Statistical Mechanics of Lipid Membranes and Data-Driven Reaction Learning

by

Clay H. Batton

A dissertation submitted in partial satisfaction of the

requirements for the degree of

Doctor of Philosophy

in

Chemical Engineering

in the

Graduate Division

of the

University of California, Berkeley

Summer 2022

Statistical Mechanics of Lipid Membranes and Data-Driven Reaction Learning

Abstract

Statistical Mechanics of Lipid Membranes and Data-Driven Reaction Learning

by

Clay H. Batton

Doctor of Philosophy in Chemical Engineering

University of California, Berkeley

Professor Kranthi K. Mandadapu, Chair

In this thesis, we apply a common set of tools to two problems that compose two separate parts. Shared between them is the use of importance sampling, in which techniques such as Monte Carlo and molecular dynamics evolve a system in a way that respects its physical conditions under the effect of thermal fluctuations, and the use of techniques from the finite element method, in which a combination of integration by parts and discretization on function spaces allows one to turn strong differential conditions into weaker integral ones.

In the first part, we consider the modeling of biological membranes, complex materials formed of lipids and proteins that serve as interfacial barriers in cells. At length scales beyond the thickness of a membrane, membrane behavior can be understood using a phenomenological model accounting for fluid in-plane and elastic out-of-plane behavior. A discretization of this model is developed using techniques from the finite element method that is evolved using Metropolis Monte Carlo. Compositional energetics are added to model phase separation into liquid-disordered and liquid-ordered domains that are observed experimentally in multicomponent biological membranes. By including the effect of proteins that can induce domains of the thermodynamically disfavored phase it is found that competition between line tension and curvature prevents macrophase separation and leads to stable microphases, providing a possible explanation for nanoscopic domains hypothesized to exist in cells.

In the second part, we consider numerical methods to find representations of the committor function in rare event processes. The committor function, the probability a configuration will commit to the product state instead of the reactant state, encodes the complete mechanistic information of a process but is costly to compute. Instead, the transition pathway is homogeneously sampled with importance sampling methods in order to solve a variational form of a partial differential equation the committor function satisfies. A neural network is used as a basis function from which optimization informs the neural network parameters. Coupling this process with fitting to empirical estimates of the committor function, the procedure is found to yield accurate estimates of the committor function and reaction rates.

To Uncle Buck and Grandpa

Contents

List of Figures

List of Tables

Acknowledgments

While the rest of this thesis is written in a serious manner, I would like this part to reflect my Eisenhowerian mantra of, "Always take your job seriously, never yourself." Graduate school is a metamorphic time that has seen me personally become more efficient, along with ending up half the man I used to be by eating less and walking up a hill roughly a thousand times. As I started writing this section, I had the intention that I probably wouldn't list a lot of names, but afterwards, I found myself having produced enough to fill a decent chunk of a local Vietnam War Memorial plaque as there are many people I want to enshrine in the annals of history. It truly does take a village to raise a child.

First, I would like to thank my advisor, Kranthi, for his support. When I was applying to graduate school, I was informed by a person in Minnesota that my top choice at Berkeley was in the middle of jumping ship to a university in Europe. For fun I still applied to Berkeley to see if I would get in, having come from a background of doing statistical mechanics and solving differential equations in a manner that is not that common for chemical engineers. Luckily for me, Kranthi was hired that year, being the person to call me on the phone a week or so after my application was submitted to let me know that I got into Berkeley. Being here has truly been a life-changing experience, and I thank Kranthi for letting me stay as long as I have to pick up more of his style and way of doing things. Hopefully, my long list of expressions remains in his vocabulary as his style remains in mine. I also thank Professors Nitash Balsara, Susan Muller, and David Limmer for their support over the years.

Secondly, I would like to thank my groupmates for making this whirlwind of a graduate school experience fun. Amaresh is the other guy who decided to hop into Kranthi's group at the start, a man infinitely more dedicated to his studies than an irreverent like me, but still a joyful soul. I would've wished for no one else to springboard a group with. Jeff joined afterward, a man with eclectic insights that I had great joy talking with about a variety of random subjects. The postdocs, Alex, Joel, Shachi, Vida, Dimitrios, and (for a week or two formally) Katie, served as great inspiration in tackling scientific problems and other parts of life. In particular, I hold a great deal of debt to Dimitrios and Katie for helping out with the post-graduate school job search, without whom I would probably be living in a van down by the river. I have had a great deal of fun "mentoring" (which mostly amounts to me sporadically stopping by random people, asking them how life is going, and then giving a series of Yoda-like riddles of research advice that may or may not be me making things up) the younger students in the group. Muhammad, the first younger student to join, has been a great coworker and friend in everything science, anime, and music. To the other graduate students who joined afterward, Yannick, Cory, Ahmad, Alison, Zach, Alhad, and Josh, and to the undergraduates who stayed for a while such as Bernardo, Alec, Lewis, and Ravtej, you are all truly wonderful.

Thirdly, I would like to thank friends who made up moments both big and small throughout this experience. The first batch of people I lived with in "Hearst Castle", Sudi, Kyle, Julie, and Jon, are a great group of folks that I still look up to, seeing how they are navigating life in much better ways than I am. After they left for better things, I had a reversal of fortune of me going from desperately searching for housing to holding onto three rooms in a house. After a referral from Julie, the second batch of people joined me at “Hearst Castle”, composed of Branden, Johnny, and Zach. I’ve had the pleasure of living with these fine folks throughout the COVID years (I thought if I was locked inside my room for such a prolonged period of time “Old Boy” style it would be the most productive year of my life, but here we are), which has cemented our friendship to the realm of them adopting me into their friend group of the wonderful people in Rose, Edwards, and Russell Street. I still had my own group I rolled with though. Jared served as another kindred spirit in the quest to do simulations, and I thank him for always being a good time since the beginning. There are the random people I have had the pleasure of meeting by virtue of graduate classes or the NPC-like patrol path I take in the basement of Gilman. These folks, Tyler, Scott, Abdul, Norm, Layne, Nikhil, and Alex, have all definitely made my graduate school experience more enjoyable in the random encounters we would have. There is a group of people I associated with online that have also helped in the fun department, and happen to be night owls like me. These folks, Zack, Jake, Chi, Cory, Brynn, Michael, Jamie, Cole, Derek, Megan, Cam, Dan, and Chris, I thank for dragging me away from work every once and a while. There’s also my group of Minnesotan friends dating back to Lakeville South High School that I have kept in contact with over the years, a growing number of which are starting to get married, reminding me that the biological clock is always ticking. To Michael, Jake, the same Zack as directly previously, Nick, Zach, Dan, Kalvin, Sam, Derek, and Riley, stay wonderful, I don’t know how you manage to survive in that frozen wasteland in the winter, Mad Max dystopia in the summer environment, but keep at it. To the rest of my friends from LSHS, if you ever encounter me in the wild, name the two parts of this thesis and I’ll send you a pizza roll in the mail.

Finally, I would like to thank my family for always sticking by me. My dad, Mark, taught me the importance of planning and preparing at a young age, and introduced me to the joys of *Star Wars* on an old laser disc collection where Han shot first. My mom, Holly, reminds me to have fun as I go about trying not to, and serves as the source of most of my Midwestern sayings and a relatively good understanding of *I Love Lucy*. My brother, John, whose much freer thinking and immense perseverance serve as a constant source of inspiration, along with rubbing onto me knowledge of mechanical things without which I would probably lead a Luddite life. My grandma, Marilyn, helped instill many values that I hold dear in my life such as an appreciation for the arts and doing good in the world. Last but not least, to the deceased that could not see my educational journey come to an end. My grandpa, Hubert, was a longtime high school math teacher and served as one of my first scientific role models, a man who always knew what to do and had an infectious goofiness that stays with me to this day. My uncle, Keith (also known as Uncle Buck, partially from the resemblance to the John Candy character of the same name but mostly because he had a lot of money), lived an improbable rise from working in a Pizza Hut to bumping heads with a future US president in business meetings, and helped me see the world in more than just black and white, along with being the source of my movie knowledge and the Rocky Balboa spirit of going the distance. I wish you were both here to see this.

Chapter 1

Introduction

In this thesis, we will consider two different topics analyzed using similar means. The tools we use consist of statistical mechanics and the finite element method. With statistical mechanics, tools from probability theory allow for the analysis of microscopic behavior in order to explain macroscopic phenomena in systems of interest. In these systems we have access to the configuration $\mathbf{x}$ and velocities $\mathbf{v}$ that give rise to a Hamiltonian H consisting of the kinetic energy $T(\mathbf{v})$ and potential energy $V(\mathbf{x})$. Using various methods, such as looking at a large system divided into subsystems, a system connected to a reservoir of energy, volume, and/or number of particles, or maximum entropy arguments [1], one arrives at the probability of observing a state $(\mathbf{x}, \mathbf{v})$ to be

$$\rho(\mathbf{x}, \mathbf{v}) = \frac{\exp(-\beta H(\mathbf{x}, \mathbf{v}))}{Z}\,, \tag{1.0.1}$$

$$Z = \int \mathrm{d}\mathbf{x}\,\mathrm{d}\mathbf{v}\;\; \exp(-\beta H(\mathbf{x}, \mathbf{v}))\,, \tag{1.0.2}$$

where Z is known as the partition function, and Eq. (1.0.1) is known as the Boltzmann distribution. Evaluating Z and having access to $\rho(\mathbf{x}, \mathbf{v})$ allows for the evaluation of quantities such as thermodynamic variables, equilibrium properties, and, through the use of near-equilibrium methods, non-equilibrium properties. As the kinetic energy for particles with masses m_i takes the form

$$T(\mathbf{v}) = \sum_i \frac{m_i}{2} \mathbf{v}_i \cdot \mathbf{v}_i\,, \tag{1.0.3}$$

the contribution of the velocities in Eq. (1.0.2) can be evaluated exactly. However, the potential energy can take a many-body form for which an exact solution of Z is considered a *tour de force.*

Fortunately one does not need to know the exact solution of Z in order to evaluate quantities of interest using computational methods. One such technique, importance sampling, allows for the generation of samples $(\mathbf{x}, \mathbf{v})$ according to $\rho(\mathbf{x}, \mathbf{v})$. Importance sampling consists of evolving systems of interest using either molecular dynamics, in which modifications of Hamilton's equations of motion are integrated forward in time, and/or Monte Carlo, in which random trial moves modify the system's state. In both methods, samples of the system are generated according to $\rho(\mathbf{x}, \mathbf{v})$ and hence the observed samples are more statistically relevant than that generated using truly random estimation of $\mathbf{x}$ and $\mathbf{v}$. As the samples obey the underlying probability distribution of the system, integrals involving $\rho(\mathbf{x}, \mathbf{v})$ can be evaluated with simple averages about the generated samples. In the presence of free energy barriers, the simple application of these sampling methods is not computationally feasible due to the system being trapped in local minima of free energy, with transitions between minima being rare events. Modifications to the importance sampling algorithms, such as umbrella sampling and the finite-temperature string method [2], are then needed to provide efficient sample generation and evaluation of integrals involving $\rho(\mathbf{x}, \mathbf{v})$.

The finite element method (FEM), on the other hand, is a methodology for solving differential equations. FEM consists of two major parts, in which we will follow the presentation of Ref. [3]. The first part is the conversion of a partial differential equation into an integral equation. As an example we consider Poisson's equation

$$\Delta u = f \,, \tag{1.0.4}$$

where Δ is the Laplacian. A function u satisfying Eq. (1.0.4) is said to be a strong solution of Poisson's equation, as Eq. (1.0.4) is satisfied everywhere inside the domain of interest Ω. Considering a Dirichlet boundary condition of $u = 0$ on $\partial\Omega$, we multiply Eq. (1.0.4) by some test function v also satisfying the Dirichlet boundary condition, integrate over Ω, and use integration by parts to yield

$$-\int_\Omega \mathrm{d}\mathbf{x}\ \nabla u \cdot \nabla v = \int_\Omega \mathrm{d}\mathbf{x}\ f v \,, \tag{1.0.5}$$

where ∇ is the gradient operator. A function u satisfying Eq. (1.0.5) is said to be a weak solution of Poisson's equation as we satisfy Eq. (1.0.4) integral-wise over Ω. Weak solutions have lower continuity requirements than strong solutions, allowing for the use of simpler function spaces.

The second major part of FEM consists of the choice in function space of u. Traditionally one uses the Galerkin method in which we represent u, v, and f using a finite-dimensional space. This choice of basis functions allows for the evaluation of the coefficients in the basis function representation of u by solving a linear system of equations given by Eq. (1.0.5). A separate, variational approach is found by noting that the weak solution u of Poisson's equation can be obtained from the functional optimization of [3]

$$u(\mathbf{x}) = \arg\min_w \frac{1}{2}\int_\Omega \mathrm{d}\mathbf{x}\ |\nabla w|^2 + \int_\Omega \mathrm{d}\mathbf{x}\ f w \,. \tag{1.0.6}$$

Obtaining u through Eq. (1.0.6) is termed the Ritz method. Typically the function space consists of local basis functions on a discretization of the domain, but it is noted that any basis function, such as non-local functions or a neural network, could be used.

1.1 Biological Membranes

In the first part of this thesis, we will use techniques from statistical mechanics and the finite element method to model biological membranes. Biological membranes serve as the barrier between the insides and outsides of the cell and some organelles. Composed of lipids and proteins, we will be concerned primarily with protein-lipid organization inside the membrane itself. It was originally proposed that biological membranes consist of a liquid-disordered lipid membrane with randomly distributed proteins in the fluid mosaic model in 1972 [4], which has been continually refined to account for the realities of experimental observation. In one such refinement, the experimental observation that detergents selectively solubilized into liquid-disordered regions enriched in unsaturated lipids and avoided liquid-ordered regions enriched in saturated lipids and cholesterol [5] was codified in the proposal of biological membranes having nanoscopic liquid-ordered regions termed lipid rafts [6]. Though indirect experimental evidence has mounted for the existence of lipid rafts, they have yet to be directly observed with microscopy due to the small length and time scales they are proposed to exist on.

In this thesis, we extend prior work that developed a mechanism for lipid rafts from protein-lipid interactions. Termed the orderphobic effect [7], it was demonstrated that proteins with lengths matching that of the thermodynamically disfavored lipid phase can induce that phase in a membrane otherwise in the thermodynamically favored phase. Doing so creates interfaces around the protein domains which provide a free energy incentive for protein assembly due to line tension. While this mechanism leads to complete domain assembly and macrophase behavior, the orderphobic effect does not explain how lipid rafts remain nanoscopic. Additionally, the orderphobic effect was demonstrated in a coarse-grained molecular dynamics simulation of a single component lipid, with extensions to multicomponent systems using coarse-grained molecular dynamics being difficult due to the relatively large length scale of lipids rafts and equilibration of the compositional variables.

To bypass these issues, we extend the orderphobic effect to multicomponent lipid membranes using computational modeling on triangulated surfaces with the needed length scales. To do so we develop a computational model of a single component biological membrane in Chapter 2, where the finite element method is used to develop a model that accounts for the elastic behavior of lipid membranes out-of-plane, and Metropolis Monte Carlo moves is used to account for the fluid behavior in-plane. Chapter 3 refines the single component model into a multicomponent model accounting for the existence of liquid-disordered, liquid-ordered, and protein-enriched domains. Sampling techniques are used to probe the interactions between large protein domains in order to understand their interactions driven by compositional and elastic energetics, and we explore how the competition between these interactions regulates the domain sizes to yield microscopic phase separation.

1.2 Data-Driven Reaction Learning

In the second part of this thesis, we will focus on data-driven reaction learning. In many kinetic processes, there are free energy barriers that make the computational study of the transition pathway between reactant and product states difficult. The usual approach to ameliorate this issue is the use of collective variables, functions of $\mathbf{x}$ which give a low-dimensional representation of the system that tracks the dynamical progress of the reaction. The use of collective variables can allow for efficient evaluation of the reaction rate, but developing good collective variables is not a trivial process. One methodology to verify the quality of collective variables is the evaluation of the committor function, the probability that a trajectory starting from some $\mathbf{x}$ will enter the product state before the reactant state. Good collective variables will identify a transition state at which configurations have committor values dense around a probability of 0.5. One can consider the committor function to be the ultimate collective variable as it tracks the reaction progress exactly, but it is computationally expensive to evaluate through naive evaluation by counting the number of trajectories starting from a configuration that commits to the product state.

Recent work [8–10] has focused on an alternative approach to computing the committor function. In these schemes, one uses a theoretical framework known as transition path theory to establish a partial differential equation, the backward Kolmogorov equation, that the committor function is a solution of. The Ritz method is then used to obtain the parameters of a neural network representation of the committor function. In these prior works, importance sampling is used to evaluate the needed integrals to perform the optimization in the Ritz method. The needed integrals for optimization are also proportional to the reaction rate, allowing for its evaluation. However, even with importance sampling, the evaluation of the needed integrals is difficult and in most cases, one paradoxically needs a good collective variable to perform adequate sampling. Reference [10] bypassed this issue using umbrella sampling with respect to the neural network representation of the committor function, providing a collective variable-free method to sample the transition pathway. To do so requires judicious tuning of the umbrella sampling parameters in order to homogeneously sample the transition pathway as the committor function can vary rapidly along it, and such a process can be difficult and requires much tuning.

In this thesis, we improve upon the method in Ref. [10] by using supervised learning to increase the accuracy of the neural network and the finite-temperature string method to increase the accuracy of sampling. As the committor function can be obtained empirically through running trajectories starting from a known configuration, we obtain these estimates and use them as fitting data for the neural network representation of the committor function in addition to applying the Ritz method to the backward Kolmogorov equation. The finite-temperature string method, on the other hand, is a path-finding algorithm that obtains an approximate transition pathway that bypasses sampling issues inherent to umbrella sampling, and can be derived through transition path theory. We review transition-path theory and the finite-temperature string method in Chapter 4. Chapter 5 reviews the algorithm of Ref. [10]

along with our extensions. We test the algorithms on low-dimensional problems in which we are able to evaluate the errors in both sampling and the neural network exactly, giving rise to an error analysis in the sampling procedure that allows for the recovery of more accurate reaction rate estimates. We then extend the method to a high-dimensional molecular system in which a message-passing neural network is used to obtain both a collective-variable free representation of the committor function and estimates of the reaction rate to high accuracy.

Part A

Biological Membranes

Chapter 2

Membrane Modeling

2.1 Introduction to Biological Membranes

Biological membranes serve as boundaries in both the cell and many of the organelles inside the cell. They are composed of amphiphilic lipid molecules and proteins. While first described as being composed of proteins randomly embedded inside the lipid bilayer per the fluid mosaic model (see Fig. 2.1), this picture has been continuously refined with additional factors such as lipid-lipid and lipid-protein interactions that drive membrane organization important in many molecular phenomena [4, 11, 12]. These molecular phenomena include processes such as cellular signaling [13, 14] and cellular trafficking [15–19], from which one hopes that modeling their biophysical behavior can advance their understanding.

While biological membranes are composed of numerous types of lipid and protein species, membrane systems have universal physical properties that can be utilized in their modeling. The first of which is that biological membranes are elastic out-of-plane, producing resistance to membrane deformations that produce curvature. Secondly, membranes are fluid in-plane, leading to no resistance to shear and the free diffusion of individual molecules in the membrane. These properties will be used to model membranes on length and time scales relevant to larger-scale biological processes of interest.

Membrane modeling, at the time this thesis is written, is done typically at small length scales corresponding to small membrane patches ($O(10\text{ nm})$) or at large length scales matching that of cells ($O(10\ \mu\text{m})$). At small length scales, individual lipid and protein molecules are represented using molecular models that are simulated using molecular dynamics. While capable of capturing local molecular features at either the atomistic [20–22] or coarse-grained level [23–25], these methods are currently incapable of scaling to the large length and time scales needed to model cellular behavior. On the other hand, continuum models can represent the membrane as a two-dimensional surface and model its dynamics with phenomenological models over large length and time scales [26–28]. However, these models typically do not include thermal fluctuations which drive many molecular phenomena such as composition dynamics.

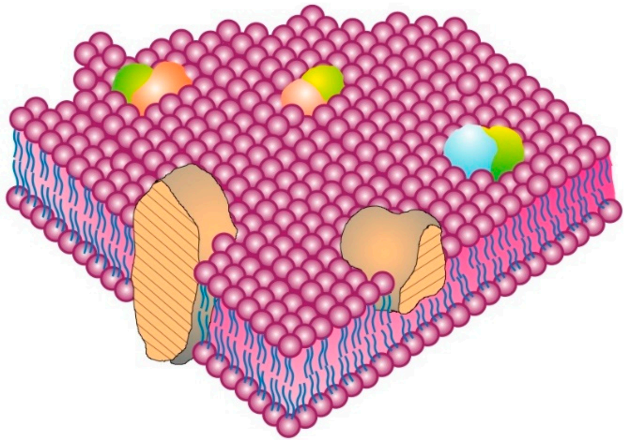


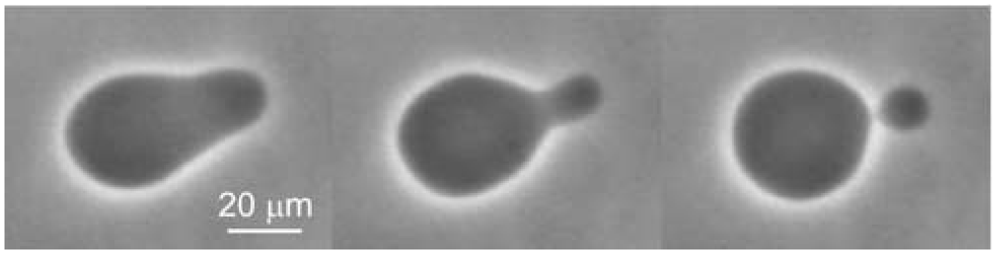
Figure 2.1: The fluid mosaic model of a biological membrane, in which integral proteins are randomly distributed in a disordered-liquid bilayer formed by lipids. Adapted from G. L. Nicolson and G. Ferreira de Mattos, "A brief introduction to some aspects of the fluid–mosaic model of cell membrane structure and its importance in membrane lipid replacement", Membranes **11**, 10.3390/membranes11120947 (2021) licensed under CC BY 4.0 http://creativecommons.org/licenses/by/4.0/.

In Chapters 2 and 3, we will model biological membranes on length and time scales beyond what can be done in standard molecular dynamics simulations and including thermal fluctuations missing in standard continuum models. To do so, we will introduce a phenomenological model of biological membranes that captures their in-plane fluidity and out-of-plane elasticity in Section 2.2. This model will then be discretized in various forms on a triangulated surface for use in Monte Carlo simulations in Section 2.3. The discretizations will then be tested in Section 2.4 on two model problems, with the first testing the convergence properties of the discretization on spheres and cylinders, and the second looking at the relation between the height fluctuations of a planar periodic surface and the inputted membrane parameters.

2.2 Modeling Membranes on Long Length and Time Scales

We are interested in modeling processes involving biological membranes that occur on relatively long length and time scales. Lipid molecules and proteins can have important molecular details essential to modeling some processes, but we consider large-scale processes in which we utilize the separation of length scales between the thickness of the lipid bilayer, around 5 nm, and the surface the bilayer spans, which can be micrometers. This disparity in length scales allows for the modeling of membranes as a two-dimensional surface in three-dimensional space. With this perspective, we model the energetics of this surface through its surface invariants such as area and curvature, which are used in a way that respects the symmetries and fundamental properties of the system as one does in a Landau expansion for a free energy [29].

(a)



(b)

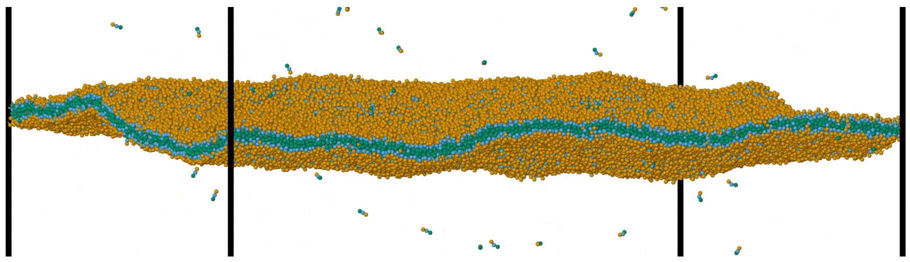


Figure 2.2: (a) Budding of a vesicle induced by insertion of amphiphilic molecules into the external leaflet of a vesicle over the course of about 5 s, demonstrating the in-plane fluid behavior of lipid membranes. Adapted from R. Dimova et al., "A practical guide to giant vesicles. probing the membrane nanoregime via optical microscopy", Journal of Physics: Condensed Matter **18**, S1151–S1176 (2006), reproduced with permission of IOP Publishing, Ltd. in the format "Republish in a thesis/dissertation" via Copyright Clearance Center. (b) Simulation snapshot of the Cooke-Kremer-Deserno model, a coarse-grained lipid model that captures a lipid bilayer's out-of-plane elastic behavior [23]. Rendered using Ovito [31].

With this model, we will capture the fundamental membrane properties of in-plane fluidity and out-of-plane elasticity [27, 28]. The fluidity allows for molecular rearrangement in the membrane, both at the scale of individual molecules through free diffusion and at the scale of large-scale rearrangements such as budding (Fig. 2.2a). The elasticity, on the other hand, is the primary source of the energetics with the membrane resisting deformations away from its preferred curvature out-of-plane (Fig. 2.2b). We first focus on capturing the elastic energetics and account for the fluidity in the subsequent numerical discretization.

The first elastic energy term is from the surface area A. Biological membranes, being composed of amphiphilic materials, are capable of having a surface tension conjugate to the surface area that can range from negligible values to values capable of rupturing the membrane [32]. It would seem sufficient to use a potential energy of the form

$$V_{\text{tension}} = \gamma A\,, \tag{2.2.1}$$

with γ being the surface tension. As previously mentioned, Eq. (2.2.1) can be interpreted with γ serving as a conjugate variable to A. Additionally one can interpret γ as a Lagrange multiplier that sets the area.

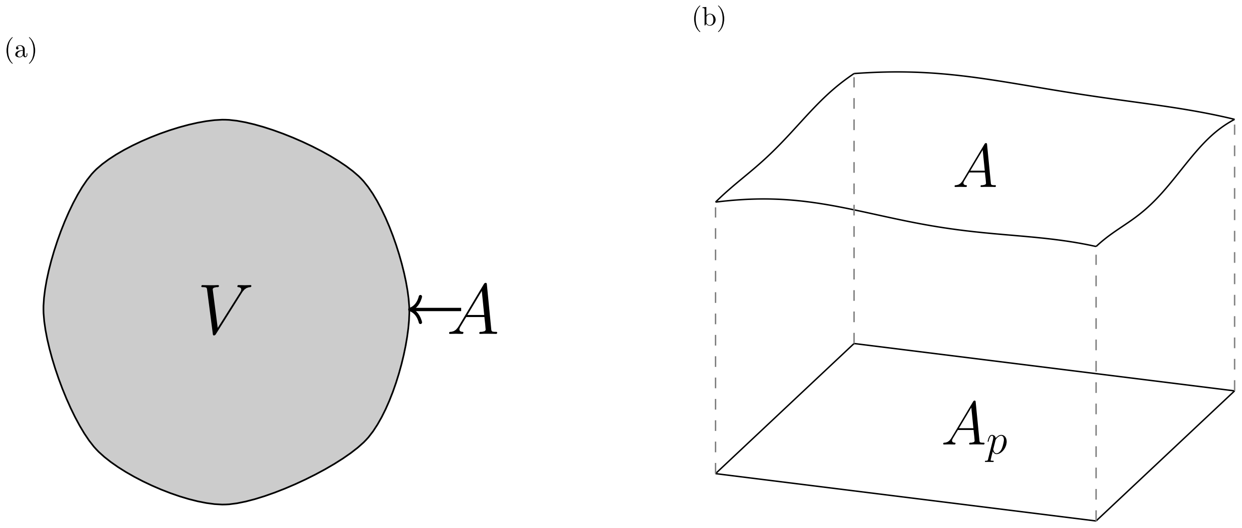


Figure 2.3: (a) Example of a spherical system with area A and volume V. (b) Example of a planar periodic system with area A and projected area A_p.

In the case of closed membrane surfaces and planar periodic membrane surfaces, there are other surface invariants of the same order as the surface area. We first examine the behavior of a lipid membrane enclosing a 3D surface, as shown in Fig. 2.3a. In this system, one can consider the volume enclosed by the surface, V. The energy of the first order surface invariants is then [33]

$$V_{\text{tension}} = \gamma A - \Delta p V\,, \tag{2.2.2}$$

where Δp is the pressure difference between the inside and outside of the surface. The pressure difference Δp, similar to the surface tension γ, serves as a thermodynamic variable conjugate to the enclosed volume. In the case of a planar periodic surface, as shown in Fig. 2.3b, an equivalent term to $\Delta p V$ is needed of the form [33]

$$V_{\text{tension}} = \gamma A - \tau A_p\,, \tag{2.2.3}$$

where τ is the frame tension and A_p is the projected area. Though γ is separate from τ and p in the planar periodic and closed cases, respectively, they are related to each other by thermodynamic arguments [33]. In the case of a closed surface, this is through Laplace's law

$$\Delta p = 2\gamma/R_c \,, \tag{2.2.4}$$

where R_c is the effective radius of curvature given by $3V/A$, and in the planar periodic case γ and τ are related by

$$\tau = \frac{A}{A_p}\gamma \,. \tag{2.2.5}$$

For a taut membrane $A \approx A_p$ to a good approximation, which will be used throughout this thesis.

