIAM Journal on Applied Dynamical Systems*, 20(4):2558–2601, January 2021.

[91] Robert W Zwanzig. *Nonequilibrium Statistical Mechanics*. Oxford University Press, New York, NY, May 2001.

[92] 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. *The Journal of Chemical Physics*, 153(1), July 2020.

[93] 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.

[94] Matthew M. Copeland, Hung N. Do, Lane Votapka, Keya Joshi, Jinan Wang, Rommie E. Amaro, and Yinglong Miao. Gaussian accelerated molecular dynamics in openmm. *The Journal of Physical Chemistry B*, 126(31):5810–5820, July 2022.

[95] Divesh Bhatt and Daniel M. Zuckerman. Beyond microscopic reversibility: Are observable nonequilibrium processes precisely reversible? *Journal of Chemical Theory and Computation*, 7(8):2520–2527, July 2011.

[96] Songyuan Yao, Richard Van, Xiaoliang Pan, Ji Hwan Park, Yuezhi Mao, Jingzhi Pu, Ye Mei, and Yihan Shao. Machine learning based implicit solvent model for aqueous-solution alanine dipeptide molecular dynamics simulations. *RSC Advances*, 13(7):4565–4577, 2023.

[97] Sathish Dasari and Bhabani S. Mallik. Conformational free-energy landscapes of alanine dipeptide in hydrated ionic liquids from enhanced sampling methods. *The Journal of Physical Chemistry B*, 124(31):6728–6737, July 2020.

[98] Amil G Anderson and Jan Hermans. Microfolding: conformational probability map for the alanine dipeptide in water from molecular dynamics simulations. *Proteins: Structure, Function, and Bioinformatics*, 3(4):262–265, 1988.

[99] T. W. Anderson. On the distribution of the two-sample cramer-von mises criterion. *The Annals of Mathematical Statistics*, 33(3):1148–1159, September 1962.

[100] David Aristoff and Daniel M Zuckerman. Optimizing weighted ensemble sampling of steady states. *Multiscale Modeling & Simulation*, 18(2):646–673, 2020.

[101] 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:2110.05050 [cond-mat, physics:physics]*, October 2021. arXiv: 2110.05050.

[102] Alexander M. Berezhkovskii and Attila Szabo. Committors, first-passage times, fluxes, Markov states, milestones, and all that. *The Journal of Chemical Physics*, 150(5):054106, February 2019. Publisher: American Institute of Physics.

[103] Haoya Li, Yuehaw Khoo, Yinuo Ren, and Lexing Ying. A semigroup method for high dimensional committor functions based on neural network. In *Proceedings of the 2nd Math-*

ematical and Scientific Machine Learning Conference, pages 598–618. PMLR, April 2022. ISSN: 2640-3498.

[104] Yin-Wen Chang, Cho-Jui Hsieh, Kai-Wei Chang, Michael Ringgaard, and Chih-Jen Lin. Training and testing low-degree polynomial data mappings via linear svm. *The Journal of Machine Learning Research*, 11:1471–1490, August 2010.

[105] Quanzhong Liu, Chihau Chen, Yang Zhang, and Zhengguo Hu. Feature selection for support vector machines with rbf kernel. *Artificial Intelligence Review*, 36(2):99–115, February 2011.

[106] Yingchao Xiao, Huangang Wang, Lin Zhang, and Wenli Xu. Two methods of selecting gaussian kernel parameters for one-class svm and their application to fault detection. *Knowledge-Based Systems*, 59:75–84, March 2014.

[107] Robert Jenssen, Deniz Erdogmus, Jose Principe, and Torbjørn Eltoft. *The Laplacian PDF Distance: A Cost Function for Clustering in a Kernel Feature Space*, volume 17. MIT Press, 2004.

[108] N. Aronszajin1950. Theory of reproducing kernels. *Transactions of the American Mathematical Society*, 68(3):337–404, May 1950.

[109] E. H. Moore. On properly positive hermitian matrices. *Bull. Amer. Math. Soc.*, 23:59, 1916.