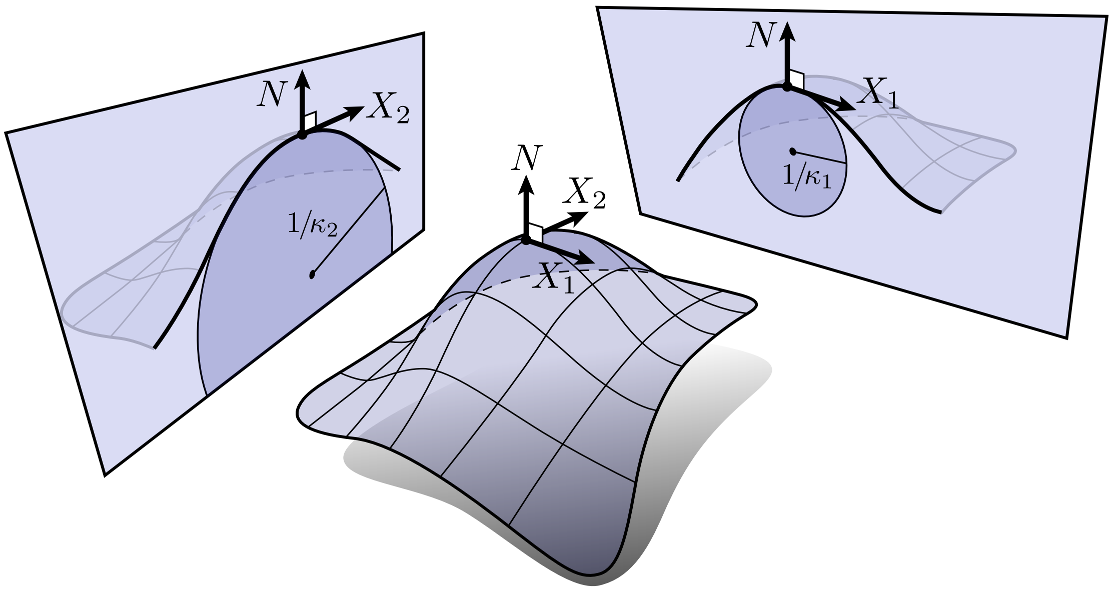


Figure 2.4: The principle directions and normal associated at a point. The curvature associated with principle direction X_1 corresponds to the minimum normal curvature κ_1 while the curvature associated with principle direction X_2 corresponds to the maximum normal curvature κ_2. Adapted from Keenan Crane's "Discrete Differential Geometry" [34].

The next contribution to the energy is composed of quantities involving the curvature of the surface. As shown in Fig. 2.4 we associate to each point of the surface two principle curvatures κ_1 and κ_2 that correspond to principle directions X_1 and X_2, respectively. The principle directions correspond to the directions of minimal and maximal curvature of the surface at that point. As the individual curvatures κ_1 and κ_2 are not invariant with respect to rotations of the system, we compute quadratic quantities from κ_1 and κ_2 that are. One such quantity is the square of the mean curvature given by $H = \frac{1}{2}(\kappa_1 + \kappa_2)$, which can be shifted by a spontaneous curvature term C to induce a preferred curvature. The other quantity is the product of the principle curvatures yielding the Gaussian curvature $K = \kappa_1\kappa_2$. The use of these quantities yields the Helfrich energy [26] which has the form

$$V_{\mathrm{b}} = \int \mathrm{d}a \ \left[k_{\mathrm{b}}(H-C)^2 + k_{\mathrm{g}}K\right] \,, \tag{2.2.6}$$

where k_{b} is the bending rigidity, k_{g} is the Gaussian modulus, and $\int \mathrm{d}a$ indicates integration over the surface. It is noted that

$$\int \mathrm{d}a = A \,. \tag{2.2.7}$$

For fixed surface topology the integral of K in Eq. (2.2.6) is constant by the Gauss-Bonnet theorem [35, 36]. As we only consider surfaces with constant topology, the energy associated with Gaussian curvature is fixed and hence we utilize a simpler potential energy for the rest of this work of the form

$$V_{\mathrm{b}} = \int \mathrm{d}a \ k_{\mathrm{b}}(H-C)^2 \,. \tag{2.2.8}$$

2.3 Discretizations of the Helfrich Model

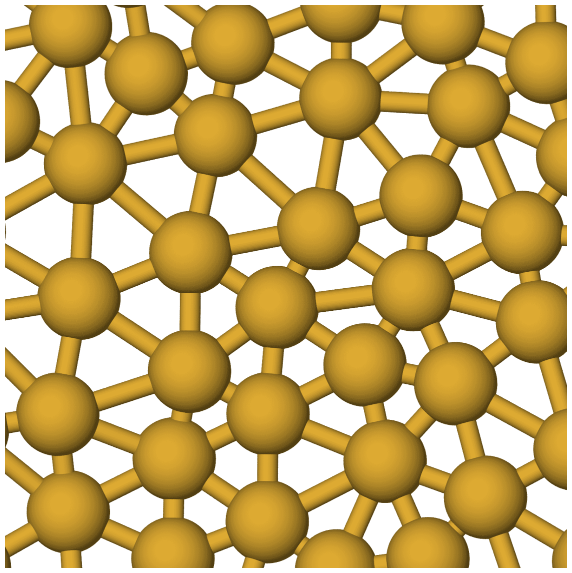


Figure 2.5: Example of a triangulated surface. Note the varying number of edges per vertex.

Though the Helfrich model is amendable to analytical methods, we will be looking at problems where computational methods are needed to compute quantities of interest. To facilitate this, the Helfrich model is discretized on a surface triangulation (Fig. 2.5). In this section, we introduce the methodology needed to evolve the membrane surface in a fluid manner along with appropriate discretizations of the Helfrich model to include elasticity Eq. (2.2.8).

2.3.1 General Scheme on a Dynamically Triangulated Surface Using Metropolis Monte Carlo

A surface, be it planar, cylindrical, spherical, etc., is discretized into a triangulated mesh composed of vertices i, edges ij, and triangles ijk. An example of a vertex and its associated edges and triangles is shown in Fig. 2.6a. Under most of these schemes, we discretize Eqs. (2.2.3) and (2.2.8) for use on the triangulated surface to yield

$$V_{\text{mem}} = \sum_i \left(k_{\text{b}} \left(H_i - C_i \right)^2 + \gamma \right) \Delta a_i - \tau A_p \,, \tag{2.3.1}$$

where H_i, C_i, and Δa_i are the discrete mean curvature, spontaneous curvature, and discrete area at a vertex, respectively. Vertices interact as hard spheres of diameter σ with all other vertices to enforce self-avoidance and the edges have a maximum length of $\sqrt{2.8}\,\sigma$ to enforce stability of the discretizations as will be discussed later [37, 38].

In order to evolve the triangulated surface and sample quantities of interest, we use Metropolis Monte Carlo [39–41]. Metropolis Monte Carlo is a scheme in which a variety of "moves" are used to evolve the system in a way that samples according to the system's underlying probability distribution. This is done by generating random moves on the system that perform slight modifications on it, with the moves either being accepted or rejected according to some criterion that is related to the system's probability distribution. Denoting the system's state with $\mathbf{\Gamma}$, containing the positions of the vertices $\mathbf{x}_i$ and the triangulation $\mathcal{T}$, and the probability distribution of the system as $\rho\left(\mathbf{\Gamma}\right)$, Metropolis Monte Carlo gives a mean of evaluating the average of some quantity $\mathcal{A}$ with respect to $\rho\left(\mathbf{\Gamma}\right)$ by a finite sum computed from samples obtained in simulation per

$$\langle \mathcal{A} \rangle = \int \mathrm{d}\mathbf{\Gamma}\; \rho\left(\mathbf{\Gamma}\right) \mathcal{A}\left(\mathbf{\Gamma}\right) \approx \frac{1}{N} \sum_{i=1}^{N} \mathcal{A}\left(\mathbf{\Gamma}_i\right) \,, \tag{2.3.2}$$

where $\mathbf{\Gamma}_i$ are states generated through the Monte Carlo moves. This is done as $\mathbf{\Gamma}$ is typically a high-dimensional space in which evaluation of the integral in Eq. (2.3.2) is not numerically feasible using standard quadrature. Metropolis Monte Carlo is referred to as a form of importance sampling with the moves allowing the sampling of $\mathbf{\Gamma} \sim \rho\left(\mathbf{\Gamma}\right)$ and hence observing higher probability states more frequently that make larger contributions to the average in Eq. (2.3.2).

With Metropolis Monte Carlo we assume the process is in equilibrium, ergodic, *i.e.* given enough Monte Carlo moves all possible states will be explored, and the Monte Carlo moves generate a Markov chain, *i.e.* the moves only depend on the state obtained by the last move performed. We now demonstrate the construction of Monte Carlo moves and their generation and acceptance that is adapted from standard references [39, 41]. We consider discrete states $\mathbf{\Gamma}_\nu$ and Monte Carlo moves that generate transitions between two states ν and μ. Under the assumptions of equilibrium behavior and ergodicity, the transition probability between states

ν and μ denoted as $R_{\nu\mu}$ and equilibrium probabilities are related by

$$\sum_{\nu} R_{\nu\mu}\rho_{\nu} = \rho_{\mu} \quad \forall\mu\,, \tag{2.3.3}$$

where $\rho_{\nu} = \rho(\mathbf{\Gamma}_{\nu})$. There are many possible choices of $R_{\nu\mu}$ that satisfy Eq. (2.3.3), but we choose a relation between values of $R_{\nu\mu}$ that implies that system is microscopically reversible. This is done as equilibrium systems do not have fluxes between states that drive the system. Microscopic reversibility is obtained by setting the fluxes between states to be equal to yield

$$R_{\nu\mu}\rho_{\nu} = R_{\mu\nu}\rho_{\mu}\,. \tag{2.3.4}$$

Equation (2.3.4) satisfies Eq. (2.3.3) per

$$\sum_{\nu} R_{\nu\mu}\rho_{\nu} = \sum_{\nu} R_{\mu\nu}\rho_{\mu} = \rho_{\mu}\,, \tag{2.3.5}$$

with $\sum_{\nu} R_{\mu\nu} = 1$ by definition of a transition probability.

We now specify the acceptance condition behind the Monte Carlo moves. To do so, the transition probability is decomposed into two different components. This decomposition takes the form

$$R_{\nu\mu} = T_{\nu\mu}A_{\nu\mu}\,, \tag{2.3.6}$$

where $T_{\nu\mu}$ is the probability of generating a move from ν to μ and $A_{\nu\mu}$ is the probability of accepting a move from state ν to μ. The microscopic reversibility condition of Eq. (2.3.4) is rewritten as

$$\frac{A_{\nu\mu}}{A_{\mu\nu}} = \frac{T_{\mu\nu}\rho_{\mu}}{T_{\nu\mu}\rho_{\nu}}\,. \tag{2.3.7}$$

To proceed, note that the probability distributions typically are in the form of the Boltzmann distribution in which [1]

$$\rho_{\nu} = \frac{\exp(-\beta V_{\nu})}{Q}\,, \tag{2.3.8}$$

$$Q = \int \mathrm{d}\mathbf{\Gamma} \exp(-\beta V(\mathbf{\Gamma}))\,, \tag{2.3.9}$$

where $\beta = \frac{1}{k_{\mathrm{B}}T}$, with k_{B} being the Boltzmann constant, T is the temperature, and Q is the (configurational) partition function. Given this form of probability, Eq. (2.3.7) becomes

$$\frac{A_{\nu\mu}}{A_{\mu\nu}} = \frac{T_{\mu\nu}}{T_{\nu\mu}} \exp(-\beta(V_{\mu} - V_{\nu}))\,. \tag{2.3.10}$$

The form of the acceptance ratios is now constructed. As we wish to sample states that are more statistically significant, moves are automatically accepted from μ to ν if the energy is lowered, *i.e.* $V_{\mu} - V_{\nu} < 0$, and to satisfy Eq. (2.3.10) we accept it with probability

$\exp(-\beta(V_\mu - V_\nu))$ if $V_\mu - V_\nu > 0$. On a computer, the probabilities are generated using a pseudorandom number generator as a number between $[0,1)$ to decide acceptance. As $\exp(-\beta(V_\mu - V_\nu)) > 1$ if $V_\mu - V_\nu < 0$, the acceptance probability is given by

$$A_{\nu\mu} = \min[1, \frac{T_{\mu\nu}}{T_{\nu\mu}} \exp(-\beta(V_\mu - V_\nu))] \,. \tag{2.3.11}$$

The criterion given by Eq. (2.3.11) is known as the Metropolis criterion.

We use three Monte Carlo moves for the evolution of $\mathbf{\Gamma}$, which modify either the positions $\mathbf{x}$ or the triangulation $\mathcal{T}$. The Monte Carlo moves incorporate the fluidity of the membrane into this model, allowing for the free diffusion of vertices through dynamically triangulating the surface [37, 42–44]. The Monte Carlo moves are as follows

1. *Displacement*: a vertex i is selected at random, and its position is displaced randomly within a cube with sides spanning from $-\delta$ to δ in all directions (Fig. 2.6b). As the vertex is selected at random and displaced by a random amount, the transition probabilities are equal to each other. The acceptance probability factor in Eq. (2.3.11) is then $\exp(-\beta\,(V_\mu - V_\nu))$. This move leads to vertices diffusing in the current triangulation.

2. *Edge Flip*: a vertex i is selected at random, from which an edge ij is randomly selected out of all the edges vertex i is part of. As the surfaces we consider have no free edges, each edge is part of two triangles. Denoting these triangles as ijk and ijl, the edge flip move attempts to "flip" the edge ij to edge kl (Fig. 2.6c). The acceptance probability factor is evaluated as follows. Selecting the edge ij occurs either through choosing vertex i or j and then selecting edge ij out of all possible edges i or j are part of. The same reasoning holds for selecting edge kl in the new triangulation generated by the edge flip move, in which the number of edges vertices k and l have is one greater than in the old triangulation. Denoting $\mathcal{N}_i$ as the number of edges vertex i is part of, the transition probability factor is

$$\frac{T_{\mu\nu}}{T_{\nu\mu}} = \frac{\frac{1}{N}\left((\mathcal{N}_k+1)^{-1} + (\mathcal{N}_l+1)^{-1}\right)}{\frac{1}{N}\left((\mathcal{N}_i)^{-1} + (\mathcal{N}_j)^{-1}\right)} = \frac{(\mathcal{N}_k+1)^{-1} + (\mathcal{N}_l+1)^{-1}}{(\mathcal{N}_i)^{-1} + (\mathcal{N}_j)^{-1}} \,. \tag{2.3.12}$$

   This move allows vertices to diffuse throughout the entire system by modifying the triangulation, and also allows for large deformations that change the local connectivity such as budding.

   An additional constraint is used with this move to maintain the topology of the system [37, 45]. The minimum number of triangles a vertex is a part of is set to be three, and the maximum number of triangles a vertex is a part of is set to be nine. Violations of these constraints lead to automatic rejection of the proposed move.

3. *Box Resizing*: in planar periodic systems we allow the projected area to change. To do so we use a Monte Carlo move that resizes the box size in the x and y directions,

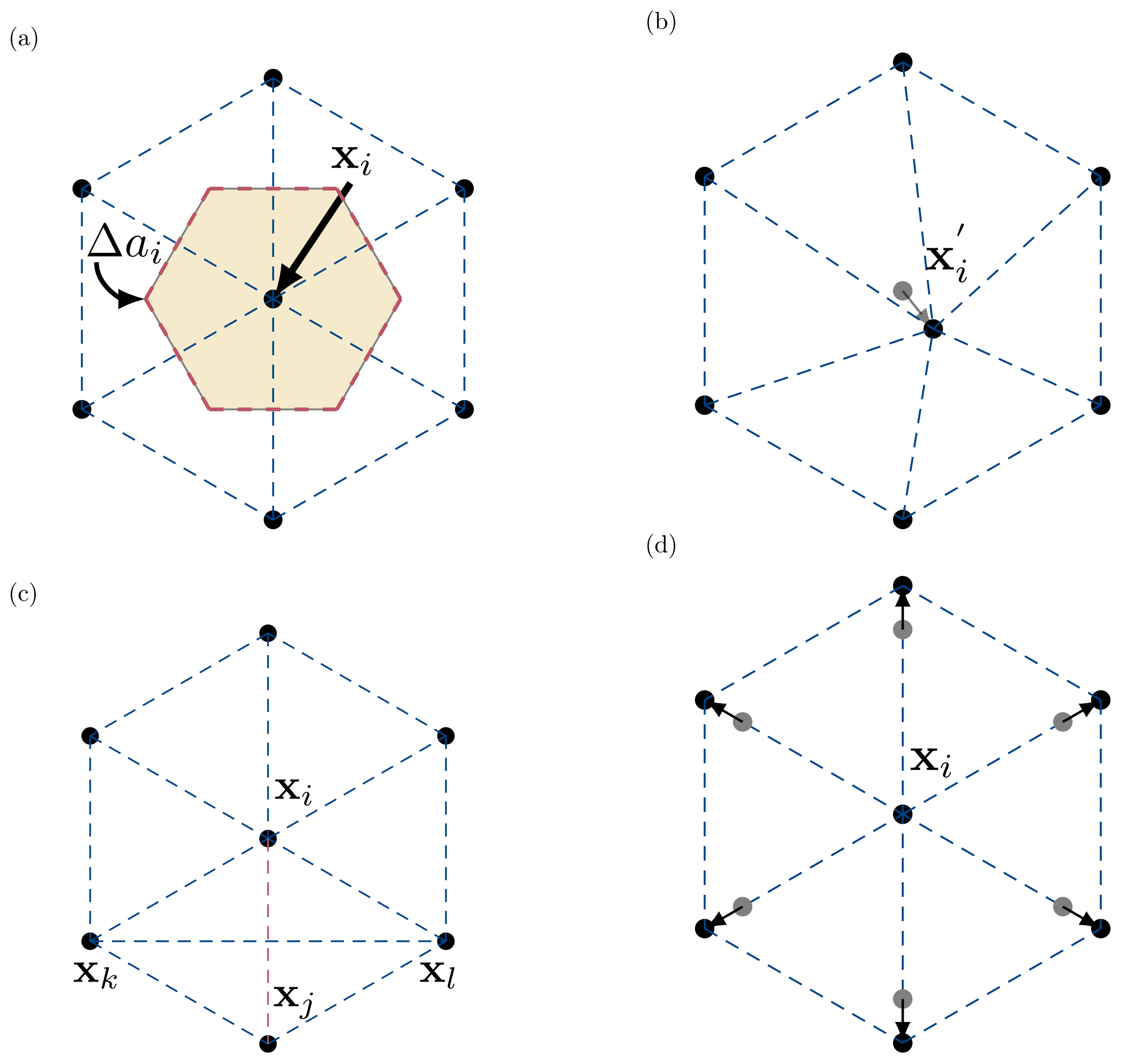


Figure 2.6: A vertex in a triangulated system with its neighbors demonstrating the possible Monte Carlo moves used to maintain the fluidity of the membrane. (a) A vertex i with position $\mathbf{x}_i$, neighboring vertices, and vertex area Δa_i. (b) Displacement of $\mathbf{x}_i$ to generate a trial position $\mathbf{x}_i^{'}$. (c) Edge flip performed on an edge containing vertex i to yield a new trial triangulation, with the edge in red being flipped to the new edge kl. (d) Resizing of the plane to generate a new trial area.

given by L_x and L_y respectively, which is related to the projected area by $A_p = L_x L_y$. Letting $\mathbf{\Gamma}$ contain all degrees of freedom aside from L_x and L_y, the partition function at T, number of vertices N, γ, and τ is then evaluated by

$$Q(T, N, \gamma, \tau) = \int_0^\infty \int \mathrm{d}A_p \, \mathrm{d}\mathbf{\Gamma} \ \exp(-\beta V_{\text{mem}}) \tag{2.3.13}$$

$$= \int_0^\infty \mathrm{d}A_p \ Q(T, N, A_p, \gamma, \tau)\,, \tag{2.3.14}$$

where $Q(T, N, A_p, \gamma, \tau)$ is the partition function at fixed projected area.

To evaluate the transition probabilities and for computational convenience we set $L = L_x = L_y$ and use L as a scaling factor for the x and y directions [39, 46–48]. This scaling factor leads to the positions in the x, y, and z directions being stored as

$$s_{x,i} = \frac{x_i}{L}\,, \tag{2.3.15}$$

$$s_{y,i} = \frac{y_i}{L}\,, \tag{2.3.16}$$

$$s_{z,i} = z_i\,. \tag{2.3.17}$$

The differential volume element, $\mathrm{d}\mathbf{x}$, can then be written in terms of $\mathrm{d}\mathbf{s}$ as

$$\mathrm{d}\mathbf{x} = A_p^N \,\mathrm{d}\mathbf{s}\,. \tag{2.3.18}$$

The box resizing move modifies L by adding to it random number between $-\delta L_{\max}$ to $\delta L_{\max}$. While the change in L can be reversed with a move having an equal and opposite displacement in L, the resizing move also maps $\mathrm{d}\mathbf{x}$ to an element with a different volume. Intuitively larger values of L generate $\mathrm{d}\mathbf{x}$ with more possible states. The ratio of the transition probabilities is then evaluated as the ratio of the volume elements

$$\frac{T_{\mu\nu}}{T_{\nu\mu}} = \frac{\mathrm{d}\mathbf{x}_\mu}{\mathrm{d}\mathbf{x}_\nu} = \frac{V_\mu^N}{V_\nu^N} = \left(\frac{L_\mu}{L_\nu}\right)^{2N}\,. \tag{2.3.19}$$

The acceptance probability is written as

$$A_{\nu\mu} = \min[1, \frac{T_{\mu\nu}}{T_{\nu\mu}} \exp(-\beta(V_\mu - V_\nu)] \tag{2.3.20}$$

$$= \min[1, \left(\frac{L_\mu}{L_\nu}\right)^{2N} \exp\left[-\beta\left(V_\mu - V_\nu\right)\right]]\,. \tag{2.3.21}$$

The box resizing move has the effect of allowing vertices to flow in the direction of large deformations. For example, in the case of bud formation vertices are able to move inwards by reducing the projected area while keeping the surface area constant. This move can be extended to closed systems by storing z_i in the same manner as x_i and y_i, where the factor of $\left(\frac{L_\mu}{L_\nu}\right)^{2N}$ is now $\left(\frac{L_\mu}{L_\nu}\right)^{3N}$ instead.

In a typical Monte Carlo simulation, the displacement and edge flip moves are attempted more frequently than the box resizing move due to the former moves changing only local environments while the box resizing is a global move requiring a full revaluation of the energy. We perform serial simulations with the frequency of a box resizing move being N_{vertices}^{-1} and the frequency of the displacement and edge flip moves being $\frac{1}{2}\left(1 - N_{\text{vertices}}^{-1}\right)$ each.

2.3.1.1 Parallel Monte Carlo

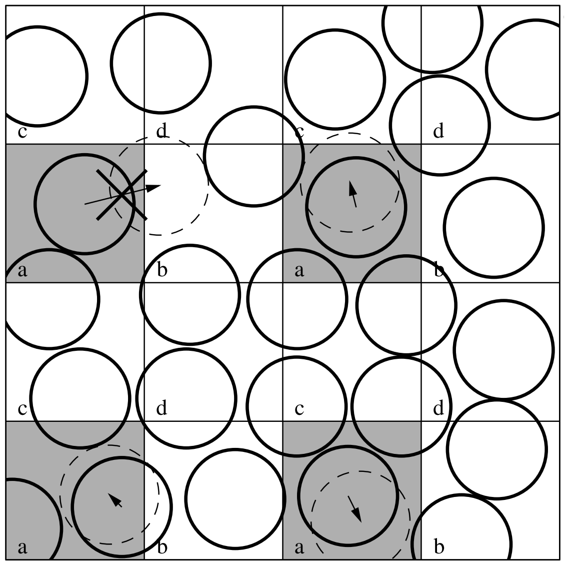


Figure 2.7: The checkerboard scheme used in the parallelization of Monte Carlo for a two-dimensional system. The domain is divided into four subdomains given by the abcd labels, such that when Monte Carlo moves are applied to subdomains of one type the inactive subdomains (in white) are large enough to make the active subdomains (in grey) independent of each other. Vertices cannot leave the subdomains they started in (see top left for an example) during the parallel move cycle. Reprinted from "Massively parallel Monte Carlo for many-particle simulations on GPUs", **254**, J. A. Anderson *et al.*, 27–38, Copyright of Elsevier (2013), with permission from Elsevier [49]

To enhance the speed of sampling, a parallelization scheme is used to decompose the simulation domain into regions that can be modified in parallel with Monte Carlo moves. This scheme, termed checkerboard decomposition [49], does so by decomposing the simulation domain into a set of subdomains (Fig. 2.7). The subdomains are sized such that a displacement or edge flip move in subdomains of one type does not affect the energies of other subdomains of the same type. To allow for the diffusion of vertices, the subdomain centers generated in each checkerboard decomposition are chosen randomly. In planar periodic systems, four sets of subdomains spanning the xy plane are used, while in spherical and cylindrical systems eight sets of subdomains are used.

In the parallel Monte Carlo simulation, we choose with equal frequency either the box-resizing move or a parallel checkerboard move. Note that the box-resizing move is trivially parallelizable with the needed energy evaluation. The parallel checkerboard move proceeds by generating the subdomains and the modification order of the subdomains. For each set of subdomains, Metropolis Monte Carlo is performed in parallel by selecting vertices at random in the local subdomains and then selecting either a displacement or edge flip move with equal frequency. In order to satisfy microscopic reversibility in a subdomain, displacement moves are rejected if a vertex is moved out of the subdomain it originated in, as the vertex would not be able to be moved back into the subdomain during the remaining moves on that type of subdomain, and edge flip moves are rejected if the new edge contains vertices outside of the current subdomain. This is done for some amount of Metropolis Monte Carlo moves, with an average of 3 Monte Carlo moves per vertex in a subdomain. The Saru pseudorandom number generator is used to generate uncorrelated random number streams in parallel during the parallel checkerboard move [50, 51]. While the parallel checkerboard move does not satisfy microscopic reversibility at the level of every single move inside of one parallel checkerboard move, as a parallel checkerboard move could reverse the previous move exactly, *i.e.* by choosing the order of subdomain types, move types, and vertices in the reverse order of the previous move, microscopic reversibility is achieved at the level of full parallel checkerboard moves.

2.3.2 Kantor-Nelson Model

We now consider different discretizations of Eq. (2.3.1). The first model we consider is that of Kantor and Nelson, for which the bending energy is [52, 53]

$$V_{\mathrm{b}} = \frac{1}{2} k_{\mathrm{b}} \sum_{\langle\alpha,\beta\rangle} |\mathbf{n}_\alpha - \mathbf{n}_\beta|^2 = k \sum_{\langle\alpha,\beta\rangle} (1 - \mathbf{n}_\alpha \cdot \mathbf{n}_\beta) \ , \tag{2.3.22}$$

where $\langle\alpha,\beta\rangle$ indicates a sum over all neighboring triangles and $\mathbf{n}_\alpha$ is the normal of triangle α. Note $\mathbf{n}_\alpha \cdot \mathbf{n}_\beta = \cos(\theta_{\mathrm{e}})$, where θ_{e} is the dihedral angle between the two triangles (see Fig. 2.10(right)). The form involving the dihedral angle can incorporate spontaneous curvature by use of a preferred angle term θ_0 that yields [54]

$$\sum_i k_{\mathrm{c}} \left[1 - \cos(\theta_i - \theta_0)\right] \ . \tag{2.3.23}$$

This model is justified informally by the normal terms penalizing deviation from a flat plane, and formally by $\mathbf{n}_\alpha - \mathbf{n}_\beta$ being proportional to the gradient of the normal vector field which can be shown to be equivalent to $H^2 - 2K$ [53].

The primary issue of this model is the dependence of Eq. (2.3.22) on the shape of the modeled triangulated surface. It has been shown that a factor of $\frac{1}{\sqrt{3}}$ for a triangulated sphere and $\frac{\sqrt{3}}{2}$ for a triangulated cylinder is needed to match analytical values [38, 53]. Additionally, the model has a dependence on the Gaussian curvature of the surface [53], which will be shown to lead to deviations between the analytical Helfrich bending energy and that obtained from Eq. (2.3.22).

2.3.3 Gompper-Kroll Model

To rectify the shape dependence of Eq. (2.3.22), we evaluate Eq. (2.3.1) using linear finite elements on the triangulated surface. We begin with the relation [36]

$$\Delta_S \mathbf{x} = -2H\mathbf{n}\,, \tag{2.3.24}$$

where Δ_S is the surface Laplacian. If one can evaluate $\Delta_S \mathbf{x}$, the mean curvature can then be evaluated per

$$H = -\frac{1}{2}\Delta_S \mathbf{x} \cdot \mathbf{n}\,. \tag{2.3.25}$$

We evaluate $\Delta_S \mathbf{x}$ on a triangulated surface using techniques from linear finite elements, and use this framework to evaluate H_i and Δa_i in Eq. (2.3.1). This model was first proposed by Gompper and Kroll [38], though we note in its original context H_i and Δa_i were obtained using different techniques from random geometry theory [55]. The following proof is adapted from various references in the discrete differential geometry community [34, 56], along with ideas from the finite element community [57, 58].

To begin, we note that if the mapping for the surface $\mathbf{x}$ is sufficiently differentiable $\Delta_S \mathbf{x}$ can be obtained directly. However, such a surface is computationally difficult due to the number of degrees of freedom required for this differentiability [57]. We instead demonstrate how the value of the surface Laplacian at a vertex can be obtained using a technique known as projection. This is done by evaluating some linear function $\mathbf{g}$ that is set equal to $\Delta_S \mathbf{x}$ at all points such that

$$\Delta_S \mathbf{x} = \mathbf{g}\,. \tag{2.3.26}$$

We proceed by multiplying with an admissible test function $\mathbf{v}$ and integrate over the surface to yield

$$\int \mathrm{d}a\ \Delta_S \mathbf{x} \cdot \mathbf{v} = \int \mathrm{d}a\ \mathbf{g} \cdot \mathbf{v}\,. \tag{2.3.27}$$

As $\mathbf{v}$ can take any form, including one where it is nonzero for only one spatial direction, we use a simplified single-direction version of Eq. (2.3.27)

$$\int \mathrm{d}a\ \Delta_S x v = \int \mathrm{d}a\ g v\,, \tag{2.3.28}$$

and generalize to all directions afterward. Integrating Eq. (2.3.28) by parts, approximating the surface Laplacian with the regular Laplacian, and noting that the boundaries are either periodic or closed yields

$$-\int \mathrm{d}a\ \nabla x \cdot \nabla v = \int \mathrm{d}a\ g v\,. \tag{2.3.29}$$

Equation (2.3.29) is known as the weak formulation, in which we will seek a function g that satisfies the integral equation of Eq. (2.3.29) instead of the differential equation per the strong formulation of Eq. (2.3.26). This methodology in conjunction with setting x, g, and v to a function space is known as projection in which the obtained g is the orthogonal projection of $\Delta_S x$ onto the function space g is on [58].

We now represent x, g, and v with continuous piecewise linear basis functions to yield a computational framework to evaluate g numerically. Note that in this function space the surface Laplacian is equivalent to the Laplacian [59]. The basis functions $\mathbf{h}_i$ are constructed such that they are equal to 1 at vertex i and go to zero on the edges opposite vertex i. Given the value of x and g at the vertices are x_i and g_i, respectively, the linear approximation of x and g has the form

$$x = \sum_i x_i h_i(\mathbf{x})\,, \tag{2.3.30}$$

$$g = \sum_i g_i h_i(\mathbf{x})\,. \tag{2.3.31}$$

The function v can be written as any linear combination of basis functions $h_j(\mathbf{x})$, and for simplicity we set $v = h_i(\mathbf{x})$ for some index i. For the integral of gv in Eq. (2.3.29), this yields the linear system

$$\int \mathrm{d}a\ gv = \int \mathrm{d}a \left(\sum_j g_j h_j(\mathbf{x})\right) h_i(\mathbf{x}) = \sum_j g_j \int \mathrm{d}a\ h_i(\mathbf{x}) h_j(\mathbf{x})\,, \tag{2.3.32}$$

and for the integral of $\nabla x \cdot \nabla v$ in Eq. (2.3.29), this yields the linear system

$$\int \mathrm{d}a\ \nabla x \cdot \nabla v = \int \mathrm{d}a \left(\sum_j x_j \nabla h_j(\mathbf{x})\right) \cdot \nabla h_i(\mathbf{x}) = \sum_j x_j \int \mathrm{d}a\ \nabla h_i(\mathbf{x}) \cdot \nabla h_j(\mathbf{x})\,. \tag{2.3.33}$$

Denoting the integrals of the basis functions in Eqs. (2.3.32) and (2.3.33) as

$$A_{ij} = \int \mathrm{d}a\ \nabla h_i(\mathbf{x}) \cdot \nabla h_j(\mathbf{x})\,, \tag{2.3.34}$$

$$B_{ij} = \int \mathrm{d}a\ h_i(\mathbf{x}) h_j(\mathbf{x})\,, \tag{2.3.35}$$

one arrives at the linear equation

$$-\mathbf{A}\hat{\mathbf{x}} = \mathbf{B}\hat{\mathbf{g}}\,, \tag{2.3.36}$$

where $\hat{\mathbf{x}}$ and $\hat{\mathbf{g}}$ are vectors containing the coefficients x_i and g_i, respectively. Obtaining $\hat{\mathbf{g}}$ yields the value of the surface Laplacian at each vertex and hence the mean curvature at a vertex.

We now obtain A_{ij} through the properties of linear basis functions and trigonometry. We focus on a triangle lmn (Fig. 2.8a), and will establish general properties that hold for the

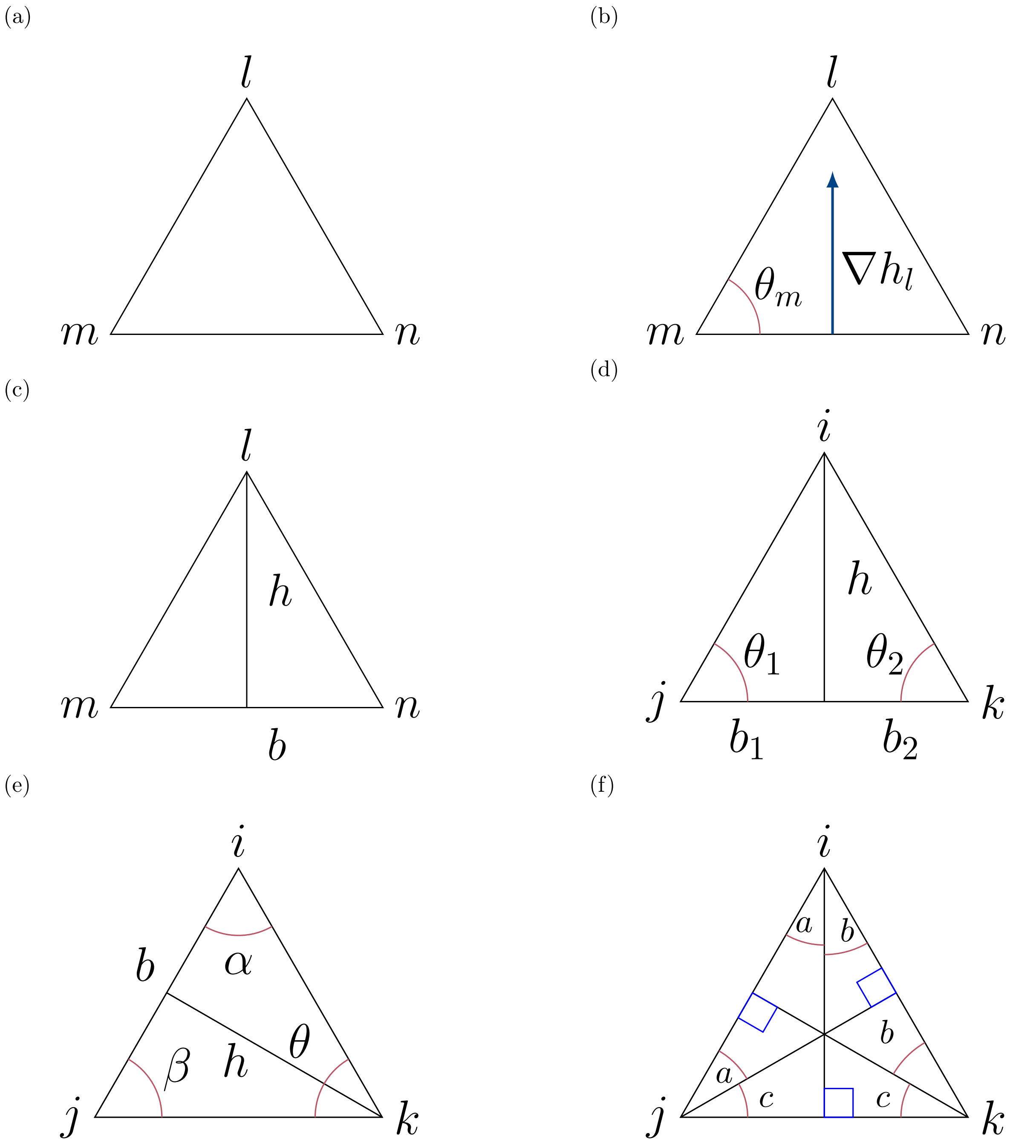


Figure 2.8: Various figures to clarify the evaluation of A_{ij}. (a) A triangle lmn. (b) The angle of ∇h_l relative to θ_m. (c) The height h of the triangle with edge mn as the base b. (d) The relevant quantities for evaluating A_{ii}. (e) The relevant quantities for evaluating A_{ij}. (f) Circumcenter and the angles associated with the perpendicular bisectors.

linear interpolation of any function $f(\mathbf{x})$. On this triangle the linear interpolation of $f(\mathbf{x})$ is given by

$$f(\mathbf{x}) = f(\mathbf{x}_l)h_l(\mathbf{x}) + f(\mathbf{x}_m)h_m(\mathbf{x}) + f(\mathbf{x}_n)h_n(\mathbf{x})\,, \tag{2.3.37}$$

and setting $f(\mathbf{x}) = 1$ yields the identity

$$h_l(\mathbf{x}) + h_m(\mathbf{x}) + h_n(\mathbf{x}) = 1\,. \tag{2.3.38}$$

Evaluating the gradient of Eq. (2.3.37) yields, noting that as the basis functions are linear their gradients are constant

$$\nabla f(\mathbf{x}) = (f_l - f_n)\nabla h_l + (f_m - f_n)\nabla h_n\,, \tag{2.3.39}$$

where we have used the identity that $\nabla h_n = -\nabla h_l - \nabla h_m$ from Eq. (2.3.38). To find an analytical form of the basis function gradients, we note that for basis function $h_l(\mathbf{x})$ that $h_l(\mathbf{x}_l) = 1$, $h_l(\mathbf{x}_m) = 0$, and $h_l(\mathbf{x}_n) = 0$. Hence the Taylor series of $h_l(\mathbf{x})$ about $\mathbf{x}_m$ and $\mathbf{x}_n$ gives for the edges lm, ln, and nm

$$\nabla h_l \cdot (\mathbf{x}_l - \mathbf{x}_m) = 1\,, \tag{2.3.40}$$

$$\nabla h_l \cdot (\mathbf{x}_l - \mathbf{x}_n) = 1\,, \tag{2.3.41}$$

$$\nabla h_l \cdot (\mathbf{x}_n - \mathbf{x}_m) = 0\,. \tag{2.3.42}$$

Equation (2.3.42) specifies the direction of ∇h_l to be orthogonal to $\mathbf{x}_n - \mathbf{x}_m$, which is found by rotating $\mathbf{x}_n - \mathbf{x}_m$ ninety degrees counterclockwise. This direction is denoted as $(\mathbf{x}_n - \mathbf{x}_m)^\perp$. The magnitude is found from Eq. (2.3.40), which from the dot product (Fig. 2.8b)

$$1 = \nabla h_l \cdot (\mathbf{x}_l - \mathbf{x}_m) = \|h_l\|\|\mathbf{x}_l - \mathbf{x}_m\| \cos\left(\frac{\pi}{2} - \theta_m\right)\,. \tag{2.3.43}$$

Denoting $\|\mathbf{x}_l - \mathbf{x}_m\|$ as ℓ_{lm}, application of a trigonometric identity and rearrangement of Eq. (2.3.43) to find $\|\nabla h_l\|$ yields

$$\|\nabla h_l\| = \frac{1}{\ell_{lm}\sin\theta_m} = \frac{1}{h}\,, \tag{2.3.44}$$

where h is the height of the triangle lmn if edge mn is taken as the base b (Fig. 2.8c). As the area of triangle lmn is

$$A_{lmn} = \frac{1}{2}\ell_{mn}h\,, \tag{2.3.45}$$

the gradient ∇h_l is then

$$\nabla h_l = \frac{1}{2A_{lmn}}(\mathbf{x}_n - \mathbf{x}_m)^\perp\,. \tag{2.3.46}$$

Having obtained the gradients ∇h_l, the entries of A_{ij} can now be evaluated. To begin the contribution of basis function ∇h_i with itself to A_{ii} are evaluated. Looking at the contributions on a single triangle that contains vertex i yields

$$\int_{\text{triangle } ijk} \mathrm{d}a\ \nabla h_i \cdot \nabla h_i = A_{ijk}\|\nabla h_i\|^2 = \frac{A_{ijk}}{h^2} = \frac{\ell_{jk}}{2h}\,. \tag{2.3.47}$$

For computational convenience, Eq. (2.3.47) is rewritten in terms of cotangents inside triangle ijk. To do so, we note that the edge jk can be divided by a perpendicular line originating from vertex i (Fig. 2.8d). This divides ℓ_{jk} into lengths b_1 and b_2. Denoting the angle of vertex j as θ_1 and the angle of vertex k as θ_2, we obtain

$$\ell_{jk} = h\frac{b_1 + b_2}{h} = h\cot\theta_1 + h\cot\theta_2\,. \tag{2.3.48}$$

Substitution of Eq. (2.3.48) into Eq. (2.3.47) yields

$$\int_{\text{triangle } ijk} \mathrm{d}a\ \nabla h_i \cdot \nabla h_i = \frac{1}{2}\left(\cot\theta_1 + \cot\theta_2\right)\,. \tag{2.3.49}$$

We proceed to evaluate the contribution of ∇h_i with ∇h_j for $i \neq j$ on triangle ijk. Use of Eq. (2.3.46) obtains

$$\int_{\text{triangle } ijk} \mathrm{d}a\ \nabla h_i \cdot \nabla h_j = \frac{1}{4A_{ijk}}(\mathbf{x}_j - \mathbf{x}_k)^\perp \cdot (\mathbf{x}_k - \mathbf{x}_i)^\perp\,. \tag{2.3.50}$$

We label the angles associated with vertices i, j, and k as α, β, and θ, respectively (Fig. 2.8e). Expanding the dot product, setting the base to be edge ij with associated height h, and use of trigonometric relations yields

$$\begin{aligned}
\int_{\text{triangle } ijk} \mathrm{d}a\ \nabla h_i \cdot \nabla h_j &= -\frac{1}{4A_{ijk}}\|(\mathbf{x}_j - \mathbf{x}_k)^\perp\|\|(\mathbf{x}_k - \mathbf{x}_i)^\perp\|\cos\theta && (2.3.51)\\
&= -\frac{1}{2bh}\frac{h}{\sin\beta}\frac{h}{\sin\alpha}\cos\theta && (2.3.52)\\
&= -\frac{1}{2}\frac{h\cos\theta}{(h\cot\alpha + h\cot\beta)\sin\alpha\sin\beta} && (2.3.53)\\
&= -\frac{1}{2}\frac{\cos\theta}{\cos\alpha\sin\beta + \cos\beta\sin\alpha} && (2.3.54)\\
&= -\frac{1}{2}\cot\theta\,. && (2.3.55)
\end{aligned}$$

We evaluate the entries of A_{ij} by computing Eqs. (2.3.49) and (2.3.55) on all triangles sharing vertices i and j. To conclude, for A_{ii} we obtain

$$A_{ii} = \int_{\text{triangles with vertex } i} \mathrm{d}a\ \nabla h_i \cdot \nabla h_i = \sum_{j(i)} \frac{1}{2}\left(\cot\theta_1 + \cot\theta_2\right)\,, \tag{2.3.56}$$

where $j(i)$ is a summation over all vertices j that share an edge with i, and θ_1 and θ_2 are the angles opposite edge ij, and for A_{ij} we obtain

$$A_{ij} = \int_{\text{triangles with vertices } i,j} \mathrm{d}a\ \nabla h_i \cdot \nabla h_j = -\frac{1}{2}\left(\cot\theta_1 + \cot\theta_2\right)\,. \tag{2.3.57}$$

Setting $\mathbf{y} = \mathbf{A}\hat{\mathbf{x}}$ yields for $\mathbf{y}$

$$y_i = \sum_{j(i)} \frac{1}{2}(x_i - x_j)(\cot\theta_1 + \cot\theta_2)\,. \tag{2.3.58}$$

The values of B_{ij} are obtained by direct computation. This is done by evaluating the integral of $h_i h_j$ using an isoparametric mapping on which the integral is evaluated on a simplified space that is mapped to the true domain using the Jacobian of the mapping [57]. The reference space is taken to be a triangle with vertices 1, 2, and 3 where

$$\mathbf{x}_1 = [0,0,0]^{\mathsf{T}}\ , \tag{2.3.59}$$
$$\mathbf{x}_2 = [1,0,0]^{\mathsf{T}}\ , \tag{2.3.60}$$
$$\mathbf{x}_3 = [0,1,0]^{\mathsf{T}}\ . \tag{2.3.61}$$

On this triangle the basis function for $h_1(\mathbf{x})$ associated with vertex 1 and $h_2(\mathbf{x})$ associated with vertex 2 are

$$h_1(\mathbf{x}) = 1 - x - y\,, \tag{2.3.62}$$
$$h_2(\mathbf{x}) = x\,. \tag{2.3.63}$$

The relevant integrals for B_{ij} are evaluated to obtain

$$\int_{\text{triangle 123}} \mathrm{d}a\ h_1^2(\mathbf{x}) = \int_0^1 \int_0^{1-x} \mathrm{d}y\,\mathrm{d}x\ \ (1-x-y)^2 = \frac{1}{12}\,, \tag{2.3.64}$$
$$\int_{\text{triangle 123}} \mathrm{d}a\ h_1(\mathbf{x})h_2(\mathbf{x}) = \int_0^1 \int_0^{1-x} \mathrm{d}y\,\mathrm{d}x\ \ (1-x-y)\,x = \frac{1}{24}\,. \tag{2.3.65}$$

To apply these results to general vertices i, j, k, we use the parameterization

$$\mathbf{x}(u,v) = \mathbf{x}_i + (\mathbf{x}_j - \mathbf{x}_i)u + (\mathbf{x}_k - \mathbf{x}_i)v\,, \tag{2.3.66}$$

where $0 \le u, v, 1-u-v \le 1$, which corresponds in u, v space to the vertices specified by Eqs. (2.3.59)–(2.3.61). Equation (2.3.66) is used to evaluate the needed integral for B_{ij} through change of variables, which yields for a general function $f(\mathbf{x})$ [60]

$$\int_{\text{triangle } ijk} \mathrm{d}a\ f(\mathbf{x}) = \int_{\text{triangle 123}} \mathrm{d}u\,\mathrm{d}v\ f(u,v)\|\mathbf{N}(u,v)\|\,, \tag{2.3.67}$$

where $\mathbf{N}$ is the normal to the tangents of $\mathbf{x}(u,v)$ evaluated through

$$\mathbf{N}(u,v) = \partial_u \mathbf{x}(u,v) \times \partial_v \mathbf{x}(u,v) = (\mathbf{x}_j - \mathbf{x}_i) \times (\mathbf{x}_k - \mathbf{x}_i)\,. \tag{2.3.68}$$

As $\|N(u,v)\| = 2A_{ijk}$, Eq. (2.3.67) yields the generalizations of Eqs. (2.3.64) and (2.3.65) for any triangle ijk

$$\int_{\text{triangle } ijk} \mathrm{d}a \; h_i^2(\mathbf{x}) = \frac{A_{ijk}}{6}\,, \tag{2.3.69}$$

$$\int_{\text{triangle } ijk} \mathrm{d}a \; h_i(\mathbf{x}) h_j(\mathbf{x}) = \frac{A_{ijk}}{12}\,. \tag{2.3.70}$$

One evaluates B_{ij} using Eqs. (2.3.69) and (2.3.70) to yield

$$B_{ii} = \sum_{T(i)} \frac{1}{6} A_{ij'k'}\,, \tag{2.3.71}$$

$$B_{ij} = \sum_{T(i,j)} \frac{1}{12} A_{ijk'}\,, \tag{2.3.72}$$

where $T(i)$ and $T(i,j)$ are the sets of triangles containing vertex i and triangles containing edge ij, respectively. Using the identity that $\sum_i h_i(\mathbf{x}) = 1$ for all $\mathbf{x}$, the B_{ij} factors yield the surface area of the triangulation per

$$\int \mathrm{d}a \; 1 = \int \mathrm{d}a \; \sum_j h_j \tag{2.3.73}$$

$$= \int \mathrm{d}a \; \sum_j h_j \left(\sum_i h_i \right) \tag{2.3.74}$$

$$= \int \mathrm{d}a \; \sum_{ij} B_{ij} = \mathbf{1}^{\intercal}\mathbf{B}\mathbf{1} = A\,. \tag{2.3.75}$$

Having schemes to compute $\mathbf{A}$ and $\mathbf{B}$, one obtains the vertex-wise surface Laplacian by inverting $\mathbf{B}$ in Eq. (2.3.36) to yield $\hat{\mathbf{g}}$. However, the inversion of a matrix is a computationally costly operation that scales cubically with the number of vertices and makes the vertex-wise surface Laplacian a non-local operator due to the inverse matrix of $\mathbf{B}$ being dense [61]. However, the inversion of a matrix is a computationally costly operation that scales cubically with the number of vertices, and as the inverse matrix of $\mathbf{B}$ is dense [61] the vertex-wise surface Laplacian is non-local. To construct a vertex-wise surface Laplacian that depends only on vertices that share an edge with vertex i, we use a lumping procedure to approximate $\mathbf{B}$ with a diagonal matrix $\mathbf{C}$ [62]. In the lumping procedure, the diagonals of the new matrix $\mathbf{C}$ correspond to the area associated with vertex i. Multiple choices for the areas are possible, but due to the evaluation of A_{ij} using many terms that serve useful in the following derivation, we will evaluate $\mathbf{C}$ using the Voronoi cell areas constructed from the circumcenters of the triangles (see Fig. 2.9a) [59].

To proceed, one uses the property that the perpendicular bisectors of each edge of a triangle yield the Voronoi center [63]. Drawing lines from the vertices to the circumcenter,

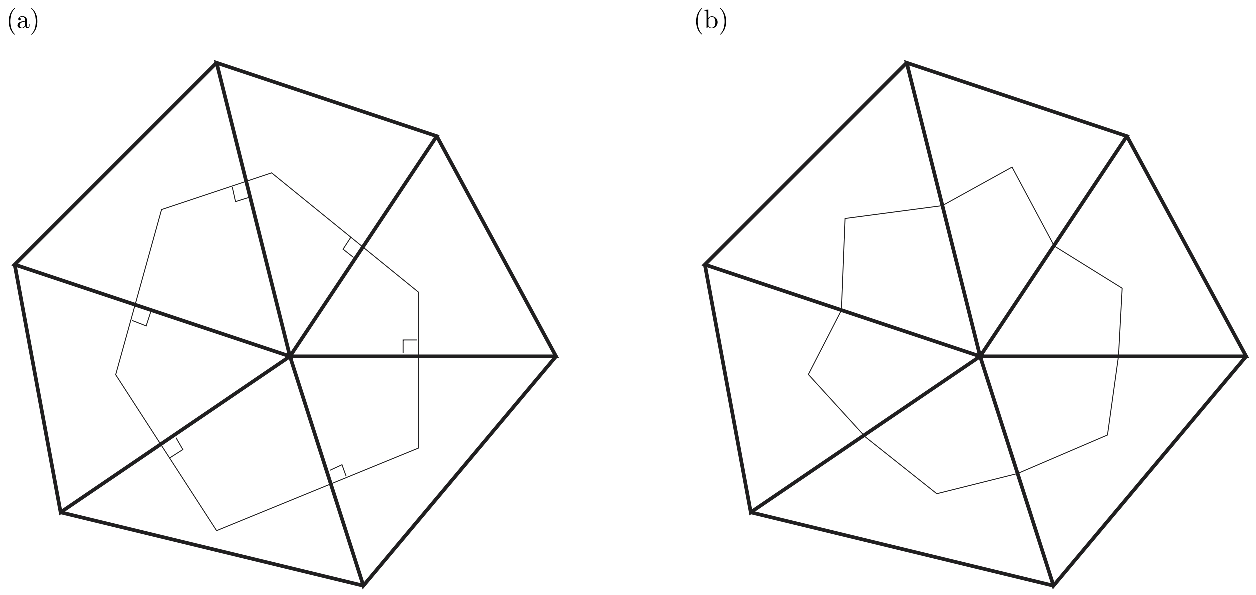


Figure 2.9: Examples of the Voronoi and barycentric cell constructions. Reprinted/adapted by permission from Springer Nature Customer Service Centre GmbH: Springer Nature, "Discrete Differential-Geometry Operators for Triangulated 2-Manifolds" Mark Meyer *et al.*, Copyright Springer-Verlag Berlin Heidelberg (2003) [59]. (a) The Voronoi cell around a vertex, formed from the circumcenters of the triangles that vertex is part of. (b) The barycentric cell around a vertex, formed from the centroids of the triangles that vertex is part of.

three triangles are generated (Fig. 2.8f). The angles of these subtriangles are related to that of triangle ijk by

$$\angle i = a + b\,, \tag{2.3.76}$$
$$\angle j = a + c\,, \tag{2.3.77}$$
$$\angle k = b + c\,. \tag{2.3.78}$$

The angles a, b, and c satisfy the relation

$$a + b + c = \frac{\pi}{2}\,, \tag{2.3.79}$$

which is used with Eqs. (2.3.77) and (2.3.78) to obtain

$$a = \frac{\pi}{2} - \angle k\,, \tag{2.3.80}$$
$$b = \frac{\pi}{2} - \angle j\,. \tag{2.3.81}$$

To find the area associated with vertex i in triangle ijk, we find the subareas associated with angle a and b. The subarea associated with angle b, A_b, is

$$A_b = \frac{1}{2}\frac{\ell_{ik}}{2}\frac{\ell_{ik}}{2}\tan\left(\frac{\pi}{2} - \angle j\right) = \frac{\ell_{ik}^2}{8}\cot\angle j\,, \tag{2.3.82}$$

where the subarea associated with angle a, A_a, is found similarly. We obtain the area associated with vertex i in triangle ijk by summing over all triangles vertex i is part of to yield

$$C_i = \frac{1}{8}\sum_{j(i)} \ell_{ij}^2 \left(\cot\theta_1 + \cot\theta_2\right)\,. \tag{2.3.83}$$

Approximating $\mathbf{B}$ as $\mathbf{C}$, the component-wise surface Laplacian $\hat{\mathbf{g}}$ is

$$g_i = \frac{\sum_{j(i)} \frac{1}{2}\left(x_j - x_i\right)\left(\cot\theta_1 + \cot\theta_2\right)}{\frac{1}{8}\sum_{j(i)} \ell_{ij}^2(\cot\theta_1 + \cot\theta_2)}\,. \tag{2.3.84}$$

Defining the cotangent dependent terms as

$$\sigma_{ij} = \frac{l_{ij}}{2}\left(\cot\left(\theta_1\right) + \cot\left(\theta_2\right)\right)\,, \tag{2.3.85}$$

$$\Delta a_i = \sigma_i = \frac{1}{4}\sum_{j(i)} l_{ij}\sigma_{ij}\,, \tag{2.3.86}$$

one obtains for g_i

$$g_i = \frac{1}{\sigma_i}\sum_{j(i)}\frac{\sigma_{ij}}{\ell_{ij}}\left(x_j - x_i\right)\,. \tag{2.3.87}$$

As Eq. (2.3.87) holds for any spatial dimensions, its extension to three dimensional space yields for the surface Laplacian at $\mathbf{x}_i$

$$\mathbf{g}(\mathbf{x}_i) = \frac{1}{\sigma_i}\sum_{j(i)}\frac{\sigma_{ij}}{\ell_{ij}}\left(\mathbf{x}_j - \mathbf{x}_i\right)\,. \tag{2.3.88}$$

Substitution of Eq. (2.3.88) into Eq. (2.3.25) obtains the vertex-wise mean curvature as

$$H_i = \frac{1}{2}\left[\frac{1}{\sigma_i}\sum_{j(i)}\frac{\sigma_{ij}}{\ell_{ij}}\left(\mathbf{x}_i - \mathbf{x}_j\right)\right]\cdot\mathbf{n}_i\,, \tag{2.3.89}$$

where $\mathbf{n}_i$ is the vertex-wise surface normal. The vertex-wise surface normal is evaluated as the area-weighted average of all triangle normals that vertex i is on to yield

$$\mathbf{n}_i = \frac{\sum_{T(i)} A_{ij'k'}\mathbf{n}_{ij'k'}}{\sum_{T(i)} A_{ij'k'}}\,. \tag{2.3.90}$$

Other choices of vertex-wise surface normals are possible, but the area-weighted average is used as it provides both good speed and accuracy [56]. Equations (2.3.86) and (2.3.89) constitute the terms used in the discrete Helfrich model Eq. (2.3.1) for the Gompper-Kroll model [38].

We now discuss the use of a maximum edge length of $\sqrt{2.8}\,\sigma$. This condition along with the hard sphere conditions sets the maximum angle obtainable on the triangles to be 104 degrees [38]. Obtuse angles can cause σ_{ij} to be negative per Eq. (2.3.85) σ_{ij}, and the sum of both positive and negative σ_{ij} can yield $\sigma_i \leq 0$. As the overall energy given by the sum of $\sigma_i H_i^2$ terms scales inversely with σ_i, values of $\sigma_i = 0$ causes the Gompper-Kroll model to be numerically unstable. This is hence avoided by the use of hard spheres and maximum edge length constraints which limit the obtuseness of the triangles.

Using the σ_{ij} and σ_i terms Eq. (2.3.22) can be made shape-independent. This yields [38]

$$V_\mathrm{b} = \frac{1}{2} k_\mathrm{b} \sum_i \sum_{j(i)} \frac{l_{ij}}{\sigma_{ij}} \left(\mathbf{n}_{ijk} - \mathbf{n}_{ijl}\right)^2 \,, \tag{2.3.91}$$

where ijk and ijl are the two triangles that share edge ij. While this corrects for the shape-dependent issues in Eq. (2.3.22), it is possible that $\sigma_{ij} \approx 0$ which leads to this discretization being numerically unstable. Note that the prefactor of $\frac{\sqrt{3}}{2}$ needed for use of Eq. (2.3.22) on a cylinder can be recovered from Eq. (2.3.91) by assuming that all triangles are equilateral. Under this assumption $\sigma_{ij} = \frac{1}{\sqrt{3}} l_{ij}$, and hence

$$V_\mathrm{b} = \frac{1}{2} k_\mathrm{b}' \sum_i \sum_{j(i)} \sqrt{3} \left(\mathbf{n}_{ijk} - \mathbf{n}_{ijk}\right)^2 = \frac{\sqrt{3}}{2} k_\mathrm{b}' \sum_{\langle \alpha,\beta \rangle} \left(1 - \mathbf{n}_\alpha \cdot \mathbf{n}_\beta\right) , \tag{2.3.92}$$

as wanted.