[110] E. H. Moore. General analysis. *Memoirs of the American Philosophical Society, Part I*, 1935.

[111] E. H. Moore. General analysis. *Memoirs of the American Philosophical Society, Part II*, 1939.

[112] M.A. Hearst, S.T. Dumais, E. Osuna, J. Platt, and B. Scholkopf. Support vector machines. *IEEE Intelligent Systems and their Applications*, 13(4):18–28, 1998.

[113] Mahdi Movahedian Moghaddam, Alireza Afzal Aghaei, and Kourosh Parand. Rational jacobi kernel functions: A novel massively parallelizable orthogonal kernel for support vector machines. In *2024 Third International Conference on Distributed Computing and High Performance Computing (DCHPC)*, pages 1–8. IEEE, 2024.

[114] Xiaofei Liu, Wuqiang Yang, Fan Meng, and Tengchen Sun. Material recognition using robotic hand with capacitive tactile sensor array and machine learning. *IEEE Transactions on Instrumentation and Measurement*, 2024.

[115] Jair Cervantes, Farid Garcia-Lamont, Lisbeth Rodríguez-Mazahua, and Asdrubal Lopez. A comprehensive survey on support vector machine classification: Applications, challenges and trends. *Neurocomputing*, 408:189–215, 2020.

[116] Zhi Hong Kok, Abdul Rashid Mohamed Shariff, Meftah Salem M Alfatni, and Siti Khairunniza-Bejo. Support vector machine in precision agriculture: a review. *Computers and Electronics in Agriculture*, 191:106546, 2021.

[117] Corinna Cortes and Vladimir Vapnik. Support-vector networks. *Machine Learning*, 20(3):273–297, September 1995.

[118] S. Sathiya Keerthi, Shirish K. Shevade, Chiranjib Bhattacharyya, and K. R. K. Murthy. Improvements to platt's smo algorithm for svm classifier design. *Neural Computation*, 13:637–649, 2001.

[119] Carl Edward Rasmussen and Christopher K I Williams. *Gaussian processes for machine learning*. Adaptive Computation and Machine Learning series. MIT Press, London, England, November 2005.

[120] Sadeep Jayasumana, Richard Hartley, Mathieu Salzmann, Hongdong Li, and Mehrtash Harandi. Kernel methods on riemannian manifolds with gaussian rbf kernels. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 37(12):2464–2477, 2015.

[121] Andrea Vedaldi, Varun Gulshan, Manik Varma, and Andrew Zisserman. Multiple kernels for object detection. In *2009 IEEE 12th International Conference on Computer Vision*, pages 606–613, 2009.

[122] Bernhard Scholkopf and Alexander J Smola. *Learning with kernels*. Adaptive Computation and Machine Learning series. MIT Press, London, England, December 2001.

[123] Charu C. Aggarwal, Alexander Hinneburg, and Daniel A. Keim. *On the Surprising Behavior of Distance Metrics in High Dimensional Space*, page 420–434. Springer Berlin Heidelberg, 2001.

[124] Gustavo Camps-Valls, Antonio Rodrigo-Gonzalez, Jordi Munoz-Mari, Luis Gomez-Chova, and Javier Calpe-Maravilla. Hyperspectral image classification with mahalanobis relevance vector machines. In *IEEE International Geoscience and Remote Sensing Symposium*, pages 3802–3805, 2007.

[125] Yuya Kamada and Shigeo Abe. Support vector regression using Mahalanobis kernels. In *IAPR Workshop on Artificial Neural Networks in Pattern Recognition*, 2006.

[126] PC Mahalanobis, Raj Chandra Bose, and Samarendra Nath Roy. Normalisation of statistical variates and the use of rectangular co-ordinates in the theory of sampling distributions. *Sankhyā: The Indian Journal of Statistics (1933-1960)*, 3(1):1–40, 1937.

[127] Ronald R. Coifman and St'ephane Lafon. Diffusion maps. *Applied and Computational Harmonic Analysis*, 21(1):5–30, 2006.