2.3.4 Jülicher Model

We will now consider a shape-independent discretization of Eq. (2.3.1) that relies on the dihedral angles of the triangles. This model, the Jülicher model, was first derived by Jülicher [65] and then discussed in more detail by Atilgan and Sun [64]. The curvature at vertex i in this model is given by

$$H_i = \frac{1}{2\Delta a_i} \sum_{j(i)} \left[\tan\left(\frac{\theta_{ij}}{2}\right) + \sin\left(\frac{\theta_{ij}}{2}\right) \right] \frac{\ell_{ij}}{2} \,, \tag{2.3.93}$$

where Δa_i is given by the barycentric cell area (see Fig. 2.9b), which is obtained as

$$\Delta a_i = \frac{1}{3} \sum_{T(i)} A_{ij'k'} \,, \tag{2.3.94}$$

and θ_{ij} is the dihedral angle across edge ij.

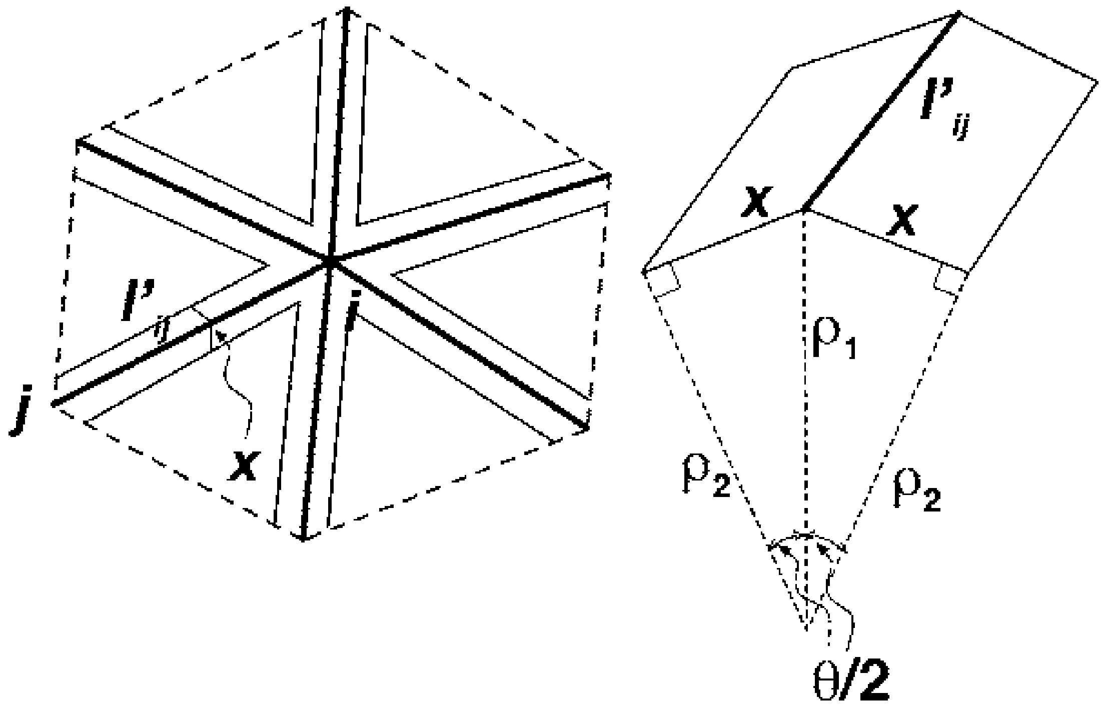


Figure 2.10: Integration setup to derive the Jülicher model. (Left) The triangulation about a vertex i with edge ij, and the thin stripes of width x that the integration is performed on. (Right) The short and long components of the principle direction perpendicular to the edge ij. Reprinted from E. Atilgan and S. X. Sun, "Shape transitions in lipid membranes and protein mediated vesicle fusion and fission", The Journal of Chemical Physics **126**, 03B604 (2007), with the permission of AIP Publishing [64].

To obtain Eq. (2.3.93) we will evaluate the mean curvature of each vertex about its barycentric cell area. This is evaluated as

$$H_i = \frac{1}{\Delta a_i} \int_{\Delta a_i} \mathrm{d}a \, \frac{\kappa_1 + \kappa_2}{2} \, . \tag{2.3.95}$$

On a triangulated surface, we select the principle directions to be along the edge ij and the direction perpendicular to the edge ij. Integration of Eq. (2.3.95) is 0 on much of the triangulated surface as it is flat. The parts of the triangle that have a nonzero contribution to Eq. (2.3.95) are the edges, which we target by integrating about them with an infinitesimal width x (Fig. 2.10(left)). The integration in Eq. (2.3.95) is then evaluated as

$$H_i = \lim_{x \to 0} \frac{1}{\Delta a_i} \int_{\frac{1}{2}\ell_{ij}} 2x \, \mathrm{d}\ell_{ij} \, \frac{\kappa_1 + \kappa_2}{2} \, . \tag{2.3.96}$$

The curvature along the first principle direction, κ_1, is 0 due to the edge being flat. The curvature along the second principle direction, κ_2, is related to the dihedral angle of edge

ij (Fig. 2.10(right)). This contribution is approximated as two parts, with a longer inner contribution ρ_1 and a shorter outer contribution ρ_2. These terms can be evaluated using trigonometry to yield for κ_1

$$\rho_1 = \frac{x}{\sin\frac{\theta_{ij}}{2}}\,, \tag{2.3.97}$$

$$\rho_2 = \frac{x}{\tan\frac{\theta_{ij}}{2}}\,, \tag{2.3.98}$$

$$\kappa_1 = \frac{1}{2}\left(\rho_1^{-1} + \rho_2^{-1}\right)\,. \tag{2.3.99}$$

Substitution of Eqs. (2.3.97)–(2.3.99) into Eq. (2.3.96) yields Eq. (2.3.93).

2.4 Testings of the Discretizations

We now test the various discretizations of the Helfrich model on two separate model problems. The first problem is a convergence study of the area and bending energy on a cylinder and a sphere. The second problem will look at the height fluctuations of a planar periodic surface, where tools from statistical mechanics will be used to relate the height fluctuations to a measure of the bending rigidity and surface tension. The second study will also look at the interconnection between different area ensembles, the surface tension, and the frame tension, and see what contributes to the fluctuating surface tension that is measured from the height fluctuations.

2.4.1 Energies of Cylinders and Spheres

We now evaluate the bending energy and area expressions of the Kantor-Nelson, Gompper-Kroll, and Jülicher model on triangulations of cylinders and spheres for which analytical results can be computed. For a cylinder, the area and bending energy are given by

$$A_{\text{cylinder}} = 2\pi R L\,, \tag{2.4.1}$$

$$V_{\text{b,cylinder}} = \int \mathrm{d}a\; k_{\text{b}} H^2 = \int \mathrm{d}a\; k_{\text{b}} (\frac{1}{2R})^2 = \frac{\pi k_{\text{b}} L}{2R}\,, \tag{2.4.2}$$

where R and L are the radius and length of the cylinder, respectively. For a sphere, the area and bending energy are given by

$$A_{\text{sphere}} = 4\pi R^2\,, \tag{2.4.3}$$

$$V_{\text{b,sphere}} = \int \mathrm{d}a\; k_{\text{b}} H^2 = \int \mathrm{d}a\; k_{\text{b}} (\frac{1}{R})^2 = 4\pi k_{\text{b}}\,, \tag{2.4.4}$$

where R is the radius of the sphere.

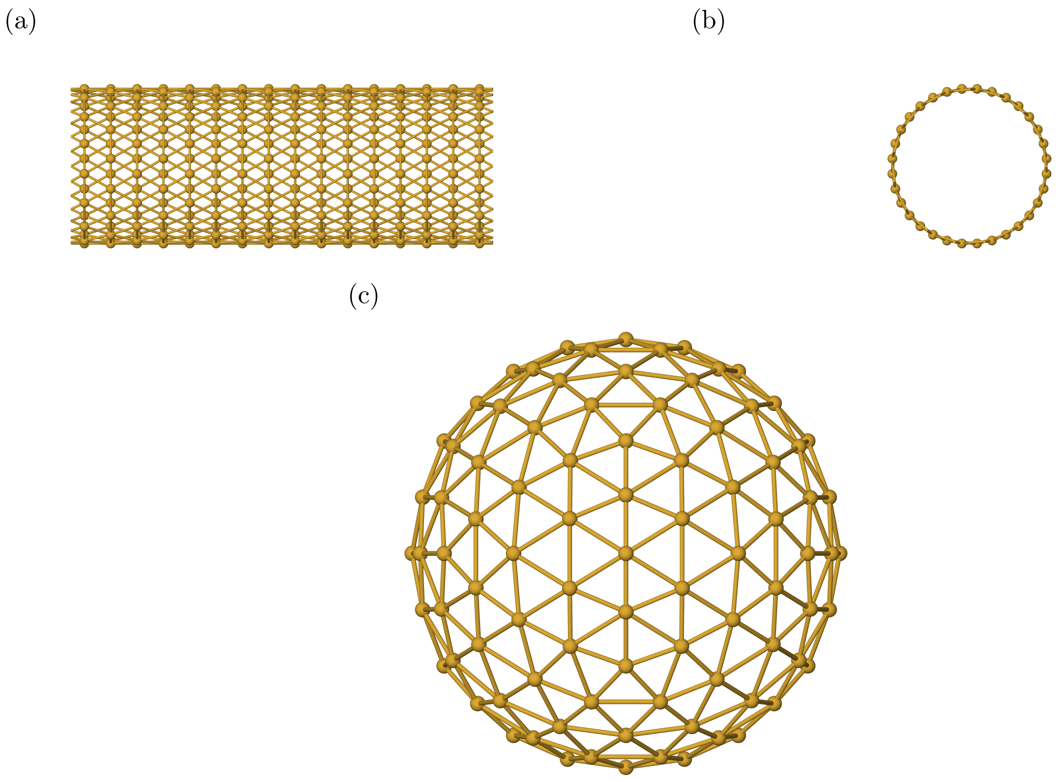


Figure 2.11: (a) Lengthwise-view of a triangulation of a cylinder with 196 vertices. (b) Radialwise-view of a triangulation of a cylinder with 196 vertices. (c) View of a triangulation of a sphere with 162 vertices.

The triangulation of a cylinder is generated with N_L vertices in the length direction (Fig. 2.11a) and N_R vertices in the radial direction (Fig. 2.11b). Equilateral triangles are created using alternating radial spacings of vertices in the length direction. To do so, we index a vertex in the length and radial-wise directions with indices i and j, respectively, to yield the total index as $n = j + iN_R$. The coordinates of vertex n are then given by

$$x_n = L\frac{i}{N_L}\,, \tag{2.4.5}$$

$$y_n = R\cos\left(2\pi\frac{j+\omega_j}{N_R}\right)\,, \tag{2.4.6}$$

$$z_n = R\sin\left(2\pi\frac{j+\omega_j}{N_R}\right)\,, \tag{2.4.7}$$

where ω_j is 0.25 for i even and 0.75 for i odd.

The triangulation of a sphere is generated using Loop subdivision [66]. In this scheme, an initial coarse triangulation on an icosahedron is refined to yield finer triangulations by iteratively subdividing the triangles. The subdivision occurs by introducing the midpoints of each triangle as new vertices. Connecting the new vertices into new triangles generates 3 additional triangles per original triangle. The spherical shape is maintained by projecting the position of the vertices onto that of a sphere, yielding triangulations such as that of Fig. 2.11c. As we initialize the spherical triangulation from an icosahedron, each triangulation has 12 vertices with 5 edges while all other vertices have 6 edges. As the total number of vertices, edges, and triangles is related to the integrated Gaussian curvature via the Gauss-Bonnet theorem [35], the presence of vertices with 5 edges instead of 6 edges on a spherical triangulation is unavoidable.

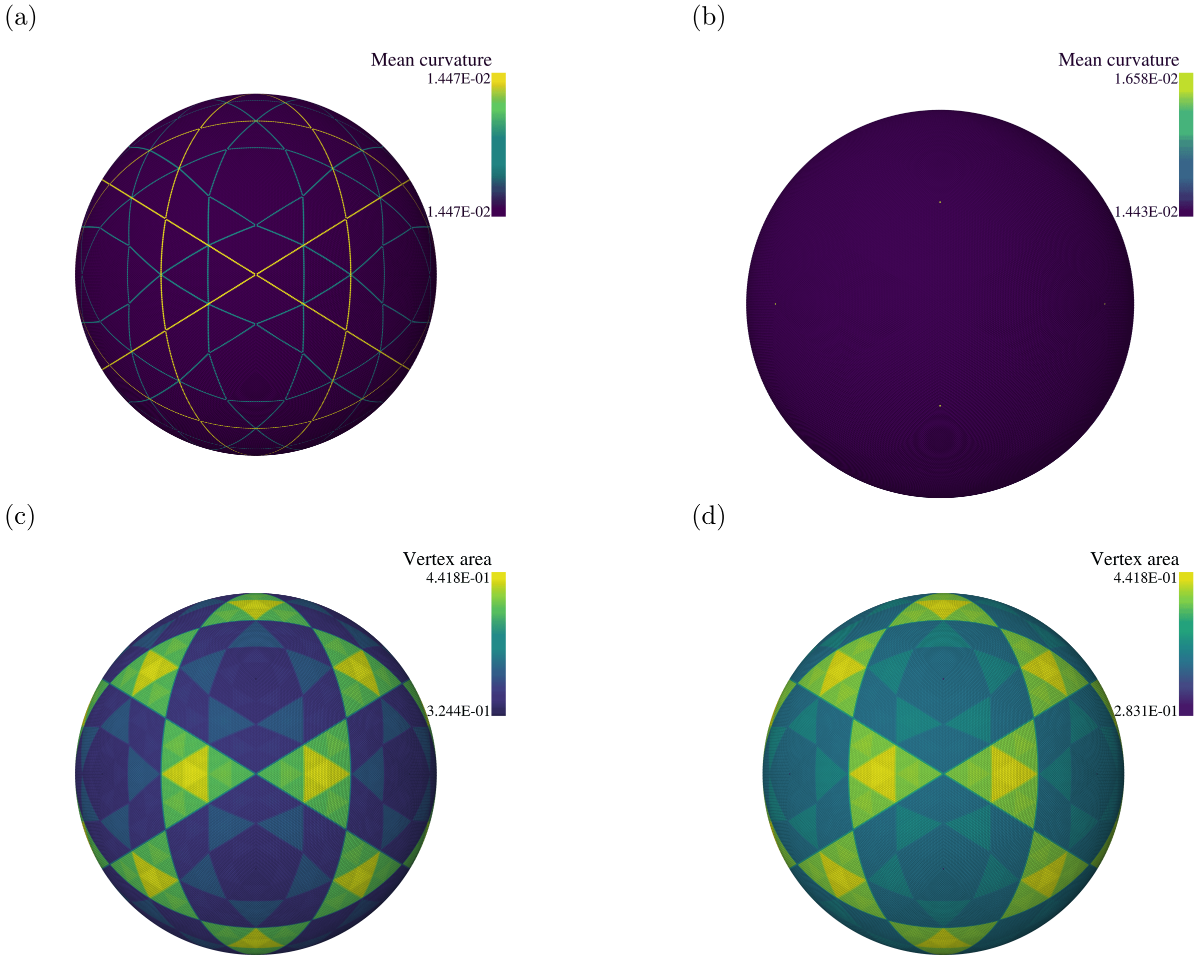


Figure 2.12: H_i and Δa_i on a sphere with 163842 vertices. (a) H_i for the Gompper-Kroll model. (b) H_i for the Jülicher model. (c) Δa_i for the Gompper-Kroll model. (d) Δa_i for the Jülicher model.

The values of H_i and Δa_i are constant for both the Gompper-Kroll and Jülicher model on a cylindrical triangulation due to the triangulation consisting of identical equilateral environments. For a spherical triangulation, the triangulation scheme leads to non-constant H_i and Δa_i on the surface (Fig. 2.12). For H_i, the Gompper-Kroll model yields relatively constant values of H_i (Fig. 2.12a) while the Jülicher model shows slight variance between the 12 five-edge vertices and the other six-edge vertices (Fig. 2.12b). For Δa_i, both the Voronoi cell areas in the Gompper-Kroll model (Fig. 2.12c) and the barycentric cell areas in the Jülicher model yield a pattern in their values due to the Loop subdivision method as shown in Fig. 2.12d.

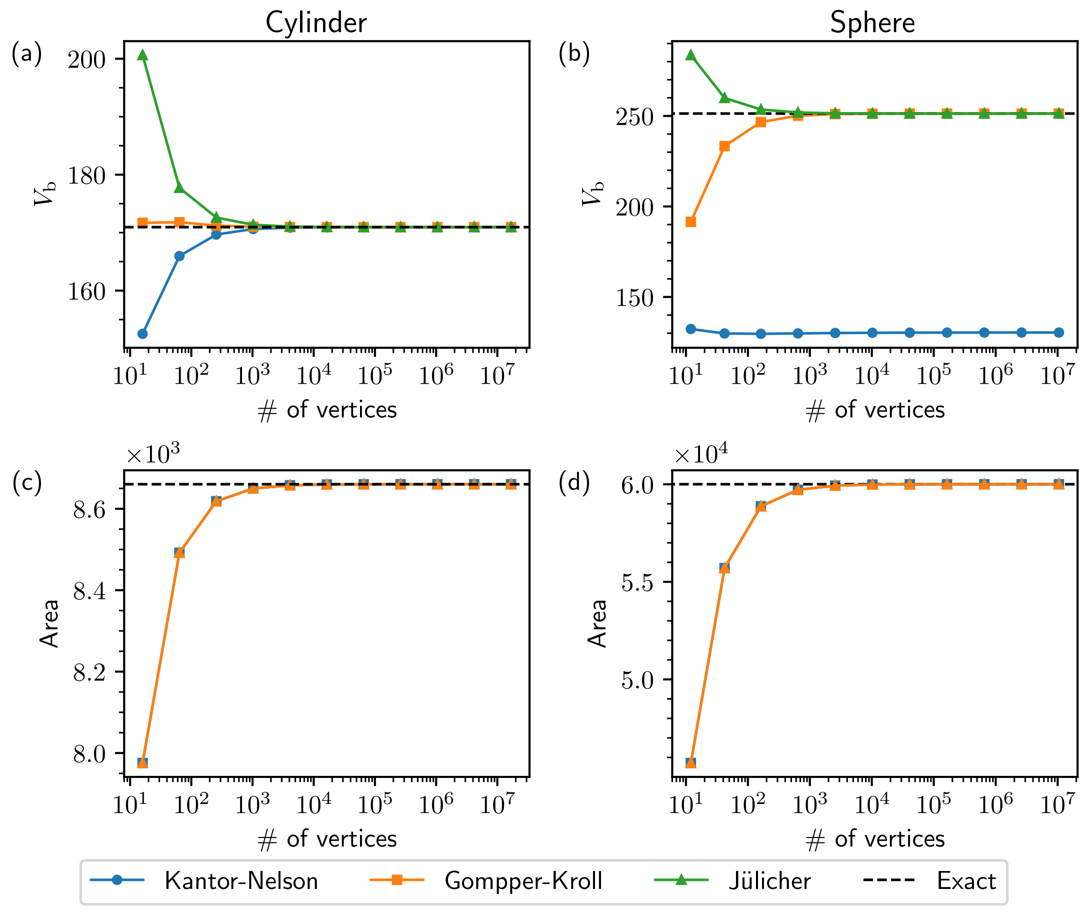


Figure 2.13: The bending energy and area of cylindrical and spherical membrane systems with $k_b = 20\ k_BT$ for varying number of vertices. The cylinder is of length $100\frac{\sqrt{3}}{2}$ and radius $\frac{50}{\pi}$, and the sphere is of radius 69.1.

The bending energy and total area are computed for each method on the cylindrical and spherical triangulations (Fig. 2.13). For the cylindrical bending energy, all models converge to the correct limit given by Eq. (2.4.2) with the Kantor-Nelson model value of k_b being weighted by a factor of $\frac{\sqrt{3}}{2}$ as noted in Section 2.3.2. For the spherical bending energy, the

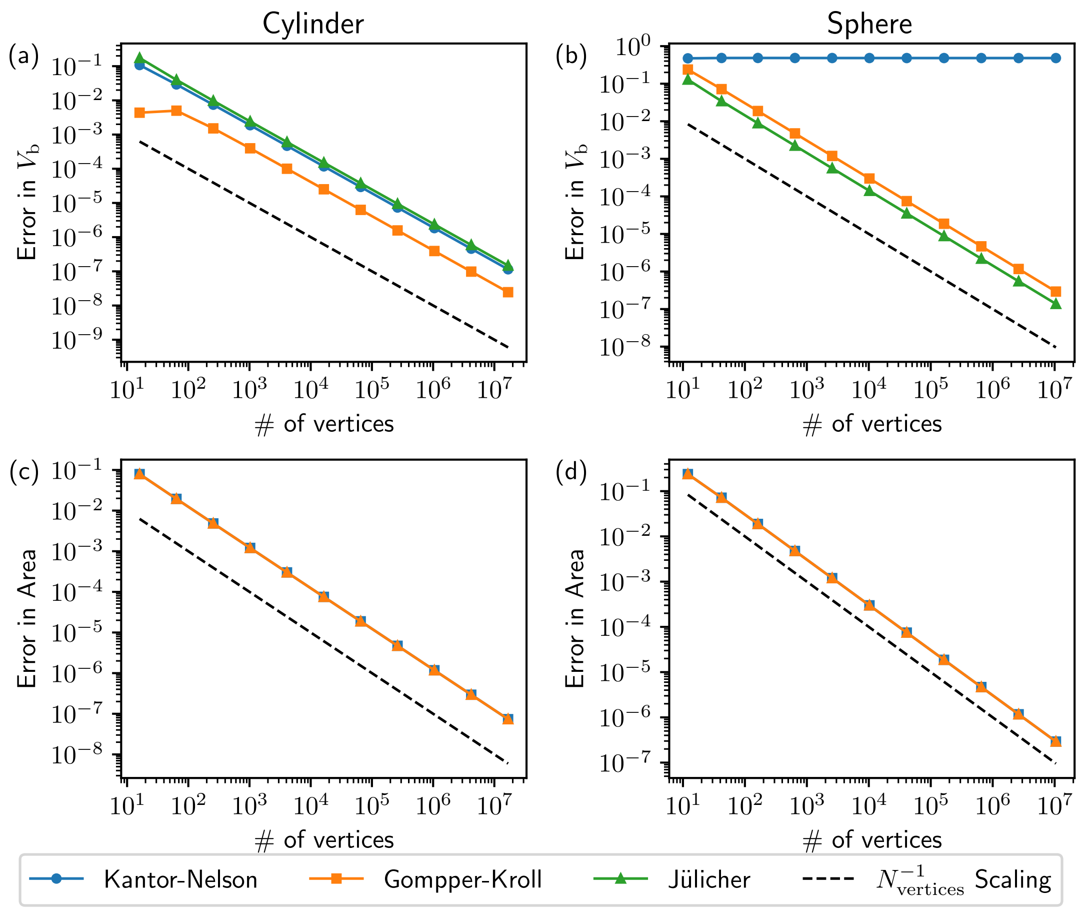


Figure 2.14: The error in the bending energy and area of cylindrical and spherical membrane systems with $k_b = 20\ k_BT$ for varying number of vertices. The cylinder is of length $100\frac{\sqrt{3}}{2}$ and radius $\frac{50}{\pi}$, and the sphere is of radius 69.1.

Gompper-Kroll and Jülicher models converge to the correct limit given by Eq. (2.4.4) while the Kantor-Nelson model converges to a different limit even after rescaling by a factor $\frac{1}{\sqrt{3}}$. This is due to the sphere having a global non-zero Gaussian curvature which the Kantor-Nelson model will have contributions from. The same area is obtained using both schemes on the triangulations which converges to the area of a smooth surface as the number of vertices increases (Fig. 2.13(c)–(d)). We analyze these trends by looking at the relative error between the exact quantities and the discrete quantities in Fig. 2.14. For the models and systems that converge to the correct limits, the convergence has linear scaling with respect to the number of vertices. For the Gompper-Kroll model, this is the expected scaling with linear finite elements [3].

2.4.2 Fluctuations of a Planar Periodic Surface

We now analyze the Gompper-Kroll model using its height fluctuations via results from capillary wave theory [67, 68]. Capillary wave theory relates the height fluctuations of the membrane surface, induced through the Monte Carlo moves, to the bending rigidity and surface tension of the membrane. To do so, we begin with a planar periodic system with a fixed projected area. The potential energy of this system is given by the sum of Eqs. (2.2.3) and (2.2.8) to yield

$$V_{\text{fluct}} = \int \mathrm{d}a \, \left(k_{\text{b}} H^2 + r\right) , \tag{2.4.8}$$

where r is the surface tension of this model. To obtain a form of Eq. (2.4.8) where integrals involving $\exp(-\beta V_{\text{fluct}})$ can be done analytically, the Monge parameterization of the membrane surface is used in which the z coordinate is given by a height-field $h(x, y)$ [69]. The Monge parameterization has area element

$$A = \int \mathrm{d}A \, \sqrt{1 + \left(\frac{\partial h(x,y)}{\partial x}\right)^2 + \left(\frac{\partial h(x,y)}{\partial y}\right)^2} = \int \mathrm{d}A \, \sqrt{1 + (\nabla h(x,y))^2} , \tag{2.4.9}$$

where $\mathrm{d}A$ indicates integration over the xy plane in which

$$\int \mathrm{d}A = L^2 = A_p , \tag{2.4.10}$$

where L is the length of the system in both the x and y direction. The Monge parameterization is suitable when the membrane does not map to more than one part per (x, y) value, which is satisfied for sufficiently high values of bending rigidity or surface tension.

We proceed by expanding Eq. (2.4.8) to quadratic order about a small perturbation to the membrane surface of the form

$$h(x, y) = h_0 + \delta h(x, y) . \tag{2.4.11}$$

The $O(\delta h(x, y))$ expansion of H from Eq. (2.3.25) is

$$H \approx -\frac{1}{2}\nabla^2 h(x, y) = -\frac{1}{2}\nabla^2 \delta h(x, y) , \tag{2.4.12}$$

and the $O(\delta h(x, y)^2)$ expansion of the area element is

$$\sqrt{1 + (\nabla h(x,y))^2} \approx 1 + \frac{1}{2}\left(\nabla \delta h(x,y)\right)^2 . \tag{2.4.13}$$

Hence, the quadratic expansion of V_{fluct} is

$$V_{\text{fluct}} \approx rL^2 + \int \mathrm{d}A \, \left[\frac{1}{4} k_b \left(\nabla^2 \delta h(x,y)\right)^2 + \frac{r}{2}\left(\nabla \delta h(x,y)\right)^2\right] . \tag{2.4.14}$$

To evaluate the partition function corresponding to Eq. (2.4.14), $\delta h(x,y)$ is written in terms of a Fourier series

$$\delta h(x,y) = \sum_{k_m}\sum_{k_n} \delta h_k e^{ik_m x + ik_n y}\,, \tag{2.4.15}$$

where $k_m = \frac{2\pi m}{L}$ and $k_n = \frac{2\pi n}{L}$ are the wavevectors in the x and y directions, respectively. We obtain for the area term

$$\int \mathrm{d}A \, \left[\left(\frac{\partial \delta h(x,y)}{\partial x}\right)^2 + \left(\frac{\partial \delta h(x,y)}{\partial y}\right)^2\right] \tag{2.4.16}$$

$$= \int \mathrm{d}A \, \left[\left(\sum_{k_m}\sum_{k_n} ik_m \delta h_k e^{ik_m x + ik_n y}\right)^2 + \left(\sum_{k_m}\sum_{k_n} ik_n \delta h_k e^{ik_m x + ik_n y}\right)^2\right] \tag{2.4.17}$$

$$= -\int \mathrm{d}A \, \sum_{k_m}\sum_{k'_m}\sum_{k_n}\sum_{k'_n} \delta h_k \delta h'_k (k_m k'_m + k_n k'_n) e^{i(k_m + k'_m)x + i(k_n + k'_n)y} \tag{2.4.18}$$

$$= -L^2 \sum_{k_m}\sum_{k'_m}\sum_{k_n}\sum_{k'_n} \delta h_k \delta h'_k (k_m k'_m + k_n k'_n) \delta(k_m + k'_m)\delta(k_n + k'_n) \tag{2.4.19}$$

$$= L^2 \sum_{k_m}\sum_{k_n} |\delta h_k|^2 (k_m^2 + k_n^2)\,, \tag{2.4.20}$$

where $\delta\left(k_m + k'_m\right)$ is the Kronecker delta and $|\delta h_k|^2 = \delta h_k \delta h_{-k}$. For the mean curvature term, we obtain

$$\int \mathrm{d}A \, \left[\nabla^2 \delta h(x,y)\right]^2 \tag{2.4.21}$$

$$= \int \mathrm{d}A \, \left[-\sum_{k_m}\sum_{k_n} (k_m^2 + k_n^2) \delta h_k e^{ik_m x + ik_n y}\right]^2 \tag{2.4.22}$$

$$= \int \mathrm{d}A \, \sum_{k_m}\sum_{k'_m}\sum_{k_n}\sum_{k'_n} \delta h_k \delta h'_k (k_m^2 + k_n^2)(k'^2_m + k'^2_n) e^{i(k_m + k'_m)x + i(k_n + k'_n)y} \tag{2.4.23}$$

$$= L^2 \sum_{k_m}\sum_{k'_m}\sum_{k_n}\sum_{k'_n} \delta h_k \delta h'_k (k_m^2 + k_n^2)(k'^2_m + k'^2_n) \delta(k_m + k'_m)\delta(k_n + k'_n) \tag{2.4.24}$$

$$= L^2 \sum_{k_m}\sum_{k_n} |\delta h_k|^2 (k_m^2 + k_n^2)^2\,. \tag{2.4.25}$$

The sum of Eqs. (2.4.20) and (2.4.25) yields an energy quadratic in $|\delta h_k|$

$$V_{\text{fluct}} = L^2 \left(r + \frac{r}{2}\sum_k |\delta h_k|^2 (k_m^2 + k_n^2) + \frac{1}{4}k_b \sum_k |\delta h_k|^2 (k_m^2 + k_n^2)^2\right) \tag{2.4.26}$$

$$= L^2 \left(r + \sum_k |\delta h_k|^2 \left[\frac{r}{2}k^2 + \frac{k_\mathrm{b}}{4}k^4\right]\right)\,, \tag{2.4.27}$$

where the summation is over all wavevectors $k^2 = k_m^2 + k_n^2$.

We now evaluate the height fluctuations $\langle|\delta h_k|^2\rangle$. The partition function of Eq. (2.4.27) is evaluated using properties of Gaussian integrals [70] as

$$Q = \int \prod_k \mathrm{d}\delta h_k \exp(-\beta V_{\text{fluct}}) \tag{2.4.28}$$

$$= \exp\left[-\beta r L^2\right] \int \prod_k \mathrm{d}\delta h_k \exp(-\beta L^2 |\delta h_k|^2 \left[\frac{r}{2}k^2 + \frac{k_\mathrm{b}}{4}k^4\right]) \tag{2.4.29}$$

$$= \exp\left[-\beta r L^2\right] \prod_k \sqrt{\frac{\pi}{\beta L^2 \left(\frac{r}{2}k^2 + \frac{k_\mathrm{b}}{4}k^4\right)}}\,. \tag{2.4.30}$$

The height fluctuations are then found through Eq. (2.3.2)

$$\langle|\delta h_k|^2\rangle = \frac{1}{Q}\int \prod_{k'} \mathrm{d}\delta h_{k'} |\delta h_k|^2 \exp\left(-\beta V_{\text{fluct}}\right) \tag{2.4.31}$$

$$= \frac{1}{\beta L^2 (\frac{1}{2}k_b k^4 + r k^2)}\,. \tag{2.4.32}$$

Defining $k_b' = 2k_b$, we arrive at the formula found in the literature [67, 71]

$$\langle|\delta h_k|^2\rangle = \frac{1}{\beta L^2 (k_b' k^4 + r k^2)}\,. \tag{2.4.33}$$

Fitting Eq. (2.4.32) to height fluctuation data obtained in simulation yields estimates of the bending rigidity and surface tension. To do so the membrane surface is binned spatially to yield a discrete version of $h(x, y)$. This discrete version, $h(i_1\ell, i_2\ell)$, where $\ell = \frac{L}{N}$, is obtained with N_x x grid points and N_y y grid points. This is done by finding all vertices between $x \in (\frac{i_1 L}{N_x}, \frac{(i_1+1)L}{N_x}]$ and $y \in (\frac{i_2 L}{N_y}, \frac{(i_2+1)L}{N_y}]$, which we denote with the set $\mathcal{M}$, and evaluating

$$h(i_1\ell, i_2\ell) = \frac{1}{|\mathcal{M}|}\sum_{\mathbf{x}_i \in \mathcal{M}} z_i\,. \tag{2.4.34}$$

We then convert $h(i_1\ell, i_2\ell)$ to Fourier space using FFTW [72]. The forward Fourier transform is evaluated by, denoting the wavevectors as $k_x \in [0, 2\pi(N_x - 1)/L]$ and $k_y \in [0, 2\pi(N_y - 1)/L]$,

$$\hat{h}(k_x, k_y) = \sum_{i_1=0}^{N_x-1}\sum_{i_2=0}^{N_y-1} h(i_1\ell, i_2\ell) e^{-i(k_x i_1 \ell + k_y i_2 \ell)}\,, \tag{2.4.35}$$

with the backward Fourier transform evaluated by

$$h(x, y) = \sum_{i_1=0}^{N_x-1}\sum_{i_2=0}^{N_y-1} \hat{h}(k_{i_1}, k_{i_2}) e^{i(k_{i_1}x + k_{i_2}y)}\,. \tag{2.4.36}$$

Applying the forward and backward Fourier transforms multiplies the input data by a factor of N_xN_y. In the case of $N_x = N_y = N$, we find in terms of k_x and k_y the factors needed for the height fluctuations computed by FFTW are

$$\langle|\delta h_k|^2\rangle_{FFTW} = \frac{N^4}{\beta L^2(k_b k_{x,y}^4 + r k_{x,y}^2)}\,, \tag{2.4.37}$$

where $k_{x,y}^2 = k_x^2 + k_y^2$. In practice we use integer wave vectors such that $k_x \in [0, N_x - 1]$ and $k_y \in [0, N_y - 1]$. The proper factors to compare the numerically computed height fluctuations to the analytical expression in Eq. (2.4.32) is thus

$$\langle|\delta h_k|^2\rangle_{FFTW} = \frac{N^4}{\beta L^2(\left(\frac{2\pi}{L}\right)^4 k_b k_{x,y}^4 + \left(\frac{2\pi}{L}\right)^2 r k_{x,y}^2)}\,. \tag{2.4.38}$$

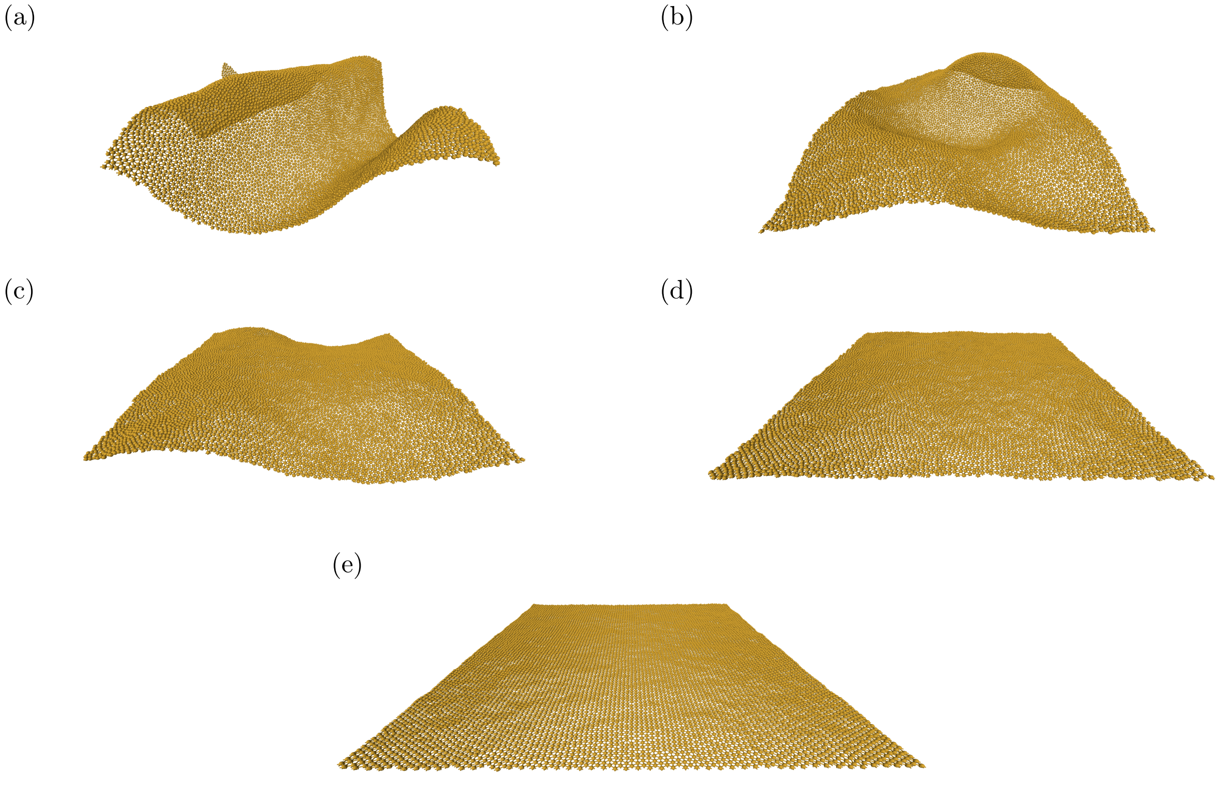


Figure 2.15: Planar periodic membrane systems with 10000 vertices, $k_\mathrm{b} = 80\ k_\mathrm{B}T$, and $\gamma = 0$ at various fixed box lengths. (a) $L = 100$. (b) $L = 110$. (c) $L = 120$. (d) $L = 130$. (e) $L = 140$.

We now evaluate the fluctuation spectrum of a planar periodic system with fixed boundaries using the Gompper-Kroll model. Images of such systems for various box lengths are found in

Fig. 2.15. At small box lengths, the membrane system develops large wrinkles due to the excess area of the membrane relative to the projected area (Figs. 2.15a and 2.15b), while at larger box sizes the membrane is held taut (Fig. 2.15e). Figure 2.16(a) shows the average area of such systems at varying γ, with the smaller box lengths having a noticeable change in the area at larger γ values while the larger box lengths have relatively constant area across γ values.

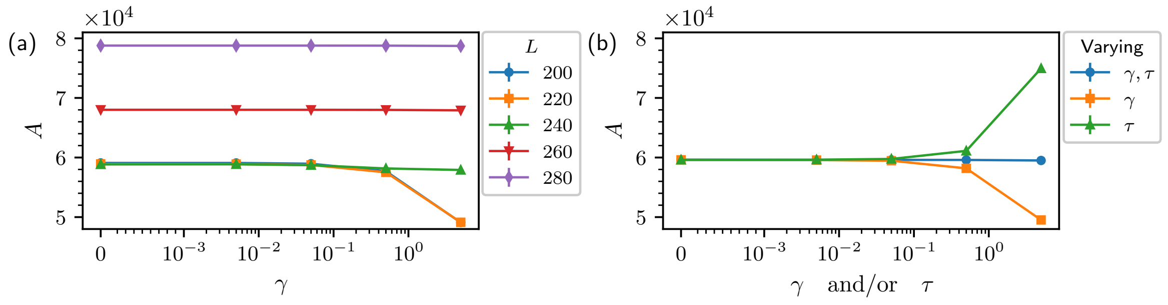


Figure 2.16: Average area of a planar periodic membrane system with 40000 vertices, $k_\text{b} = 80\ k_\text{B}T$, and varying surface tension with the Gompper-Kroll model. (a) The average area for fixed box length. (b) The average area for nonfixed box length.

Figure 2.17 shows the height fluctuation spectrums obtained in simulation for varying fixed box lengths and surface tensions. The spectrums at low box sizes have spurious height fluctuations for the lower wave vectors. For all box sizes, the spectrums saturate at high wave vectors due to protrusions [73, 74]. Estimates of the bending rigidity and the surface tension r are obtained by fitting Eq. (2.4.38) to the spectrum as shown in Tables 2.1 and 2.2. The bending rigidity estimates differ from the true values for the lower box sizes of 200 and 220 for low tensions, with increasing agreement at larger box sizes and surface tensions. The surface tension r estimates follow a similar pattern for the lower box sizes, except with the larger box sizes of 260 and 280 having a measured nonzero surface tension for low values of γ.

We now simulate the Gompper-Kroll model in a planar periodic system at nonfixed box sizes. This is done in the case of varying γ at $\tau = 0$, varying τ at $\gamma = 0$, and varying γ and τ for $\gamma = \tau$ in order to test the effects of the surface tension and frame tension on the measured k_b and r. Figure 2.16(b) shows that the average area of the membrane decreases with varying γ, as this setting penalizes area, increases with varying τ, as this setting promotes projected area, and is constant for $\gamma = \tau$.

Figure 2.18 shows the fluctuation spectrums obtained with the previous combinations of varying surface tension and frame tension. The spectrums from simulations with varying frame tension are found to have noticeable contributions from surface tension at low wave vectors, while the spectrums from varying γ at $\tau = 0$ have no such behavior. This is quantified by fitting Eq. (2.4.38) to the data, yielding the data in Tables 2.3 and 2.4. Relatively accurate estimates of k_b are obtained in all cases, though better agreement is seen in the cases of

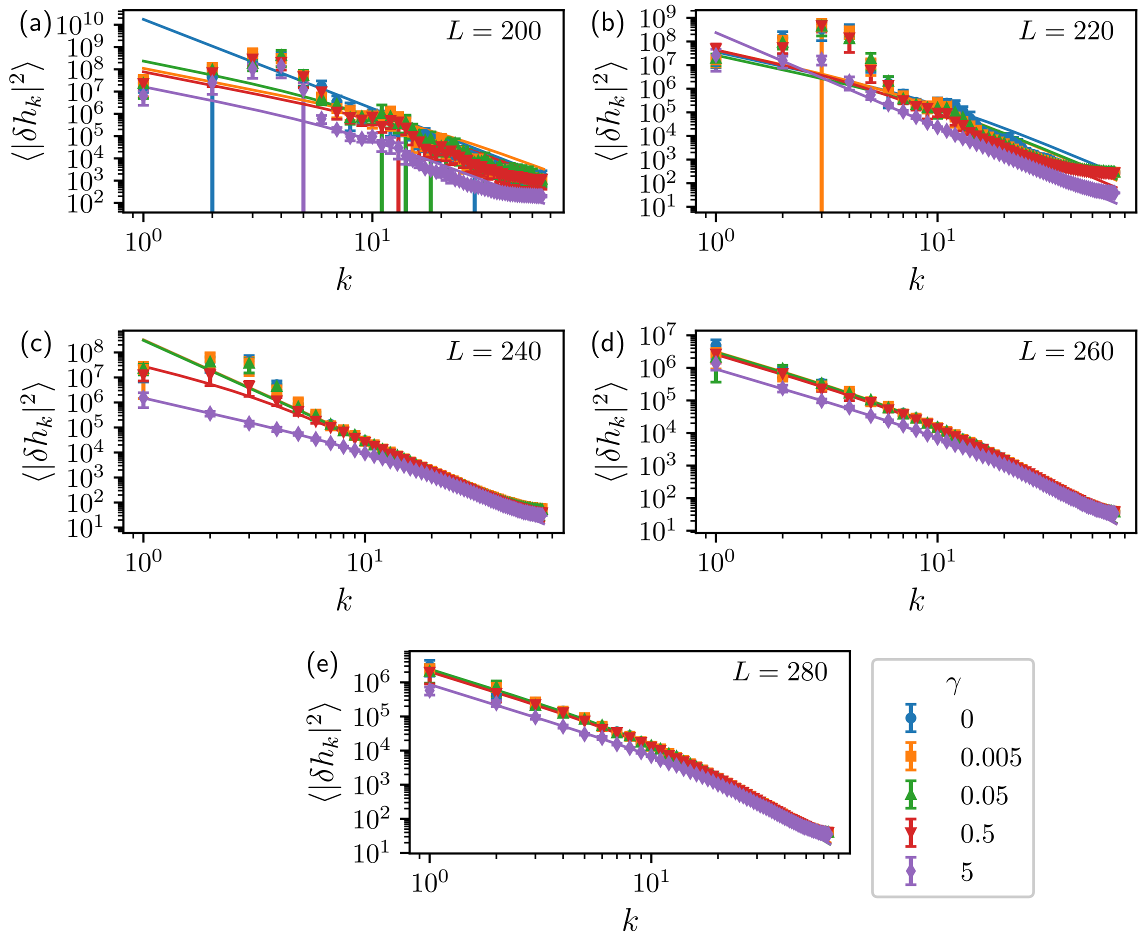


Figure 2.17: Height fluctuations for a planar periodic membrane using the Gompper-Kroll model with $k_\mathrm{b} = 80\ k_\mathrm{B}T$ and varying γ at fixed box lengths for 40000 vertices and $N_x = N_y = 128$.

L	$\gamma = 0$	$\gamma = 0.005$	$\gamma = 0.05$	$\gamma = 0.5$	$\gamma = 5$
$L = 200$	(5 ± 1)e-1	(2 ± 1)e-1	(8 ± 3)e-1	(5 ± 2)e-1	(8 ± 2)e0
$L = 220$	(4 ± 1)e0	(7 ± 2)e0	(6 ± 1)e0	(1.5 ± 0.1)e0	(7.1 ± 0.1)e1
$L = 240$	(6.2 ± 0.1)e1	(6.2 ± 0.2)e1	(6.5 ± 0.2)e1	(6.9 ± 0.1)e1	(8.1 ± 0.1)e1
$L = 260$	(7.8 ± 0.1)e1	(8.0 ± 0.2)e1	(8.1 ± 0.1)e1	(8.25 ± 0.07)e1	(7.9 ± 0.1)e1
$L = 280$	(8.0 ± 0.3)e1	(7.6 ± 0.2)e1	(8.7 ± 0.5)e1	(8.3 ± 0.6)e1	(8.1 ± 0.2)e1

Table 2.1: k_b values obtained from the fitting procedure corresponding to data in Fig. 2.17.

L	$\gamma = 0$	$\gamma = 0.005$	$\gamma = 0.05$	$\gamma = 0.5$	$\gamma = 5$
$L = 200$	(0 ± 2)e-3	(3.6 ± 0.8)e-2	(1.7 ± 0.8)e-2	(5 ± 1)e-2	(2.4 ± 0.5)e-1
$L = 220$	(1.8 ± 0.3)e-1	(1.5 ± 0.3)e-1	(2.6 ± 0.3)e-1	(1.5 ± 0.3)e-1	(0 ± 2)e-2
$L = 240$	(0 ± 2)e-2	(0 ± 6)e-2	(0 ± 4)e-2	(2.2 ± 0.2)e-1	(4.47 ± 0.02)e0
$L = 260$	(2.26 ± 0.03)e0	(2.16 ± 0.07)e0	(2.22 ± 0.03)e0	(2.62 ± 0.01)e0	(7.40 ± 0.05)e0
$L = 280$	(2.84 ± 0.05)e0	(3.01 ± 0.03)e0	(2.76 ± 0.07)e0	(3.30 ± 0.07)e0	(7.83 ± 0.05)e0

Table 2.2: r values obtained from the fitting procedure corresponding to data in Fig. 2.17.

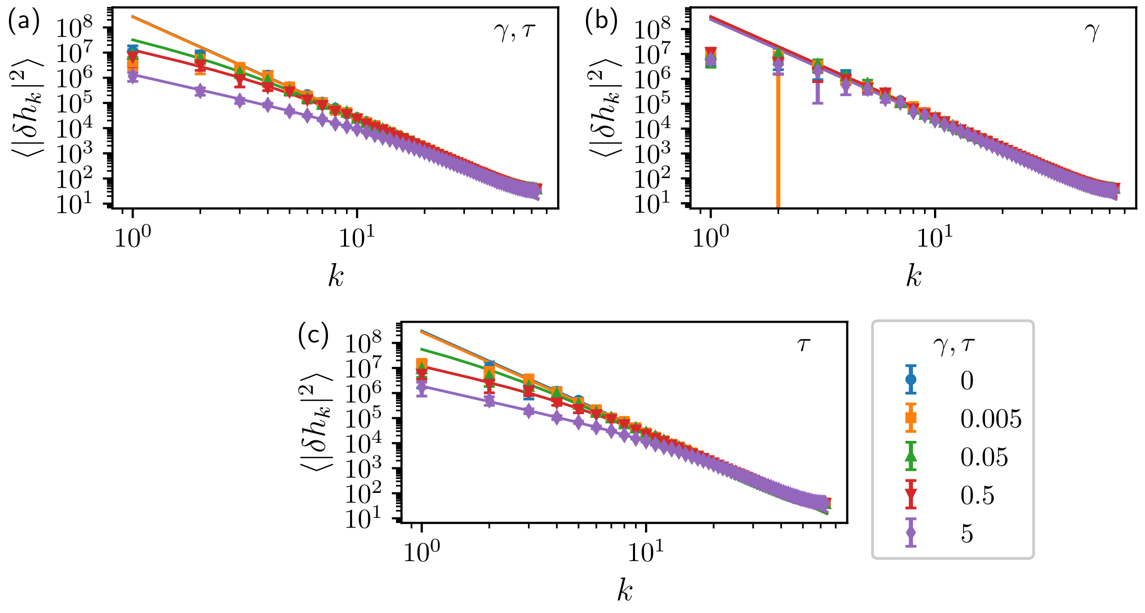


Figure 2.18: Height fluctuations for a planar periodic membrane using the Gompper-Kroll model with $k_\mathrm{b} = 80\ k_\mathrm{B}T$ and varying various surface tensions at nonfixed box size for 40000 vertices and $N_x = N_y = 128$.

Varied value	0	0.005	0.05	0.5	5
γ, τ	(7.5 ± 0.2)e1	(7.8 ± 0.2)e1	(8.1 ± 0.2)e1	(7.7 ± 0.1)e1	(7.8 ± 0.1)e1
γ	(6.6 ± 0.2)e1	(6.7 ± 0.3)e1	(7.7 ± 0.1)e1	(6.2 ± 0.6)e1	(7.0 ± 0.8)e1
τ	(6.8 ± 0.2)e1	(7.5 ± 0.1)e1	(8.0 ± 0.1)e1	(7.0 ± 0.2)e1	(8.1 ± 0.2)e1

Table 2.3: k_b values obtained from the fitting procedure corresponding to data in Fig. 2.18.

Varied value	0	0.005	0.05	0.5	5
γ, τ	(0 ± 3)e-2	(0 ± 2)e-2	(1.8 ± 0.2)e-1	(5.1 ± 0.2)e-1	(5.02 ± 0.03)e0
γ	(0 ± 3)e-2	(0 ± 4)e-2	(0 ± 2)e-2	(0 ± 8)e-2	(0 ± 4)e-2
τ	(0 ± 1)e-2	(0 ± 8)e-3	(1.0 ± 0.1)e-1	(5.6 ± 0.2)e-1	(3.59 ± 0.03)e0

Table 2.4: r values obtained from the fitting procedure corresponding to data in Fig. 2.18.

varying τ at $\gamma = 0$ and varying γ and τ. Varying γ at $\tau = 0$ yields r values near 0 for all γ values. Varying τ at $\gamma = 0$ and varying γ and τ produces non-zero r values, with the case of varying γ and τ producing the best agreement. These results demonstrate that the surface tension in the fluctuation analysis r is equivalent to τ in the discrete membrane models we consider and not γ as one would expect. This is in agreement with previous theoretical [75–78] and numerical [79] studies.

Chapter 3

Lipid Rafts

3.1 Introduction to the Multicomponent Nature of Biological Membranes

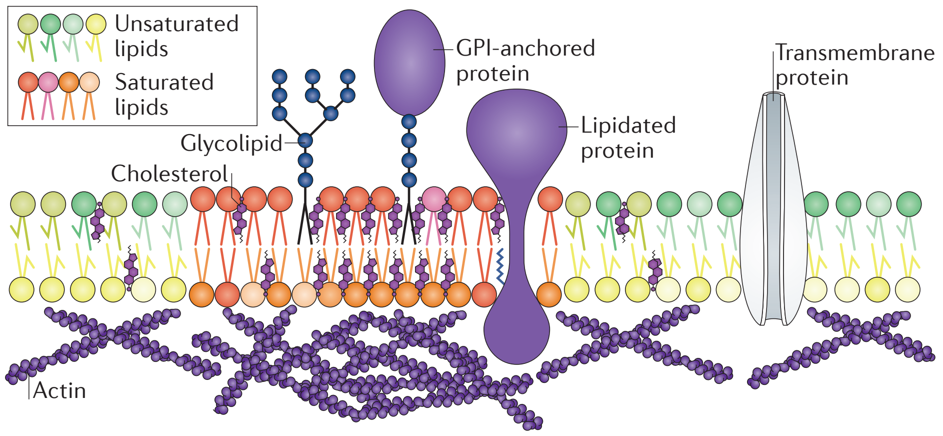


Figure 3.1: Schematic of a lipid raft, in which small, liquid-ordered domains composed of saturated lipids and cholesterol form in an otherwise liquid-disordered domain of unsaturated lipids. Proteins will have different affinities for domain types, giving lipid rafts the capability of clustering proteins involved in processes such as cell signaling. Reprinted by permission from Springer Nature Customer Service Centre GmbH: Springer Nature, *Nature Reviews Molecular Cell Biology*, "The mystery of membrane organization: composition, regulation and roles of lipid rafts", E. Sezgin et al., Copyright Nature Publishing Group (2017) [80]

The fluid mosaic model, when first developed [4], hypothesized that biological membranes were composed of lipid molecules forming a bilayer with randomly distributed proteins. One of the many refinements to the fluid mosaic model, the lipid raft hypothesis [6], allows for heterogeneous phase behavior within the membrane. In the lipid raft hypothesis proteins and

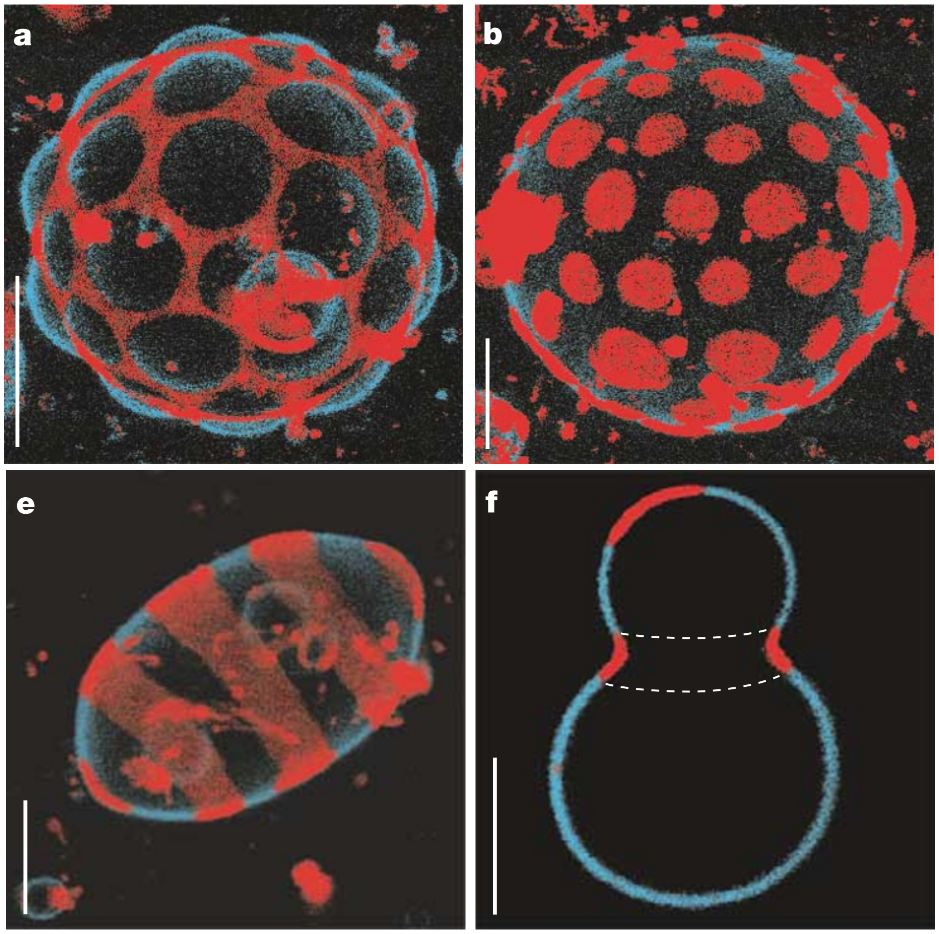


Figure 3.2: Two-photon fluorescence microscopy images of giant unilamellar vesicles composed of sphingomyelin, DOPC, and cholesterol in which microphase separation is observed. Reprinted by permission from Springer Nature Customer Service Centre GmbH: Springer Nature, *Nature*, "Imaging coexisting fluid domains in biomembrane models coupling curvature and line tension", T. Baumgart et al., Copyright Macmillan Magazines (2003) [81].

lipids form liquid-ordered (L_o) nanoscopic domains enriched in saturated lipids and cholesterol in an otherwise liquid-disordered (L_d) lipid membrane enriched in unsaturated lipids (see Fig. 3.1 for a schematic). This hypothesis has importance in many cellular phenomena such as signaling and cellular trafficking due to the ability of lipid rafts to cluster select lipids and proteins [6, 82–85]. Indirect evidence of lipid rafts was first observed through the use of detergents preferentially avoiding the hypothesized lipid raft domains of a cell membrane [5], and more indirect evidence has been found through other experimental techniques such as Förster resonance energy transfer [86, 87], measurement of molecular diffusion by stimulated emission depletion microscopy [88, 89], mass spectrometry [90–92], and single particle tracking [93, 94]. Despite this, lipid rafts remain poorly understood as their small length and time scales, corresponding to being on the order of $10-200$ nanometers and less than a second respectively [95], cause direct observation to be difficult in live systems. In comparison, the existence of a first-order phase transition between L_o–L_d phases is well understood in artificial lipid membrane systems that possess multiple lipid species and proteins due to the larger domain sizes in these systems that allow direct observation of the domains with optical microscopy, with an example shown in Fig. 3.2 [81, 96–99]. While recent cryogenic electron microscopy studies have shown probe-free observation of phase separation in model systems on the necessary spatial scale [100, 101], this technique has not yet been applied to live cells for direct observation of lipid raft domains. In this chapter, we will further develop the models described in Chapter 2 to elucidate the properties of lipid rafts by addressing their biophysical properties that can lead to their formation.