[128] Haochuan Chen, Benoît Roux, and Christophe Chipot. Discovering reaction pathways, slow variables, and committor probabilities with machine learning. *Journal of Chemical Theory and Computation*, 19(14):4414–4426, 2023.

[129] Christof Schütte, Stefan Klus, and Carsten Hartmann. Overcoming the timescale barrier in molecular dynamics: Transfer operators, variational principles and machine learning. *Acta Numerica*, 32:517–673, 2023.

[130] Oliver T. Unke, Stefan Chmiela, Huziel E. Sauceda, Michael Gastegger, Igor Poltavsky, Kristof T. Schütt, Alexandre Tkatchenko, and Klaus-Robert Müller. Machine learning force fields. *Chemical Reviews*, 121(16):10142–10186, 2021.

[131] Christopher K. I. Williams and Matthias Seeger. Using the nyström method to speed up kernel machines. In *Proceedings of the 13th International Conference on Neural Information Processing Systems*, 2000.

[132] Alessandro Rudi, Luigi Carratino, and Lorenzo Rosasco. FALKON: An optimal large scale kernel method. In *Proceedings of the 31st International Conference on Neural Information Processing Systems*, 2017.

[133] Sergei V. Krivov. Protein folding free energy landscape along the committor - the optimal folding coordinate. *Journal of Chemical Theory and Computation*, 14(7):3418–3427, May 2018.

[134] William M. Jacobs and Eugene I. Shakhnovich. Accurate protein-folding transition-path statistics from a simple free-energy landscape. *The Journal of Physical Chemistry B*, 122(49):11126–11136, August 2018.

[135] 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, April 2021.

[136] Agnan Kessy, Alex Lewin, and Korbinian Strimmer. Optimal whitening and decorrelation. *The American Statistician*, 72(4):309–314, January 2018.

[137] Mikhail Belkin, Daniel Hsu, Siyuan Ma, and Soumik Mandal. Reconciling modern machine-learning practice and the classical bias–variance trade-off. *Proceedings of the National Academy of Sciences*, 116(32):15849–15854, 2019.

[138] Adam Paszke, Sam Gross, Francisco Massa, Adam Lerer, James Bradbury, Gregory Chanan, Trevor Killeen, Zeming Lin, Natalia Gimelshein, Luca Antiga, Alban Desmaison, Andreas Kopf, Edward Yang, Zachary DeVito, Martin Raison, Alykhan Tejani, Sasank Chilamkurthy, Benoit Steiner, Lu Fang, Junjie Bai, and Soumith Chintala. Pytorch: An imperative style, high-performance deep learning library. *arXiv preprint arXiv:1912.01703*, 2019.

[139] Ilya Loshchilov and Frank Hutter. Decoupled weight decay regularization. In *Proceedings of the Seventh International Conference on Learning Representations (ICLR)*, 2019.

[140] David Aristoff, Mats Johnson, Gideon Simpson, and Robert Webber. Fast committor machines. https://github.com/davidaristoff/Fast-Committor-Machine, 2024.

[141] 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, June 2015.

[142] Steve O. Nielsen, Carlos F. Lopez, Ivaylo Ivanov, Preston B. Moore, John C. Shelley, and Michael L. Klein. Transmembrane peptide-induced lipid sorting and mechanism of l - to-inverted phase transition using coarse-grain molecular dynamics. *Biophysical Journal*, 87(4):2107–2115, October 2004.

[143] Robert Schaback and Holger Wendland. Kernel techniques: From machine learning to meshless methods. *Acta Numerica*, 15:543–639, May 2006.

[144] J. R. Norris. *Markov Chains*. Cambridge University Press, February 1997.

[145] David Aristoff, Mats Johnson, and Danny Perez. Arbitrarily accurate, nonparametric coarse graining with markov renewal processes and the mori–zwanzig formulation. *AIP Advances*, 13(9), 2023.

[146] D Aristoff, M Johnson, G Simpson, and RJ Webber. The fast committor machine: Interpretable prediction with kernels. *arXiv preprint arXiv:2405.10410*, 2024.