To extend the previously described membrane models we will consider the nature of a multicomponent lipid membrane along with the interaction of proteins with lipids. The behavior of multicomponent lipid membranes is well described by modifying the Helfrich model to allow for compositional dependence in the bending parameters along with compositional interactions between L_o and L_d domains. Proteins can have many potential interactions with their surrounding lipid environment [102, 103], but we will focus on an effect caused by the difference between a transmembrane protein's length and the membrane thickness. In a previous coarse-grained molecular dynamics study of S. Katira *et. al* [7], the behavior of model proteins with set protein lengths in a single component lipid membrane demonstrated that the proteins are capable of controlling local phase behavior, which is of relevance to the heterogeneity described in the lipid raft hypothesis. The DPPC membrane used in the study has a first-order phase transition between a liquid-disordered state (Fig. 3.3a) and a solid-ordered state (Fig. 3.3b) with the two phases having different membrane thicknesses. By inserting a protein with a length matching the thermodynamically disfavored phase's thickness, the protein leverages pre-transition phenomena by inducing a local region of the thermodynamically disfavored phase (Fig. 3.3c). The interface formed by this region was found to have identical line tension to that of a bulk interface despite the region being nanoscopic. The interface formation creates a free energy incentive for the assembly of proteins to reduce the interfacial perimeter, and it was demonstrated in a two-protein system that assembly occurs on the physiological timescale of microseconds as shown in Fig. 3.3d. This behavior was termed the orderphobic effect due to its similarity to the hydrophobic

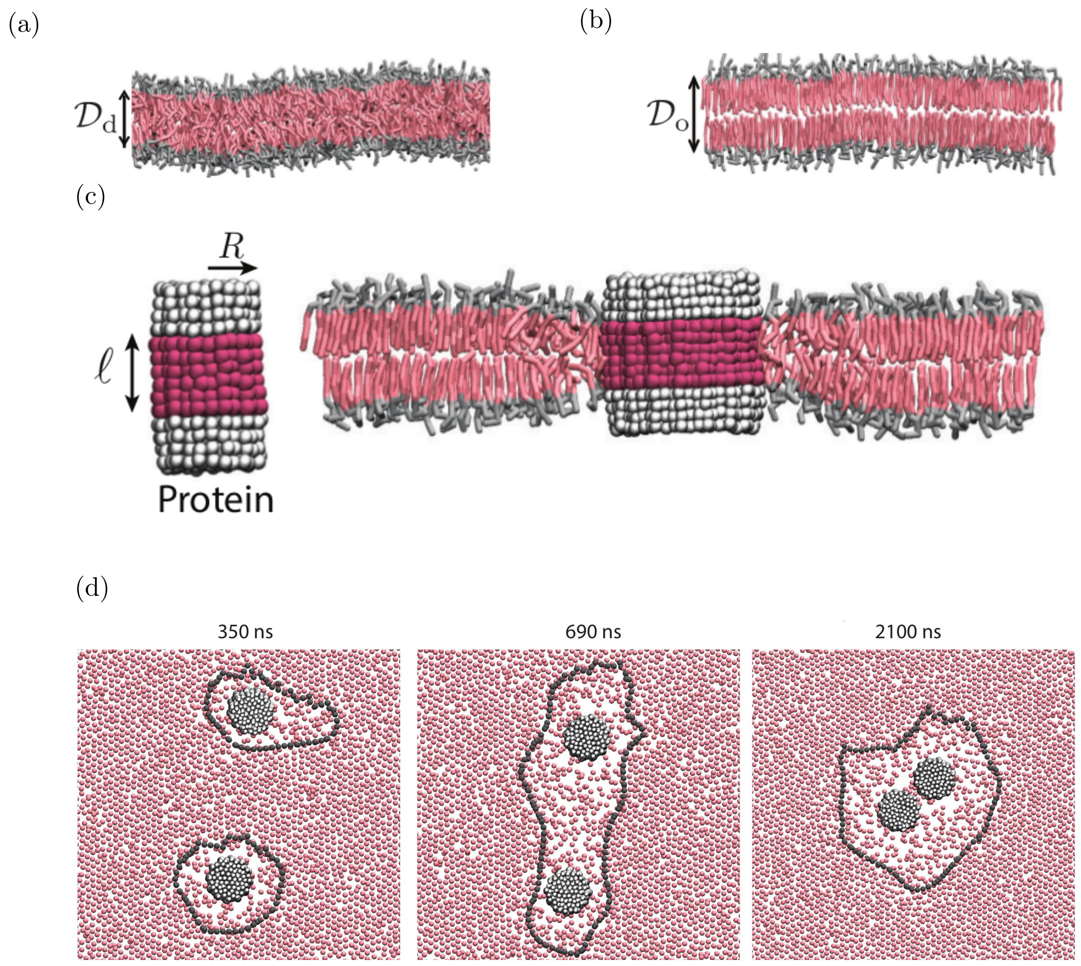


Figure 3.3: Overview of the orderphobic effect. Adapted from S. Katira et al., "Pre-transition effects mediate forces of assembly between transmembrane proteins", eLife **5**, e13150 (2016), licensed under CC0 1.0 `https://creativecommons.org/publicdomain/zero/1.0/`. (a) DPPC in the liquid-disordered state. (b) DPPC in the solid-ordered state. (c) An orderphobic protein, as set by its hydrophobic thickness, induces a disordered domain around it. (d) The assembly of two orderphobic proteins on the timescale of microseconds. The snapshot is a top view of the tail-end particles of each lipid in one monolayer.

effect, in which hydrophobic solutes assemble on large length scales because of similar free energy incentives driven by surface tension [104]. Though the orderphobic effect has not been experimentally verified, protein length has been identified experimentally as a key parameter for protein domain preference [103, 105, 106].

The orderphobic effect provides a mechanism for the macroscopic assembly of proteins driven by line tension, but as previously described lipid rafts are believed to exhibit microphase behavior in which they exist in distinct nanoscopic domains. In this chapter, we will resolve this paradox by considering the additional tunable parameters in a multicomponent lipid membrane with protein system. A computational model is developed in Section 3.2 based on the Gompper-Kroll model discussed in Section 2.3.3 that accounts for the energetics of a multicomponent lipid membrane in terms of both bending and compositional energies along with the effect of proteins that are either orderphobic (preferring the L_d phase) or orderphilic (preferring the L_o phase). To perform studies in which phase diagrams and the energetics of protein domains are evaluated, the basics of free energy sampling using low-dimensional representations of a configuration, termed collective variables (CVs), are discussed in Section 3.3. Using these tools the compositional and elastic interactions of protein domains are quantified in Sections 3.4 and 3.5, respectively, to yield a mechanistic understanding of lipid rafts existing due to the competition of line tension, which drives macrophase behavior, with the spontaneous curvature of induced domains, which drives microphase behavior. We conclude with further discussions and potential directions for future work in Section 3.6.

3.2 Modeling Multicomponent Membranes

We extend the modeling framework of Sections 2.2 and 2.3 to develop a computational model that describes the physics of multicomponent membranes with proteins on large length and time scales. The main contributions to the membrane-protein energetics on these scales are from the composition energy and the bending energy, with the effect of the proteins being handled by an adaptation of the orderphobic effect. Over long length scales the membrane-protein system can be represented by a two-dimensional surface in three-dimensional space, in which we will use the triangulated surface framework of Section 2.3 for the computational model. On the surface, the energy of the system V is decomposed into

$$V = V_c + V_b \,, \tag{3.2.1}$$

where V_c and V_b are the compositional and bending energetics, respectively. We will now describe the energetics in detail.

For the compositional energetics, it has been found experimentally that the behavior of coexisting L_o–L_d phases is described by the two-dimensional Ising universality class [107]. Thus, we use the Ising model to describe the compositional energetics of the system [108]. Within the framework of a triangulated surface, each vertex will represent a region of the membrane consisting of either the L_o phase, L_d phase, or the protein-enriched phase. The composition of a vertex is denoted by a scalar, ϕ_i, that returns different values depending on the identity of the vertex, which is given by

$$\phi_i = \begin{cases} -1, & \text{if } \mathrm{L_d} \text{ or orderphobic protein vertex}\,, \\ 1, & \text{if } \mathrm{L_o} \text{ or orderphilic protein vertex}\,. \end{cases} \tag{3.2.2}$$

The form of V_c is that of the Ising model [109]

$$V_\mathrm{c} = -\frac{J}{2}\sum_i \sum_{j(i)} \phi_i \phi_j - \mu \sum_i \phi_i\,, \tag{3.2.3}$$

where J is the coupling parameter and μ is the chemical potential.

The bending energetics are described by the Gompper-Kroll model of Section 2.3.3, which will be modified to account for multicomponent behavior along with a slight modification to make the model in line with what is more commonly used in the literature. This slight modification is the calculation of H_i as

$$H_i = \left[\frac{1}{\sigma_i}\sum_{j(i)} \frac{\sigma_{ij}}{\ell_{ij}} \left(\mathbf{x}_i - \mathbf{x}_j\right)\right] \cdot \mathbf{n}_i\,, \tag{3.2.4}$$

which is twice that of the form used in Eq. (2.3.89) and in line with the forms found in the literature [38, 110]. To construct a version of the Helfrich model for multicomponent lipids, it is noted that membrane bending parameters of k_b and γ can vary between the $\mathrm{L_o}$ and $\mathrm{L_d}$ phases [81]. The membrane bending parameters are also dependent on if a vertex is a protein vertex, which is labeled with the variable n_i such that

$$n_i = \begin{cases} 0, & \text{if not a protein vertex}\,, \\ 1, & \text{if protein vertex}\,. \end{cases} \tag{3.2.5}$$

The bending energy V_b is then given by a modified version of the single component version of Eq. (2.3.1) [111, 112]

$$V_\mathrm{b} = \sum_i \Delta a_i \big[k_\mathrm{b}\left(\phi_i, n_i\right)\left(H_i - C\left(\phi_i, n_i\right)\right)^2 + \gamma\left(\phi_i, n_i\right)\big] - \tau A_p\,, \tag{3.2.6}$$

where Δa_i is the Voronoi cell area of a vertex given by Eq. (2.3.86), $k_\mathrm{b}(\phi_i, n_i)$ is the composition-dependent bending rigidity, H_i is the mean curvature given by Eq. (3.2.4), $C(\phi_i, n_i)$ is the composition-dependent spontaneous curvature, and $\gamma(\phi_i, n_i)$ is the composition-dependent surface tension. It is assumed that the topology of the membrane is constant and the Gaussian moduli for the $\mathrm{L_o}$–$\mathrm{L_d}$ phases are equal, fixing the Gaussian curvature contribution in the Helfrich model per the Gauss-Bonnet theorem [36].

The relevant length scale and value of parameters for simulation are now discussed. The fundamental length scale of the system is set by the hard sphere size σ, which is at least greater than the membrane thickness (O(5 nm)) for the surface approximation of a lipid bilayer to hold. The contributions in the energy due to mean curvature and the Ising model are length scale-independent, while surface and frame tension are length scale-dependent. However, while the Ising model parameters are length scale-independent, the Ising model below the critical temperature, in which phase separation occurs, generates a length scale-dependent line tension, λ. Physiologically the values of γ and τ are between $10^{-4} - 1\ k_{\mathrm{B}}T/\mathrm{nm}^2$ [32], and the value of λ is between $0\ k_{\mathrm{B}}T/\mathrm{nm}$ at the critical temperature and $O(1)\ k_{\mathrm{B}}T/\mathrm{nm}$ at physiological temperatures [113, 114]. We take $\gamma = \tau$ to enforce approximate surface incompressibility across varying surface tension values as seen in Fig. 2.16(b). Values of the spontaneous curvature of a monolayer vary between near $0\ \mathrm{nm}^{-1}$ to $1\ \mathrm{nm}^{-1}$ [115–117]. The spontaneous curvature of a bilayer is set by the difference between the inner and outer leaflet spontaneous curvatures, with only asymmetric membranes having a non-zero spontaneous curvature [116]. Spontaneous curvature can also be driven by proteins due to a variety of mechanisms such as shape, insertion, scaffolding, or inducing locally asymmetric compositions [116, 118–120]. The value of the bending rigidity for a $\mathrm{L_d}$ system is around $20\ k_{\mathrm{B}}T$ [121], while the addition of cholesterol to create a $\mathrm{L_o}$ system can increase the bending rigidity to $100\ k_{\mathrm{B}}T$ with typically values being around $60\ k_{\mathrm{B}}T$ [122–127]. We use values of $k_{\mathrm{b}}(L_{\mathrm{d}}) = 20\ k_{\mathrm{B}}T$ and $k_{\mathrm{b}}(L_{\mathrm{o}}) = 60\ k_{\mathrm{B}}T$ throughout the rest of this chapter unless specified otherwise. Similarly, we set the spontaneous curvature of the $\mathrm{L_o}$–$\mathrm{L_d}$ vertices to be zero and vary the spontaneous curvature of the protein vertices, labeled as $C = C\left(\phi_i = 1, n_i = 1\right)$, unless specified otherwise.

This triangulated surface model is used in Monte Carlo simulations with the Metropolis algorithm of Eq. (2.3.11). The previously described displacement, edge flip, and box resizing moves of Section 2.3.1 will be used along with additional moves to evolve ϕ_i and n_i. Note that in all cases we consider the protein vertices to be all of one type with the number of protein vertices fixed, and as such all Monte Carlo moves will conserve the number of protein vertices. The moves that evolve ϕ_i and n_i are as follows.

1. *Mass conserving move on ϕ_i or n_i*: a vertex i is selected at random, and its identity (either ϕ_i or n_i) is swapped with another vertex. The other vertex can either be selected locally by randomly selecting from vertices that share an edge with vertex i, or selected non-locally by randomly selecting another vertex that is not vertex i. If vertex i and the other vertex have the same identity, the move is automatically accepted as the energy does not change and the transition probabilities are equal, as will now be shown. Labeling the other vertex as j, the move can be reversed by performing another mass conserving move by selecting vertex i and then vertex j or vertex j and then vertex i. As such, the transition probabilities are equal to each other and this leads to the acceptance probability factor in Eq. (2.3.11) being $\exp(-\beta\left(V_\mu - V_\nu\right))$.

2. *Non-mass conserving move on ϕ_i*: a vertex i is selected at random and then ϕ_i is changed to the opposite value. If vertex i is a protein vertex, the move is automatically rejected. As the vertex is selected at random, the transition probabilities are equal to each other and hence the acceptance probability factor in Eq. (2.3.11) is $\exp(-\beta\left(V_\mu - V_\nu\right))$.

3. *Non-mass conserving move on ϕ_i with local mass conserving moves on n_i*: a vertex i is selected at random, and ϕ_i is changed to the opposite value. If vertex i is not a protein vertex, it proceeds as in the case of a standard non-mass conserving move on ϕ_i. If instead vertex i is a protein vertex, an exchange of n_i with a randomly selected vertex sharing an edge with vertex i is attempted instead. As the vertex i and its neighbor j are selected at random, the reverse move requires the vertex j to be randomly selected and then vertex i to be randomly selected out of the edges of vertex j. This leads to the ratio of the transition probabilities being

$$\frac{T_{\mu\nu}}{T_{\nu\mu}} = \frac{\frac{1}{N\mathcal{N}_j}}{\frac{1}{N\mathcal{N}_i}} = \frac{\mathcal{N}_i}{\mathcal{N}_j} \,. \tag{3.2.7}$$

   This move allows for the simultaneous evolution of non-mass conserved ϕ_i with the evolution of mass conserved n_i.

The Monte Carlo moves on the compositional variables are extendable to the parallel checkerboard Monte Carlo scheme of Section 2.3.1.1. For the local moves, a similar constraint to the displacement and edge flip moves is used with identity exchange attempts between vertices in different subdomains being automatically rejected. For the non-local moves, this is extended by randomly pairing subdomains of one type together. The non-local moves are then made between vertices of the paired subdomains.

3.3 Methods to Compute Free Energies

Throughout the rest of this thesis, we will be interested in processes with free energy barriers that lead to a slow sampling of configuration space using standard importance sampling methods. Correspondingly, averages computed through Eq. (2.3.2) are likely inaccurate due to the inefficiency of sampling, with potentially prohibitive amounts of simulation time being needed for high accuracy. To improve upon this we will use umbrella sampling (US) [128–130]. In umbrella sampling, many independent simulations, termed replicas, run in parallel with different bias potentials to efficiently sample the wanted configuration space. Enumerating the replicas with indexing variable $\alpha \in \{1, \ldots, M\}$, the bias potential for each replica $W_\alpha(\mathbf{x})$ is

$$V_\alpha(\mathbf{x}) = V(\mathbf{x}) + W_\alpha(\mathbf{x}) \,. \tag{3.3.1}$$

The bias potentials take a form that makes the system explore the collective variables of interest, such as the average ϕ value or the distance between two protein domains. Denoting the collective variables as $\mathbf{\Theta}(\mathbf{x})$, the biasing potentials used are that of harmonic potentials

$$W_\alpha(\mathbf{x}) = \frac{\kappa_\alpha}{2}\left(\mathbf{\Theta}(\mathbf{x}) - \mathbf{\Theta}_\alpha\right)^2 \,, \tag{3.3.2}$$

where κ_α and $\mathbf{\Theta}_\alpha$ are the biasing strength and target CV values for replica α, respectively. The biasing potential allows for the sampling of values of $\mathbf{\Theta}$ that are uncommon in unbiased simulations, with the values of κ_α and $\mathbf{\Theta}_\alpha$ chosen such that values near $\mathbf{\Theta}_\alpha$ for all α are adequately sampled and replicas achieve overlap in the obtained distributions of $\mathbf{\Theta}$ with neighboring replicas. As the potentials bias the simulation results, an unbiasing scheme is required to compute unbiased averages or to obtain the probability of observing certain values of $\mathbf{\Theta}$ in an unbiased simulation. We will find that the factors necessary to perform this unbiasing procedure are related to the free energies of the replicas. In what follows, we describe a method to compute the free energies, the multistate Bennett acceptance ratio (MBAR) [131, 132]. We follow the presentation of Ref. [132] to provide a self-contained introduction to MBAR.

To begin, we define the free energy of replica i as

$$F_i = -\beta_i^{-1} \ln Q_i \,, \tag{3.3.3}$$

where β_i is the inverse temperature of replica i and Q_i is the partition function of replica i per Eq. (2.3.9). It is difficult to compute F_i in simulation via evaluating the partition function via Eq. (2.3.2). An easier method is to evaluate the free energy difference between replica i and another replica j, which can be done through the reweighting of samples obtained in one replica to the other replica. To do so, denoting the probability distribution functions of replicas i and j as $\rho_i(\mathbf{x})$ and $\rho_j(\mathbf{x})$, respectively, we note that in importance sampling with respect to replica j samples are generated as $\mathbf{x} \sim \rho_j(\mathbf{x})$. To reweight averages from samples obtained in replica j to that of replica i, one computes a modified form of Eq. (2.3.2) for an observable $\mathcal{A}$

$$\langle \mathcal{A} \rangle_i = \int \mathrm{d}\mathbf{x}\ \mathcal{A}(\mathbf{x})\rho_i(\mathbf{x}) = \int \mathrm{d}\mathbf{x}\ \mathcal{A}(\mathbf{x})\frac{\rho_i(\mathbf{x})}{\rho_j(\mathbf{x})}\rho_j(\mathbf{x}) \tag{3.3.4}$$

$$\approx \frac{1}{N}\sum_{n=1}^{N} \mathcal{A}\left(\mathbf{x}_n\right) \frac{\rho_i\left(\mathbf{x}_n\right)}{\rho_j\left(\mathbf{x}_n\right)} = \frac{1}{N}\sum_{n=1}^{N} \mathcal{A}\left(\mathbf{x}_n\right) \frac{Q_j \exp\left[-\beta_i V_i(\mathbf{x}_n)\right]}{Q_i \exp\left[-\beta_j V_j(\mathbf{x}_n)\right]} \,. \tag{3.3.5}$$

By setting $\mathcal{A}(\mathbf{x}) = 1$, one obtains

$$\frac{Q_i}{Q_j} = \frac{1}{N}\sum_{i=1}^{N} \frac{\exp\left[-\beta_i V_i(\mathbf{x}_n)\right]}{\exp\left[-\beta_j V_j(\mathbf{x}_n)\right]} \approx \langle \exp\left[-\beta_i V_i(\mathbf{x}) + \beta_j V_j(\mathbf{x})\right] \rangle_j \,, \tag{3.3.6}$$

where $\langle \ldots \rangle_j$ is an ensemble average over $\rho_j(\mathbf{x})$. From Eq. (3.3.3) the free energy difference is obtained as

$$\beta_j F_j - \beta_i F_i = -\ln \frac{1}{N}\sum_{i=1}^{N} \exp\left[-\beta_i V_i(\mathbf{x}_n) + \beta_j V_j(\mathbf{x}_n)\right] \tag{3.3.7}$$

$$\approx -\ln \langle \exp\left[-\beta_i V_i(\mathbf{x}) + \beta_j V_j(\mathbf{x})\right] \rangle_j \,, \tag{3.3.8}$$

which is known as the free-energy perturbation (FEP) method to compute free energy differences [133].

To extend this to multiple replicas we utilize a formalism known as the mixture distribution [134]. Let's assume we perform simulations in the M different replicas where we have collected N_α samples in each replica. Each replica obeys its own probability distribution $\rho_\alpha(\mathbf{x})$, and from these distributions a new probability distribution is constructed that contains all $N = \sum_{\alpha=1}^{M} N_\alpha$ samples. This distribution, the mixture distribution, is prescribed the form

$$\rho_m(\mathbf{x}) = \frac{1}{N}\sum_{\alpha=1}^{M} N_\alpha \rho_\alpha(\mathbf{x}) \ , \tag{3.3.9}$$

with the intuition that one has a N_α/N chance of getting a sample from replica α, and as each $\rho_\alpha(\mathbf{x})$ is normalized the resulting mixture distribution is also normalized. We now apply the previous reweighting trick that was used to derive Eq. (3.3.7) to determine the partition functions by reweighting from the mixture distribution to replica i. Reweighting the observable $\mathcal{A}(\mathbf{x}) = 1$ from the mixture distribution to replica i yields

$$1 = \frac{1}{N}\sum_{n=1}^{N}\frac{\rho_i(\mathbf{x}_n)}{\rho_m(\mathbf{x}_n)} = \sum_{n=1}^{N}\frac{Q_i^{-1}\exp\left[-\beta_i V_i(\mathbf{x}_n)\right]}{\sum_{\alpha=1}^{M} N_\alpha Q_\alpha^{-1}\exp\left[-\beta_\alpha V_\alpha(\mathbf{x}_n)\right]} \, , \tag{3.3.10}$$

from which we obtain an expression for Q_i,

$$Q_i = \sum_{n=1}^{N}\frac{\exp\left[-\beta_i V_i(\mathbf{x}_n)\right]}{\sum_{\alpha=1}^{M} N_\alpha Q_\alpha^{-1}\exp\left[-\beta_\alpha V_\alpha(\mathbf{x}_n)\right]} \, . \tag{3.3.11}$$

This is the MBAR estimator for free energy [131]. There are M equations for each of the partition functions, with only $M-1$ independent equations due to Eq. (3.3.11) being the same upon multiplication by a constant. By convention, we fix the partition function of the first replica to be $Q_1 = 1$. Once all of the partition functions have been obtained by solving the system of equations given by Eq. (3.3.11), an estimate of the partition function for any condition is obtained through Eq. (3.3.11), though more accurate results are obtained if the conditions of the wanted replica are well sampled with the simulated dataset. For example, this can be done in the case of the unbiased ensemble corresponding to $V(\mathbf{x})$ in which one obtains

$$Q = \sum_{n=1}^{N}\frac{\exp\left[-\beta V(\mathbf{x}_n)\right]}{\sum_{\alpha=1}^{M} N_\alpha Q_\alpha^{-1}\exp\left[-\beta_\alpha V_\alpha(\mathbf{x}_n)\right]} \, . \tag{3.3.12}$$

From Eq. (3.3.12), one sees that the unbiased weight associated with a sample $\mathbf{x}$ is

$$W(\mathbf{x}) = \frac{Q^{-1}\exp\left[-\beta V(\mathbf{x})\right]}{\sum_{\alpha=1}^{M} N_\alpha Q_\alpha^{-1}\exp\left[-\beta_\alpha V_\alpha(\mathbf{x})\right]} \, , \tag{3.3.13}$$

and the corresponding unbiased average of an observable $\mathcal{A}$ is

$$\langle A\rangle = \sum_{n=1}^{N}\mathcal{A}(\mathbf{x}_n)W(\mathbf{x}_n) \, . \tag{3.3.14}$$

While there are multiple ways to obtain the MBAR estimator of free energy, such as maximum likelihood [135, 136] and extended bridge sampling estimators [131], this particular derivation is useful in its simplicity and ease in seeing how Eqs. (3.3.10) and (3.3.11) are used for reweighting to different conditions per Eqs. (3.3.13) and (3.3.14).

Using umbrella sampling it is straightforward to evaluate the free energies of the replicas by Eq. (3.3.11). Another piece of information can be gained by evaluating the unbiased free energy with respect to a specific binned value of $\mathbf{\Theta}(\mathbf{x})$, which will be helpful in understanding the phase and shape behavior of the membrane model. Analytically this is given by the marginalization of Eq. (2.3.9) to a specific value of $\mathbf{\Theta}(\mathbf{x})$ as

$$Q\left(\mathbf{\Theta}\right) = \int \mathrm{d}\mathbf{x}\ \exp\left[-\beta V\left(\mathbf{x}\right)\right]\delta\left[\mathbf{\Theta} - \mathbf{\Theta}\left(\mathbf{x}\right)\right]\ , \tag{3.3.15}$$

$$F\left(\mathbf{\Theta}\right) = -\beta^{-1}\ln Q\left(\mathbf{\Theta}\right)\ . \tag{3.3.16}$$

To compute a discrete version of Eq. (3.3.16) from simulation data, the data is binned according to uniform discrete values of $\mathbf{\Theta}$. Let $\mathbf{\Theta}_i$ be a discrete bin of $\mathbf{\Theta}$ and $\mathcal{P}_i$ be the set of configurations $\mathbf{\Theta}(\mathbf{x}) \in \mathbf{\Theta}_i$. We estimate $F\left(\mathbf{\Theta}_i\right)$ by summing the weights given by Eq. (3.3.13) of the configurations in $\mathcal{P}_i$ to yield

$$\beta F\left(\mathbf{\Theta}_i\right) = -\ln \sum_{\mathbf{x}\in\mathcal{P}_i} W(\mathbf{x})\,. \tag{3.3.17}$$

We note that additional schemes for obtaining free energies and unbiasing are possible, and we will use a different unbiasing scheme derived in Ref. [137] for Chapter 5.

In practice, pymbar[1] is used to evaluate quantities involving MBAR [131, 138]. The timeseries module of pymbar is also used to decorrelate data obtained from simulation. This is done by evaluating the autocorrelation time of the data, which is then used to subsample the data. Bootstrapping is used to generate estimates of the mean and variance of quantities such as free energies and averaged quantities by using randomly sampled datasets generated from the decorrelated data [139, 140].

In addition to standard umbrella sampling, we also use replica-exchange umbrella sampling (REUS) [141, 142] in some cases to enhance the efficiency of sampling. In REUS we attempt to exchange configurations between replicas during sampling to enhance sampling of configuration space. This is done through a Monte Carlo move that, given randomly selected replica indices i and j from the K total replicas, accepts a swap of configurations $\mathbf{x}_i$ and $\mathbf{x}_j$ between replicas i and j with probability

$$A_{ij} = \min\left[1, \frac{\exp\left[-(\beta_i V(\mathbf{x}_j) + \beta_j V(\mathbf{x}_i))\right]}{\exp\left[-(\beta_i V(\mathbf{x}_i) + \beta_j V(\mathbf{x}_j))\right]}\right]\ . \tag{3.3.18}$$

We follow the procedure of Ref. [143] in which we attempt K^4 swaps between randomly selected pairs of replicas, and such a swap move is done infrequently compared to the parallel checkerboard move or box resizing move to allow the systems to decorrelate between swaps.

[1] https://github.com/choderalab/pymbar

3.4 Composition Effects

We first examine composition effects driven by V_c in membrane-protein systems.

3.4.1 Phase Diagrams

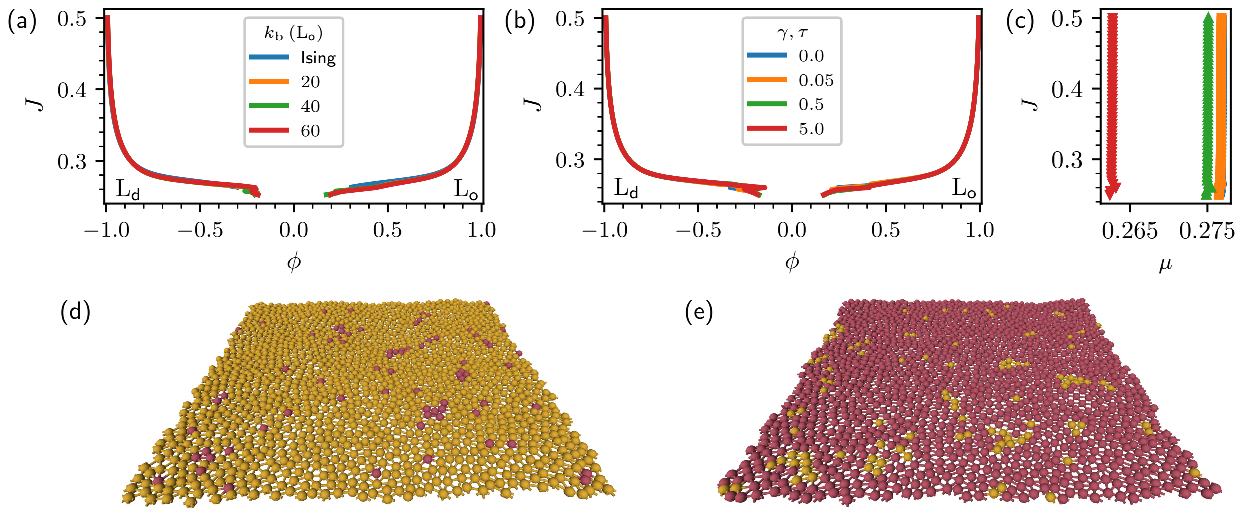


Figure 3.4: (a) Phase diagram in ϕ vs J space for variations in $k_\mathrm{b}\,(L_\mathrm{o})$ with $\gamma, \tau = 0$. (b) Phase diagram in ϕ vs J space for variations in γ, τ of both phases, with $k_\mathrm{b}\,(L_\mathrm{o}) = 60\ k_\mathrm{B}T$. (c) Phase diagram in μ vs J space for the systems in (b) with lines indicating μ_eq. The same coloring scheme is used as (b). (d) System with 1600 vertices in equilibrium in the L_d phase at $\mu_\mathrm{eq} - \mu = $ 1e-4 $k_\mathrm{B}T$, $J = 0.3\ k_\mathrm{B}T$, $k_\mathrm{b}\,(L_\mathrm{d}) = 20\ k_\mathrm{B}T$, and $k_\mathrm{b}\,(L_\mathrm{o}) = 60\ k_\mathrm{B}T$. (e) System with 1600 vertices in equilibrium in the L_o phase at $\mu_\mathrm{eq} - \mu = -$1e-4 $k_\mathrm{B}T$, $J = 0.3\ k_\mathrm{B}T$, $k_\mathrm{b}\,(L_\mathrm{d}) = 20\ k_\mathrm{B}T$, and $k_\mathrm{b}\,(L_\mathrm{o}) = 60\ k_\mathrm{B}T$.

To allow measurement of distance away from phase coexistence, phase diagrams with respect to composition and chemical potential are obtained in Fig. 3.4. This is done with US through sampling with respect to the average composition given by

$$\phi = \frac{1}{N} \sum_{i=1}^{N} \phi_i \,, \tag{3.4.1}$$

with the chemical potential at phase coexistence, μ_eq, being obtained through reweighting with MBAR. The value of μ_eq is identified by setting the probability of each phase to be equal [144, 145]. The probability of each phase is obtained by integration of $\exp(-\beta F(\phi))$, with the local free energy maxima between the phases being used to distinguish the two phases (see the blue curve in Fig. A.2(a) for reference of what the free energy with respect to ϕ is). The value of μ_eq is then evaluated using Brent's method [146]. Figures 3.4(a,b) show that the phase diagrams obtained for variations in bending rigidity and surface tension are similar to that of the Ising model on a hexatic lattice [147], reproducing the critical coupling parameter $J_c = \frac{\ln 3}{4}\, k_{\mathrm{B}}T$ along with the symmetric composition values. Differences in bending rigidity and tension between $\mathrm{L_o}$–$\mathrm{L_d}$ phases lead to nonzero μ_{eq} due to the $\mathrm{L_d}$ having a lower bending energy (Fig. 3.4(c)). Reference images of the $\mathrm{L_d}$ and $\mathrm{L_o}$ phases are given in Fig. 3.4(d) and Fig. 3.4(e), respectively.

A discussion on the relation of the line tension λ to simulation parameters is needed. There are two ways to tie the value of J to relevant experimental parameters. The first is setting the critical temperature given by $T_c = \frac{4}{k_{\mathrm{B}} \ln 3} J_c$ to experimental values and then tuning temperature appropriately to yield the line tension λ. The other is through tuning λ directly by setting J at constant T as is done in Figs. 3.4(a–c). For the Ising model on a hexatic lattice, an analytical formula relating λ, J, and T has been derived. The formula is per Ref. [147]

$$\beta\lambda = 2 \ln \sinh 2K + 2 \ln \left[\frac{1}{2} \left(1 + v u_1\right)^{0.5} + \frac{1}{2} \left(1 + v u_2\right)^{0.5} \right] , \tag{3.4.2}$$

where

$$K = \beta J \,, \tag{3.4.3}$$
$$v = \tanh K \,, \tag{3.4.4}$$
$$u_1 = v + 2 \,, \tag{3.4.5}$$
$$u_2 = v + v^2 + v^{-2} \,. \tag{3.4.6}$$

For $\beta = 1$, Eq. (3.4.2) yields $\lambda\left(J = 0.3\right) \approx 0.17\ k_{\mathrm{B}}T\sigma^{-1}$, $\lambda\left(J = 0.5\right) \approx 1.34\ k_{\mathrm{B}}T\sigma^{-1}$, and $\lambda\left(J = 1.0\right) \approx 3.75\ k_{\mathrm{B}}T\sigma^{-1}$. In the first approach, in order to see λ range on physiological values, the temperature would need to be lowered by a factor of $\frac{k_{\mathrm{b}} \ln 3}{4}$ which would yield unphysiological temperatures. Hence, we use the second approach of tuning λ through J.

3.4.2 Single Protein Effects

We now probe the effect of a single protein domain. The protein domain is constructructed by initializing protein vertices within a confining domain, such that displacement and identity exchange moves are allowed within the domain but not out of the domain. The confining radius R is taken to be 1.5 times greater than the initial cluster's radius. This allows sufficient room for protein vertices to move. An example of a single orderphilic protein domain is given in Fig. 3.5(a), with the confining radius given in black.

To see if a model orderphilic protein domain nucleates the $\mathrm{L_o}$ phase in its vicinity, we evaluate the composition field as a function of distance from the center of the protein, $\langle\phi\left(r\right)\rangle$, with the protein vertices counted as $\mathrm{L_o}$ vertices in the evaluation of $\langle\phi\left(r\right)\rangle$. This quantity is evaluated for a variety of chemical potentials, protein radii, and coupling parameter values in Fig. 3.5(b–d). The extent of the region nucleated by the protein increases with the proximity to μ_{eq}, the size of the protein, and proximity to J_c. This behavior replicates that found in the previous orderphobic effect study that used a coarse-grained molecular dynamics model [7].

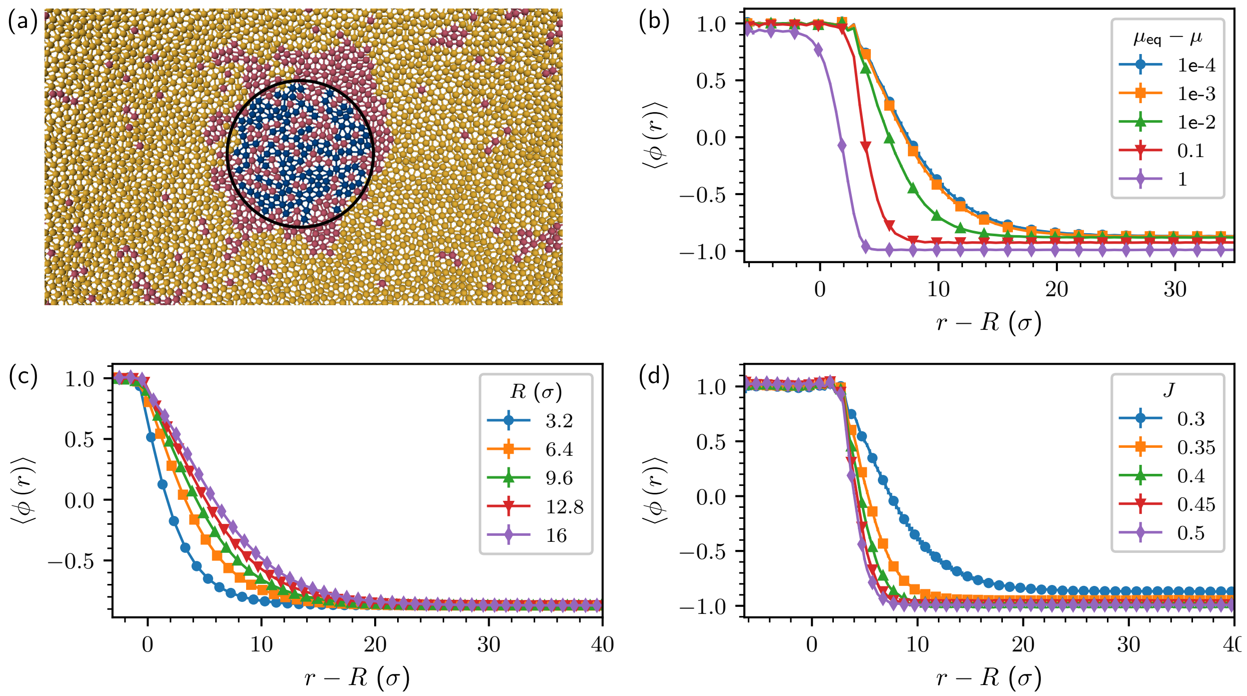


Figure 3.5: (a) Top view of a single orderphilic protein domain of radius 9.6 σ at $k_b(L_d) = 20\ k_BT$, $k_b(L_o) = 60\ k_BT$, $J = 0.3\ k_BT$, and $\mu_{eq} - \mu = $ 1e-4 k_BT with 10^4 total vertices. The confining radius is given in black. (b) Varying μ for a protein of radius 6.4 σ at $J = 0.3\ k_BT$. (c) Varying protein radius for $J = 0.3\ k_BT$ and $\mu_{eq} - \mu = $ 1e-4 k_BT. (d) Varying J for a protein of radius 6.4 σ and $\mu_{eq} - \mu = $ 1e-4 k_BT.

3.4.3 Two Protein Effects

We simulate systems with two protein domains per the domain construction of Sec. 3.4.2. The domains move through a Monte Carlo move that moves the center of the confining radius in the x-direction. This move is rejected if the domain's vertices are outside of the proposed new domain, and accepted if the domain's vertices are inside of the proposed new domain.

It is found that two orderphilic protein domains spontaneously assemble. The typical assembly behavior is seen in Fig. 3.6(a). Two protein domains are initially independent of each other at distances beyond that of their individual interaction ranges (Fig. 3.6(a,top)). Once within a certain distance, the formation of a solvation bridge between the protein domains occurs spontaneously (Fig. 3.6(a,middle)). The two protein domains then merge together (Fig. 3.6(a,bottom).

To quantify this behavior, the free energy with respect to the distance between two orderphilic protein domains is evaluated using REUS and MBAR. This is done for various coupling parameters, chemical potentials, and protein radii as shown in Figs. 3.6(b–d). The free energy minimum occurs when the protein domains are at a distance of $2R$ apart. Any further reduction of the distance between the proteins leads to repulsion due to elastic energy.

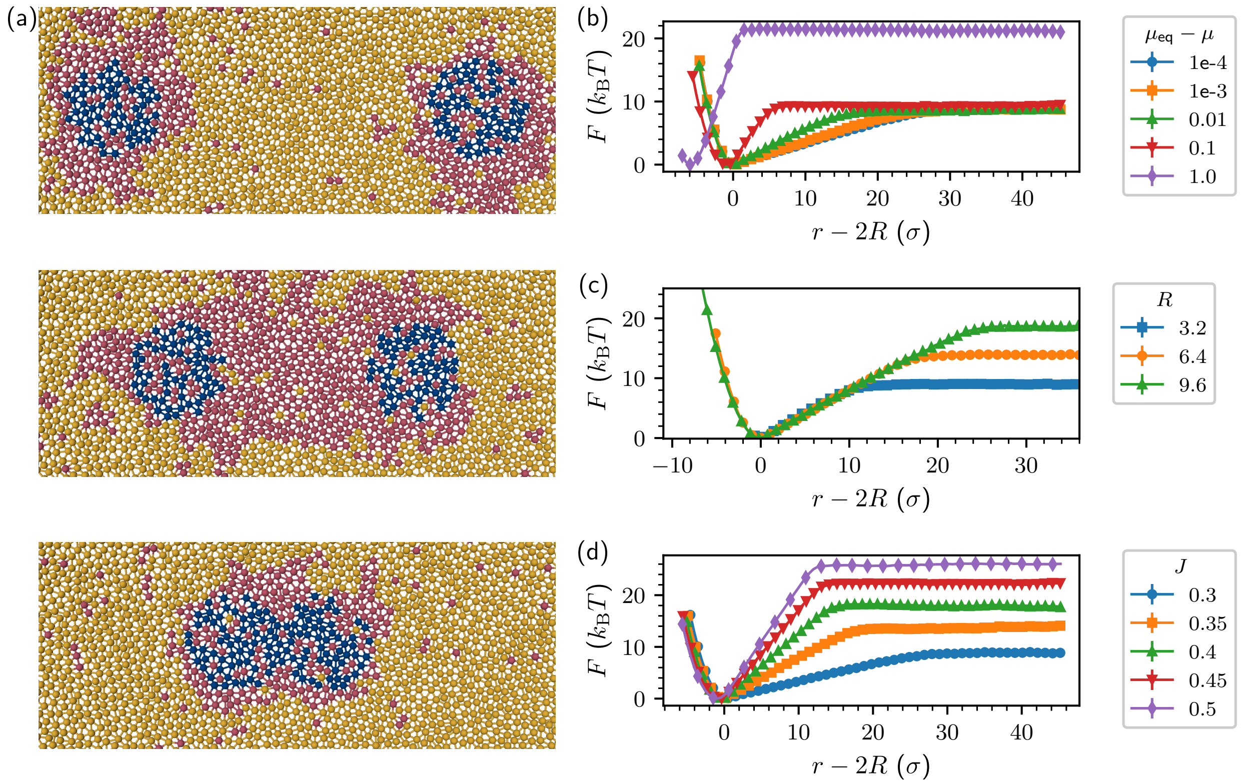


Figure 3.6: (a) The assembly of proteins with radii 6.4 σ at $k_\mathrm{b}\,(L_\mathrm{d}) = 20\ k_\mathrm{B}T$, $k_\mathrm{b}\,(L_\mathrm{o}) = 60\ k_\mathrm{B}T$, $J = 0.3\ k_\mathrm{B}T$, and $\mu_\mathrm{eq} - \mu = $ 1e-4 $k_\mathrm{B}T$. (Top) Before the formation of the solvation bridge. (Middle) During the formation of the solvation bridge. (Bottom) After merging. The following correspond to the free energy profiles of two orderphilic proteins at different parameters, $k_\mathrm{b}\,(L_\mathrm{d}) = 20\ k_\mathrm{B}T$, and $k_\mathrm{b}\,(L_\mathrm{o}) = 60\ k_\mathrm{B}T$ for a system with 10^4 total vertices. (b) Varying $\mu_\mathrm{eq} - \mu$ for a protein of radius 6.4 σ and $J = 0.3\ k_\mathrm{B}T$ (c) Varying protein radius for $\mu_\mathrm{eq} - \mu = $ 1e-4 $k_\mathrm{B}T$ and $J = 0.35\ k_\mathrm{B}T$. (d) Varying J for a protein of radius 6.4 σ and $\mu_\mathrm{eq} - \mu = $ 1e-4 $k_\mathrm{B}T$.

At distances greater than this minima, the free energy increases linearly with protein distance. A plateau is eventually reached that corresponds to when the two protein domains begin to spontaneously merge. Past this distance, the proteins are independent of one another. The range of the proteins' interactions increases with $J \to J_c$, protein size, and distance from coexistence, in agreement with the single protein trends in Sec. 3.4.2. The height of the plateau increases with the protein size and J. Variations in the bending rigidity and surface tension are found to not affect the plateau height as shown in Fig. A.1. For values of $\mu_\mathrm{eq} - \mu$ leading to energetic interactions in the same order as J, the distance from coexistence increases the plateau height.

The assembly of two proteins driven solely by composition effects follows the mechanism previously described in the orderphobic effect [7], and more generally that of the hydrophobic effect [104, 148]. This mechanism is that of two independent domains drifting until they enter some interaction range. The proteins then spontaneously assemble due to the minimization of their line tension. The free energy of this assembly is governed by λP_{int}, where P_{int} is the perimeter of the interface. The perimeter of the interface between two proteins depends linearly upon the protein distance, leading to the observed linear free energy profiles. Increases in R lead to an increased P_{int} at constant λ. Increases in J, on the other hand, lead to an increased λ that are greater than the corresponding reduction in P_{int}, hence yielding a larger value of λP_{int}. Under similar logic, one would expect that adjusting the chemical potential away from phase coexistence would lead to lower free energy incentives to assemble due to the reduction of the interface perimeter. While the observed range of interaction is reduced, the plateau height increases with distance from coexistence. This is due to a reduction in the amount of induced L_o vertices when the proteins assemble.

3.4.4 Multiple Protein Effects

The effect of multiple orderphilic proteins is considered through the use of point particle proteins, each a single vertex in size. The point particle proteins are allowed to freely diffuse under vertex displacements and mass conserving moves without the need for the previous confining radius construction. We first look at qualitative trends as the number of proteins is increased at constant J and μ, and as J is increased at constant protein number and μ. In a system with moderate line tension and close proximity to μ_{eq}, the addition of more proteins induces a larger stable L_o phase in a bulk L_d phase as shown in Fig. 3.7(a). In a system with a high protein number and a large distance from μ_{eq}, low line tensions lead to the proteins being dispersed while higher line tensions lead to protein assembly into a larger domain as seen in Fig. 3.7(b).

This behavior is quantified by obtaining phase diagrams with respect to the average composition for differing protein numbers using US and MBAR (Fig. 3.7(c)). The inclusion of protein vertices stabilizes L_o regions in a bulk L_d phase, observed through a corresponding shift in the phase diagram about the composition. This can be further understood by looking at the underlying free energy used to obtain the phase diagrams in Fig. A.2(a), where it is observed that the proteins allow for greater fluctuations in the average composition. Additionally, measurement of the line tension Fig. A.2(b) shows that the line tension is reduced by protein addition. This behavior is in agreement with the qualitative behavior previously seen in Figs. 3.7(a,b).

To conclude our discussion on composition effects, protein domains can generate large clusters of the thermodynamically disfavored phase in a bulk of the thermodynamically favored phase. Line tension drives this aggregation due to the reduction of the free energy per λP_{int}. This is in line with the previous orderphobic effect study [7] and expectations from the hydrophobic effect [104]. As previously mentioned, experimentally it is seen that L_o-L_d domains, both with and without proteins, do not necessarily phase separate macroscopically. Artificial membrane systems can have multiple phase-separated domains [81, 97, 149], while the lipid raft hypothesis for live systems speculates the existence of many dynamic nanoscopic domains [6, 80]. As line tension drives only macroscopic phase separation we will now consider additional factors in the membrane-protein model to drive microscopic phase behavior.

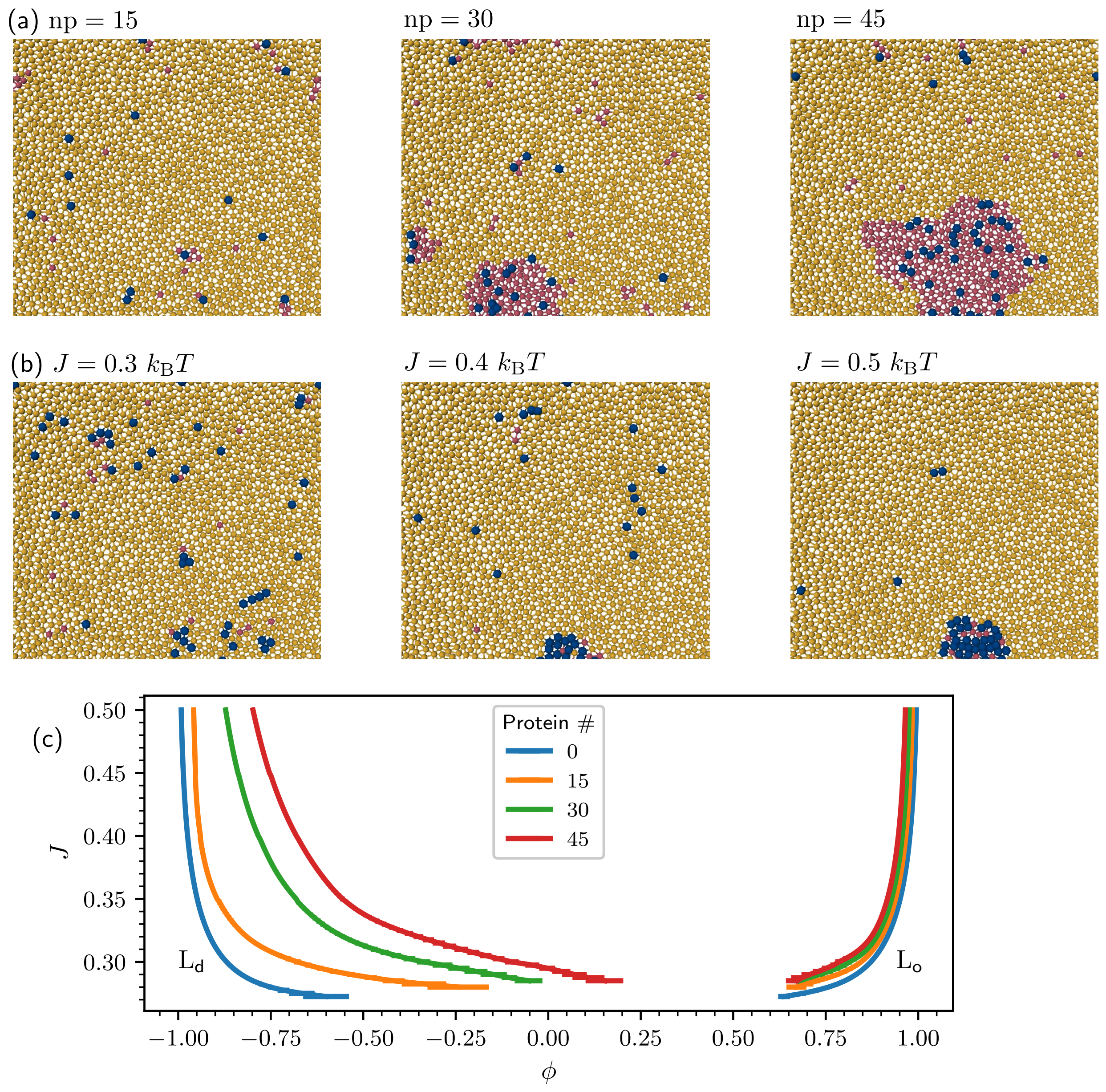


Figure 3.7: Multiple point particle protein results at $k_\mathrm{b}\,(L_\mathrm{d}) = 20\ k_\mathrm{B}T$ and $k_\mathrm{b}\,(L_\mathrm{o}) = 60\ k_\mathrm{B}T$. In (a) and (b), the size of a protein vertex is increased clarity. (a) Varying number of point particle proteins at $\mu_\mathrm{eq} - \mu = 0.04\ k_\mathrm{B}T$ and $J = 0.4\ k_\mathrm{B}T$. (b) Varying J at 45 point particle proteins and $\mu_\mathrm{eq} - \mu = 0.5\ k_\mathrm{B}T$. (c) Phase diagram of a system with orderphilic point particle proteins in a system with 1600 total vertices, $k_\mathrm{b}\,(L_\mathrm{d}) = 20\ k_\mathrm{B}T$, $k_\mathrm{b}\,(L_\mathrm{o}) = 60\ k_\mathrm{B}T$, and reweighted such that each phase has equal probability.

3.5 Bending Effects

We now examine bending effects related to the membrane-protein system. This covers the coupling between bending and compositional energies along with the effect of spontaneous curvature and the various shape transformation accessible to membranes.

3.5.1 Single Protein Effects

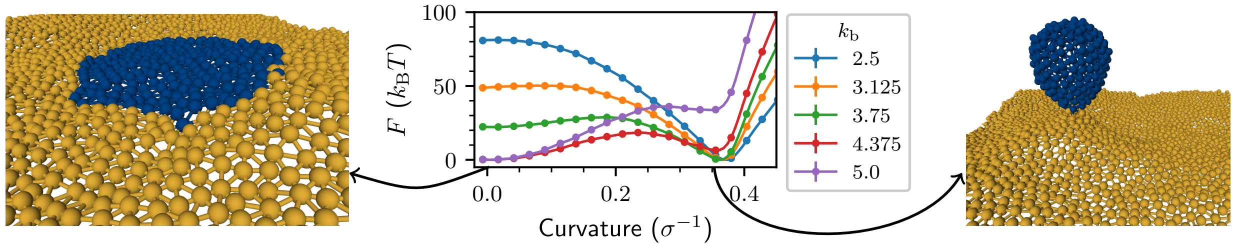


Figure 3.8: (Left) Flat protein of radii 10.3 σ at $k_\mathrm{b} = 5\ k_\mathrm{B}T$ and $J = 1\ k_\mathrm{B}T$. (Middle) Free energy versus curvature for various bending rigidities for a protein domain with radius 10.3 σ at $J = 1\ k_\mathrm{B}T$ and $\gamma, \tau = 0$ in a system with 2500 total vertices. (Right) Budded protein of initial radii 10.3 σ at $k_\mathrm{b} = 5\ k_\mathrm{B}T$ and $J = 1\ k_\mathrm{B}T$.

When an $\mathrm{L_o}$ ($\mathrm{L_d}$) domain reaches a sufficient size in a $\mathrm{L_d}$ ($\mathrm{L_o}$) bulk phase at a nonzero line tension, spontaneous budding of the domain can occur [82, 113, 150, 151]. Theoretically, this process has been well described by Lipowsky in the absence of surface tension, in which the membrane undergoes a transition from a flat state to a budded state as the line tension or spontaneous curvature of a domain increases [152, 153]. We will examine the budding process in the current computational model to see its behavior and compare it to the behavior described by Lipowsky. We then examine mechanisms for the generation of spontaneous curvature for protein domains. To do so, we evaluate the free energy of the budding process for a single protein domain.

To evaluate the free energy we sample with respect to the average mean curvature of a protein domain. The average mean curvature of the protein domain, $\bar{H}_\mathrm{protein}$ is taken to be

$$\bar{H}_\mathrm{protein} = \frac{1}{N_\mathrm{protein}} \sum_{i \in \{\text{protein vertices}\}} H_i\,, \tag{3.5.1}$$

where N_protein is the number of protein vertices. The simulation setup is shown in Fig. 3.8(left) with a single protein domain in a flat state. To keep the number of vertices, including $\mathrm{L_o}$ vertices, in the protein domain constant we fix the mass of the system, and the confining radius of the protein domain is set to 19.21 σ. Performing US with respect to $\bar{H}_\mathrm{protein}$ allows for the study of the transition between the previously seen flat state to that of the budded state (Fig. 3.8(right)).

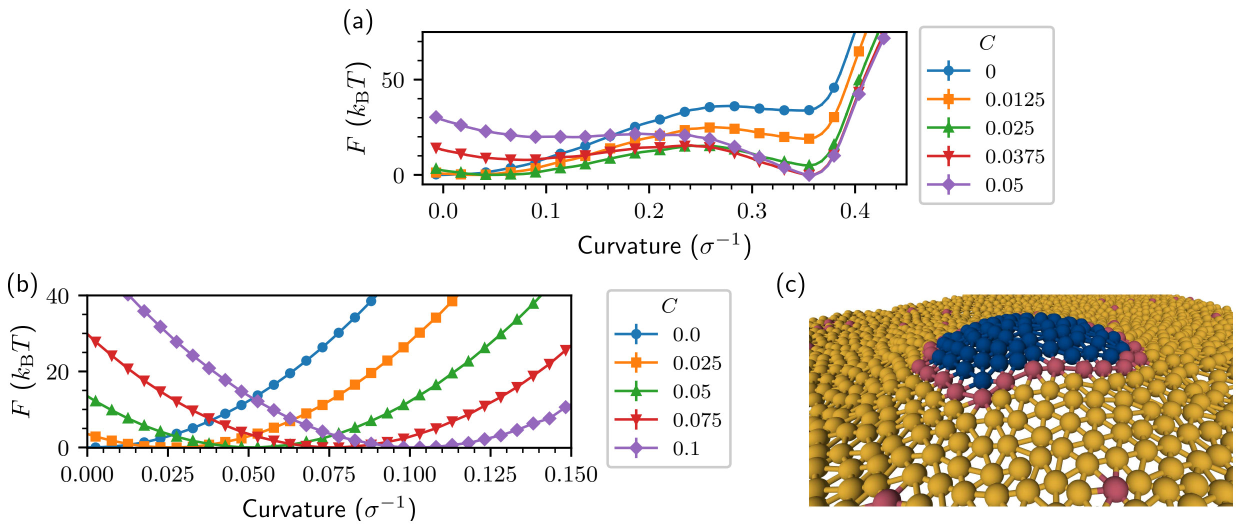


Figure 3.9: (a) Free energy versus curvature for a protein domain of radius 10.3 σ at $k_\mathrm{b} = 5\ k_\mathrm{B}T$, $J = 1\ k_\mathrm{B}T$, and $\gamma, \tau = 0$ in a system with 2500 total vertices. (b) Free energy versus curvature for a single L_o domain of radius 4.8 σ at $k_\mathrm{b}\,(L_\mathrm{d}) = 20\ k_\mathrm{B}T$, $k_\mathrm{b}\,(L_\mathrm{o}) = 60\ k_\mathrm{B}T$, and composition energy governed by Eq. 3.5.2 with parameters given in Eq. 3.5.3 for a system with 2500 total vertices and 80 $\mathrm{L_o}$ vertices at various spontaneous curvatures obtained through reweighting. (c) Dimpled protein of radii 4.8 σ at $k_\mathrm{b}\,(L_\mathrm{d}) = 20\ k_\mathrm{B}T$, $k_\mathrm{b}\,(L_\mathrm{o}) = 60\ k_\mathrm{B}T$, and composition energy governed by Eq. 3.5.2 with parameters given by Eq. 3.5.3 in a system with 2500 total vertices at $C = 0.1\ \sigma^{-1}$ with 80 $\mathrm{L_o}$ vertices.

The free energy with respect to $\bar{H}_\mathrm{protein}$ is evaluated using REUS and MBAR. A uniform bending rigidity in the range of 2.5–5 $k_\mathrm{B}T$ is used for both the $\mathrm{L_o}$ and $\mathrm{L_d}$ vertices. This is done as vertices with higher bending rigidity would be increasingly difficult to sample due to large free energy barriers. The free energy in this bending rigidity range is evaluated in Fig. 3.8(middle), where a transition between spontaneous budding at lower values of the bending rigidity to a stable flat phase at higher values of the bending rigidity is seen. This result is in line with the theory of Lipowsky [152] with the observed stable phases being a flat state of zero curvature or a fully budded state. Surface tension is found to increase the barrier related to budding (Fig. A.3(a)). This is due to τ creating an energetic cost for the reduction of projected area in bud formation. Note that performing simulations at a fixed projected area would lead to a large energetic barrier for budding as the excess area of the membrane would need to be converted into the area of the bud [154].

We now evaluate the effect of protein spontaneous curvature, which we denote as $C = C\,(\phi_i = 1, n_i = 1)$. This is done by performing reweighting with respect to $\bar{H}_\mathrm{protein}$ to yield the free energy profiles in Fig. 3.9(a). Per the theory of Lipowsky [152], relatively small spontaneous curvatures can induce spontaneous budding in conjunction with line tension. In the cases where spontaneous budding does not occur the stable flat phase's minimum is shifted by the spontaneous curvature, resulting in a dimpled phase.

To see in greater detail the shifting of the flat phase to that of the dimpled phase by C at physiological conditions, we evaluate the free energy of a protein domain with respect to $\bar{H}_{\text{protein}}$ at realistic bending rigidities and low line tension. To keep the protein domain together within the large confining radius, V_{c} is modified. This modified composition energy is

$$V_{\text{c},2} = -\frac{1}{2} \sum_{i} \sum_{j(i)} J\left(\phi_i, n_i, \phi_j, n_j\right) \phi_i \phi_j \,, \tag{3.5.2}$$

where for a system with only orderphilic proteins the values of $J\left(\phi_i, n_i, \phi_j, n_j\right)$ are

$$\begin{aligned} J\left(\phi_i = \pm 1, n_i = 0, \phi_j = \pm 1, n_j = 0\right) &= 0 \,, \\ J\left(\phi_i = 1, n_i = 0, \phi_j = 1, n_j = 1\right) &= 0 \,, \\ J\left(\phi_i = -1, n_i = 0, \phi_j = 1, n_j = 1\right) &= 10 \ k_{\text{B}}T \,, \\ J\left(\phi_i = 1, n_i = 1, \phi_j = 1, n_j = 1\right) &= 0 \ k_{\text{B}}T \,. \end{aligned} \tag{3.5.3}$$

This choice of parameters leads to the L_o vertices wetting the protein domain as seen in Fig. 3.9(c). This is due to the protein vertices having an energetic penalty to interact with L_d vertices, while interactions between L_o–protein vertices and L_d–L_o vertices have no penalty. Hence, the protein domain has effectively zero line tension. The confining radius is set to 9.61 σ with mass conserving moves used for the protein vertices and 80 L_o vertices.

The free energy with respect $\bar{H}_{\text{protein}}$ is evaluated for the system governed by Eqs. (3.5.2) and (3.5.3) with $C = 0$ using REUS and MBAR, and then reweighted to different curvatures to yield the free energy profiles in Fig. 3.9(b). A stable basin around the specified C is seen in the profiles, with the systems having a stable dimpled state for $C > 0$ due to the low line tension in comparison to the previous results of Fig. 3.8(middle) and Fig. 3.9(a). Surface tension slightly shifts the curvature of the free energy profile per Fig. A.3(b), and does not have the large effect previously seen for the budding transition in Fig. A.3(a) due to the projected area of the system being relatively constant for a dimpled domain.

3.5.2 Two Protein Effects

We now simulate systems that have two protein domains with nonzero spontaneous curvature through $C = C\left(\phi_i = 1, n_i = 1\right)$. The simulation system is seen in Fig. 3.10(a). It is observed in short simulations that dimpled protein domains of radii 6.4 σ do not spontaneously assemble in contrast with the results of Section 3.4.3. This behavior is now quantitatively analyzed using umbrella sampling.

The free energy with respect to protein distance is evaluated using REUS and MBAR. The spontaneous curvature leads to an elastic repulsion between domains at long distances, in line with previous theories of dimpled domains [154, 155]. For proteins with radii of 6.4 σ increasing curvature produces a barrier for domain assembly (Fig. 3.10(b)). The height of the barrier increases with C, and it is found at the highest tested spontaneous curvature of 0.3 σ^{-1} that the free energy incentive to assemble also increases. For larger proteins with

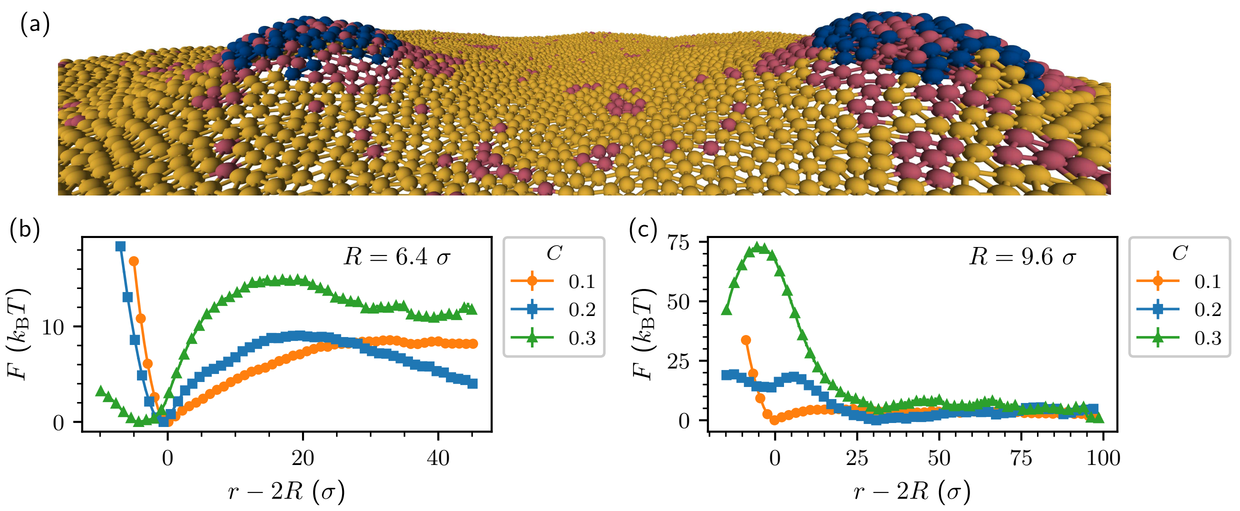


Figure 3.10: (a) Stable orderphilic protein domains of radii 6.4 σ with $k_b(L_d) = 20\ k_BT$, $k_b(L_o) = 60\ k_BT$, $\gamma, \tau = 0$, $J = 0.3\ k_BT$, $\mu_{eq} - \mu = $ 1e-4 k_BT, and $C = 0.3\ \sigma^{-1}$ in a system with 10^4 vertices. The following are free energy versus protein distance for two orderphilic proteins with varying parameters. (b) Varying C at $\gamma, \tau = 0$, $J = 0.3\ k_BT$, $k_b(L_d) = 20\ k_BT$, $k_b(L_o) = 60\ k_BT$, and $\mu_{eq} - \mu = $ 1e-4 k_BT for two orderphilic proteins of radii 6.4 σ. (c) Varying C at $\gamma, \tau = 0$, $J = 0.3\ k_BT$, $k_b(L_d) = 20\ k_BT$, $k_b(L_o) = 60\ k_BT$, and $\mu_{eq} - \mu = $ 1e-4 k_BT for two orderphilic proteins of radii 9.6 σ.

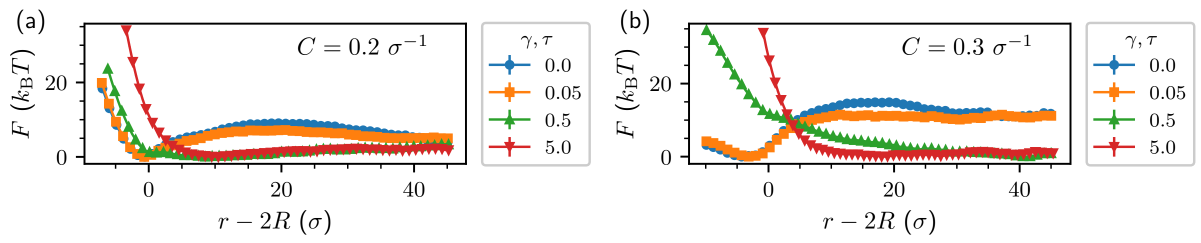


Figure 3.11: (a) Free energy versus protein distance for two orderphilic proteins of radii 6.4 σ at $k_b(L_d) = 20\ k_BT$, $k_b(L_o) = 60\ k_BT$, and $\mu_{eq} - \mu = $ 1e-4 k_BT for varying γ, τ at $C = 0.2\ \sigma^{-1}$ and $J = 0.3\ k_BT$ in a system with 10^4 total vertices. (b) Free energy versus protein distance for two orderphilic proteins of radii 6.4 σ at $k_b(L_d) = 20\ k_BT$, $k_b(L_o) = 60\ k_BT$, and $\mu_{eq} - \mu = $ 1e-4 k_BT for varying γ, τ at $C = 0.3\ \sigma^{-1}$ and $J = 0.3\ k_BT$ in a system with 10^4 total vertices.

radii of 9.6 σ the barrier is much greater (Fig. 3.10(c)). This leads to the protein domains remaining separated. Similar to the smaller protein domains, the values of the free energy profiles for $r - 2R < 0$ are not as repulsive at high spontaneous curvatures as they are for low spontaneous curvatures. The region at $r - 2R < 0$ for high spontaneous curvatures is, however, metastable with respect to the independent domain state. Previously observed trends for μ and J, with increasing distance from coexistence leading to shorter compositional interaction range and increasing J leading to shorter compositional interaction range and larger incentives to assemble, are found to hold in the presence of nonzero spontaneous curvature as seen in Fig. A.4.

Surface tension provides another barrier for domain merging by enhancing the bending interactions as seen in Fig. 3.11. This effect becomes more significant at higher curvatures with the progression of surface tension shifting the free energy plateau at low curvatures (Fig. 3.11(a)) to producing a repulsive interaction between the domains at high curvatures (Fig. 3.11(b)). One expects the elastic length scale of the protein domain interactions to depend upon the surface tension per $\sqrt{k_\mathrm{b}/\sigma}$, allowing for the modulation of the interaction range between the domains which matches the observed trend in the free energy profiles of Fig. 3.11. This length scale dependence on surface tension is in line with previous theoretical work on the assembly of rigid inclusions in lipid bilayers [155].

The elastic interaction driven by protein spontaneous curvature together with the compositional interaction provides a mechanism for microphase separation. The effects of the elastic interaction produce a barrier for assembly at large domain sizes and/or spontaneous curvatures, while line tension drives the spontaneous assembly of small domains. This suggests that domains will grow to some critical size at which the elastic and compositional interactions are in balance and prevent further assembly. Though this size is difficult to quantify in the two protein systems in which the elastic and compositional energetics have to be controlled meticulously, we now quantify it in a multiple point particle protein simulation.

3.5.3 Multiple Protein Effects

We now look at the behavior of multiple orderphilic proteins using point particle proteins with spontaneous curvature. At low spontaneous curvatures of protein vertices, the protein vertices create one large domain rich in the $\mathrm{L_o}$ phase in line with the results of Section 3.4.4. However, when the curvature is increased past a certain value, the spontaneous formation of many domains is observed as shown in Fig. 3.12(a), in line with the hypothesis at the end of Section 3.5.2. We now quantitatively examine this behavior.

The clusters formed by the orderphilic protein vertices are identified using a depth-first search algorithm [156, 157]. Pseudo-code for the implemented version of the depth-first search algorithm is given in Algorithms 1 and 2, in which a list of clusters containing orderphilic protein vertices and $\mathrm{L_o}$ vertices is obtained by searching for clusters over all vertices. This is done by identifying a protein vertex that is not in a cluster per Algorithm 1, and then recursively going over all $\mathrm{L_o}$ and protein vertices neighboring that protein vertex until all vertices in the cluster are identified per Algorithm 2. The depth-first search algorithm returns

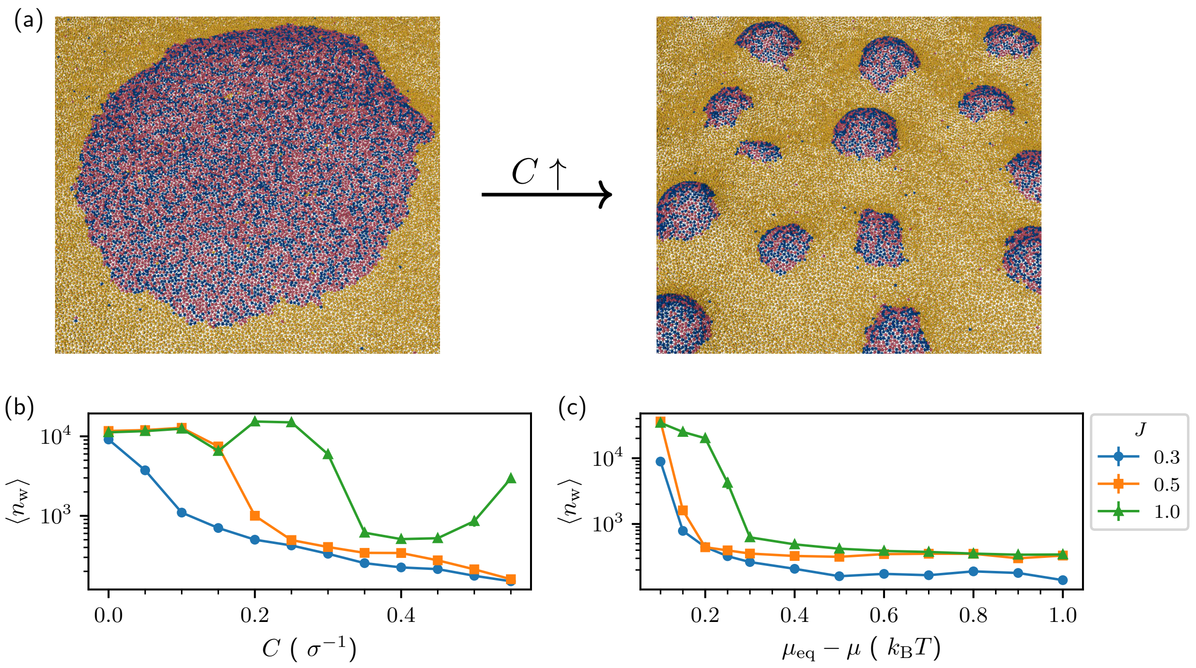


Figure 3.12: (a) System with 4800 orderphilic point proteins with 40000 total vertices at $k_\mathrm{b}\,(L_\mathrm{d}) = 20\ k_\mathrm{B}T$, $k_\mathrm{b}\,(L_\mathrm{o}) = 60\ k_\mathrm{B}T$, $J = 0.5\ k_\mathrm{B}T$, $\mu_\mathrm{eq} - \mu = 0.25\ k_\mathrm{B}T$, $\gamma, \tau = 0$, and $C = 0.05\ \sigma^{-1}$ (left) and $C = 0.25\ \sigma^{-1}$ (right). (b) Weight-averaged cluster size for a system with 4800 orderphilic point proteins with 40000 total vertices at $k_\mathrm{b}\,(L_\mathrm{d}) = 20\ k_\mathrm{B}T$, $k_\mathrm{b}\,(L_\mathrm{o}) = 60\ k_\mathrm{B}T$, $\mu_\mathrm{eq} - \mu = 0.25\ k_\mathrm{B}T$, $\gamma, \tau = 0$, and varying C and J. (c) Weight-averaged cluster size for a system with 4800 orderphilic point proteins with 40000 total vertices at $k_\mathrm{b}\,(L_\mathrm{d}) = 20\ k_\mathrm{B}T$, $k_\mathrm{b}\,(L_\mathrm{o}) = 60\ k_\mathrm{B}T$, $C = 0.3\ \sigma^{-1}$, $\gamma, \tau = 0$, and varying μ and J.

a list of the protein-induced clusters and the vertices in each cluster. Three metrics are evaluated using the cluster list. The first metric is the number of clusters, N_cluster. The second metric is the number-averaged cluster size n_n measured by

$$n_\mathrm{n} = \frac{\sum_{i=1}^{N_\mathrm{cluster}} n_i}{\sum_{i=1}^{N_\mathrm{cluster}} 1}, \tag{3.5.4}$$

where n_i is the size of cluster i. This metric averages all cluster sizes equally. The final metric

Algorithm 1: Depth-first search algorithm

```
   Data: Triangulation $\mathcal{T}$. Composition variables $\boldsymbol{\phi}$. Protein identities $\mathbf{n}$.
1  Initialize empty list of clusters $\mathbf{L}$.
2  Initialize list of vertex statuses $\mathbf{S}$ set to $-1$ for all vertices.
3  for vertex $k = 1, \dots, N$ do
4      if $n_k = 1$ and $S_k = -1$ then
5          Initialize empty list $L_j$.
6          RECURSIVE DFS($k, \mathcal{T}, \boldsymbol{\phi}, \mathbf{n}, L_j, \mathbf{S}$)
7          Store $L_j$ in $\mathbf{L}$.
```

Algorithm 2: Recursive part of depth-first search

```
   RECURSIVE DFS($i, \mathcal{T}, \boldsymbol{\phi}, \mathbf{n}, L, \mathbf{S}$)
1  Set $S_i = 0$.
2  Store $i$ in $L$.
3  for $k \in j(i)$ do
4      if $n_k = 1$ or $\phi_k = 1$ then
5          if $S_k = -1$ then
6              RECURSIVE DFS($k, \mathcal{T}, \boldsymbol{\phi}, \mathbf{n}, L, \mathbf{S}$)
```

Figure 3.13: Pseudo-code for the depth-first search algorithm used to identify clusters and their sizes.

is the weight-averaged cluster size n_w measured by

$$n_\mathrm{w} = \frac{\sum_{i=1}^{N_\mathrm{cluster}} n_i^2}{\sum_{i=1}^{N_\mathrm{cluster}} n_i}. \tag{3.5.5}$$

This metric puts more weight into clusters of larger sizes, and is useful for discerning large clusters in the presence of smaller clusters. Simulations are used to obtain $\langle N_\mathrm{cluster} \rangle$, $\langle n_\mathrm{n} \rangle$, and $\langle n_\mathrm{w} \rangle$.

We examine the trends in cluster size with respect to the spontaneous curvature and line tension. Sweeping over values of spontaneous curvature at three different values of line tension, the same general trend is observed as shown in Fig. 3.12(b) with Figs. A.5(a,b) containing additional statistics. At low spontaneous curvatures, the system is in a macrophase state with one large cluster. Beyond some value of C, the weight-averaged cluster size begins to decrease indicating microphase separation. Large line tensions necessitate larger values of C for this transition to occur. This is in line with the previous behavior seen with respect to two dimpled proteins.

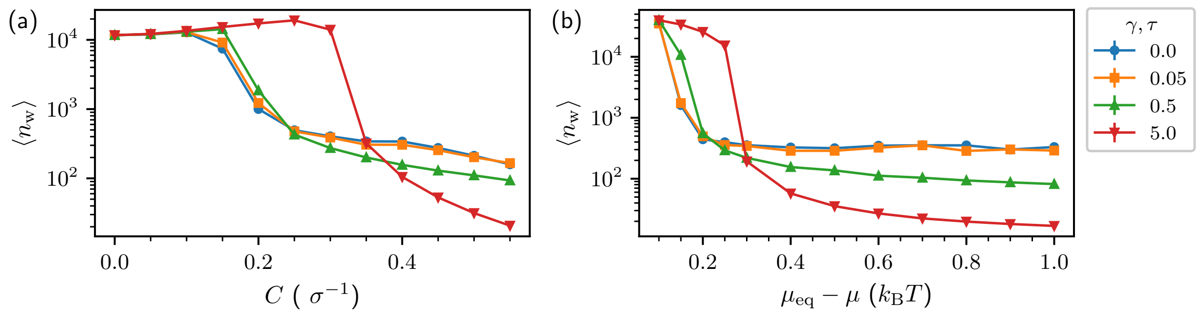


Figure 3.14: (a) Weight-averaged cluster size for a system with 4800 orderphilic point proteins with 40000 total vertices at $k_\mathrm{b}\,(L_\mathrm{d}) = 20\ k_\mathrm{B}T$, $k_\mathrm{b}\,(L_\mathrm{o}) = 60\ k_\mathrm{B}T$, $\mu_\mathrm{eq} - \mu = 0.25\ k_\mathrm{B}T$, $J = 0.5\ k_\mathrm{B}T$, and varying γ, τ and C. (b) Weight-averaged cluster size for a system with 4800 orderphilic point proteins with 40000 total vertices at $k_\mathrm{b}\,(L_\mathrm{d}) = 20\ k_\mathrm{B}T$, $k_\mathrm{b}\,(L_\mathrm{o}) = 60\ k_\mathrm{B}T$, $C = 0.3\ \sigma^{-1}$, $J = 0.5\ k_\mathrm{B}T$, and varying γ, τ, and μ.

The weight averaged cluster size decreases as $\mu_\mathrm{eq} - \mu$ increases as shown in Fig. 3.12(c) with additional statistics in Figs. A.5(c,d). This in opposition to the results in Figs. 3.6(b) and A.4(a), but it is noted the results in those figures use the artificial protein domain construction to obtain those results. With the distance from phase coexistence increasing, the clusters consist less of the $\mathrm{L_o}$ vertices and more of the protein vertices. As the $\mathrm{L_o}$ vertices have no curvature the resulting clusters have higher curvature and hence more repulsion leading to more clusters. This behavior is hard to probe with the two protein construction due to the artificial behavior with regards to the $\mathrm{L_o}$ vertices, while the point protein vertices have no such issue.

Surface tension is found to prevent microphase separation up to moderate values of C and μ, and enhances microphase separation for higher values of C and μ as shown in Fig. 3.14. This partially contradicts the result from Fig. 3.11 where surface tension is found to only enhance repulsion. Looking at the other metrics, surface tension increases the value of $\langle N_\mathrm{cluster} \rangle$ for increasing curvature per Fig. A.6(a) while having a similar trend for increasing μ except surface tension reduces $\langle N_\mathrm{cluster} \rangle$ for $\mu_\mathrm{eq} - \mu$ per Fig. A.6(c). The value of $\langle n_\mathrm{n} \rangle$ is found to steadily decrease for increasing curvature and chemical potential as seen in Figs. A.6(b,d) with surface tension having a similar trend to that seen in Fig. 3.14 by amplifying the reduction past certain values of C and μ. Intuitively one expects that $\langle N_\mathrm{cluster} \rangle$ and $\langle n_\mathrm{n} \rangle$ will be inversely related, which is seen in all cases except for increasing curvature with increasing surface tension in Figs. A.6(a,b). This suggests that in the case of moderate curvatures with high surface tension, small clusters form that do not significantly affect the value of $\langle n_\mathrm{n} \rangle$ and $\langle n_\mathrm{w} \rangle$. In the case of increasing $\mu_\mathrm{eq} - \mu$ changes in $\langle N_\mathrm{cluster} \rangle$ and $\langle n_\mathrm{n} \rangle$ are initially more pronounced than $\langle n_\mathrm{w} \rangle$ at high tensions, in line with the formation of many small clusters in the presence of a few large clusters.

3.5.3.1 Spherical Systems

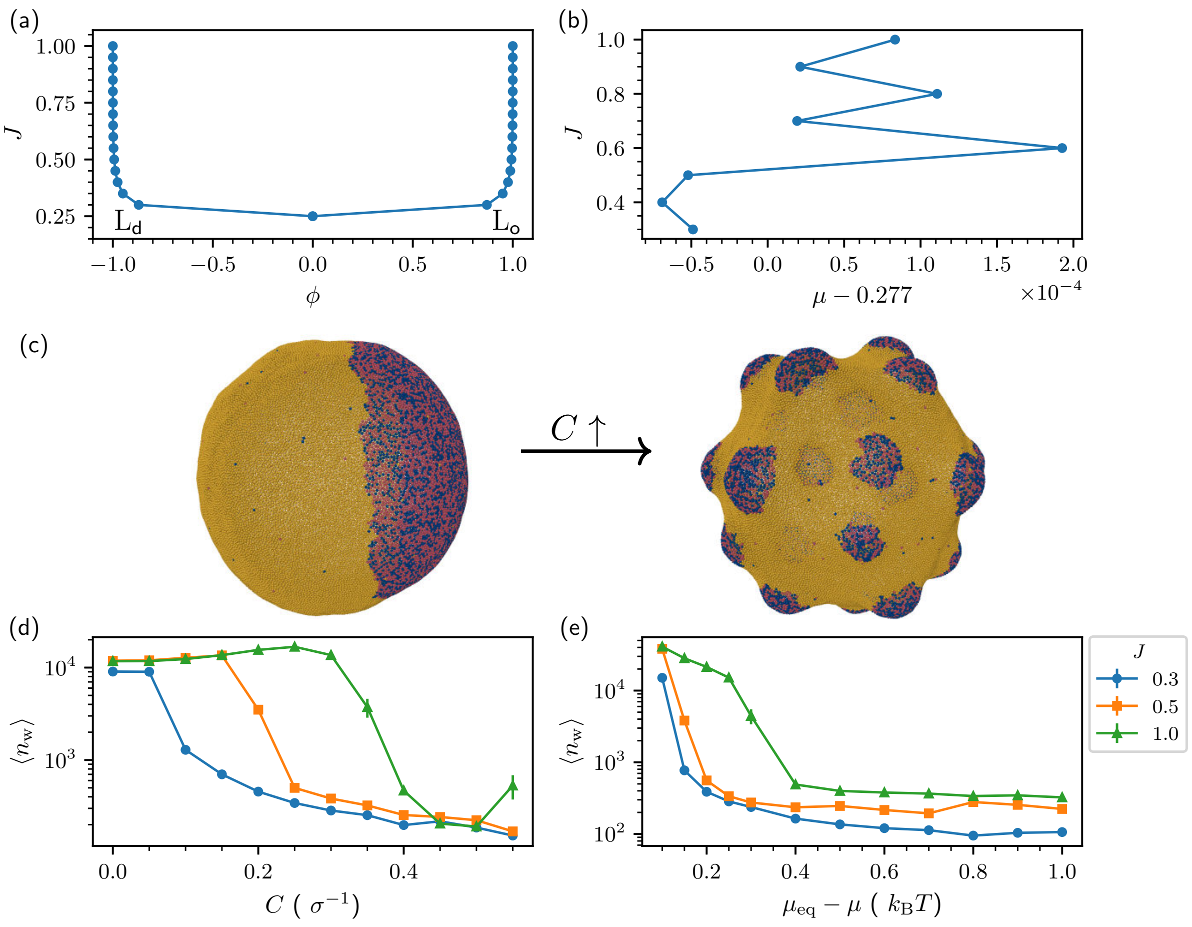


Figure 3.15: (a) Phase diagram in ϕ vs J space for a spherical membrane system with 40962 vertices, $k_\text{b}\,(L_\text{d}) = 20\ k_\text{B}T$, and $k_\text{b}\,(L_\text{o}) = 60\ k_\text{B}T$. (b) Phase diagram in μ vs J space determined using the bracketing method for the system in (a) with lines indicating μ_eq. (c) Spherical system with 4800 orderphilic point proteins with 40962 total vertices at $k_\text{b}\,(L_\text{d}) = 20\ k_\text{B}T$, $k_\text{b}\,(L_\text{o}) = 60\ k_\text{B}T$, $J = 0.5\ k_\text{B}T$, $\mu_\text{eq} - \mu = 0.25\ k_\text{B}T$, $\gamma, P = 0$, and $C = 0.05\ \sigma^{-1}$ (left) and $C = 0.25\ \sigma^{-1}$ (right). (d) Weight-averaged cluster size for a spherical system with 4800 orderphilic point proteins with 40962 total vertices at $k_\text{b}\,(L_\text{d}) = 20\ k_\text{B}T$, $k_\text{b}\,(L_\text{o}) = 60\ k_\text{B}T$, $\mu_\text{eq} - \mu = 0.25\ k_\text{B}T$, $\gamma, P = 0$, and varying C and J. (e) Weight-averaged cluster size for a spherical system with 4800 orderphilic point proteins with 40962 total vertices at $k_\text{b}\,(L_\text{d}) = 20\ k_\text{B}T$, $k_\text{b}\,(L_\text{o}) = 60\ k_\text{B}T$, $C = 0.3\ \sigma^{-1}$, $\gamma, P = 0$, and varying μ and J.

We now consider the effects of multiple point orderphilic proteins in a spherical system. We do so on a system that is roughly the same size of the planar systems in Figs. 3.12 and 3.14, consisting of 40962 vertices generated using the Loop subdivision scheme introduced in Section 2.4.1. The radius of the initial triangulation is set to 69.48 σ and the spontaneous curvature of the $\mathrm{L_o}$–$\mathrm{L_d}$ vertices is set to $\frac{1}{69.48}\ \sigma^{-1}$.

As in the planar system, the phase diagram of the spherical system is established so that distance from phase coexistence set by $\mu_{\mathrm{eq}} - \mu$ can be quantified. Determining the phase diagram using US for this system is computationally difficult due to the larger number of vertices in the spherical system than that of the planar system used for the results in Fig. 3.4. The phase diagrams are instead obtained for a system with $k_{\mathrm{b}}\,(L_{\mathrm{d}}) = 20\ k_{\mathrm{B}}T$ and $k_{\mathrm{b}}\,(L_{\mathrm{o}}) = 60\ k_{\mathrm{B}}T$ by finding values of μ that establish phase coexistence using the ansatz that the phase diagrams have similar behavior to that of the Ising model. We first simulate 10 systems with mass conserving Monte Carlo moves that have half $\mathrm{L_d}$ vertices and half $\mathrm{L_o}$ vertices at select values of J. The equilibrated systems are then simulated with non-mass conserving Monte Carlo moves from which the number of trajectories that commit to either the $\mathrm{L_d}$ or $\mathrm{L_o}$ phase is counted, which are denoted as $N_{L_{\mathrm{d}}}$ and $N_{L_{\mathrm{o}}}$ respectively. As phase coexistence has the probability of each phase to be equal [144, 145] and the phase diagram of the Ising model is symmetric with respect to composition, the value of μ that yields an equal number of trajectories committing to either phase is μ_{eq}. This objective function is written as

$$f(\mu) = N_{L_{\mathrm{d}}}(\mu) - N_{L_{\mathrm{o}}}(\mu)\,, \tag{3.5.6}$$

where the value of μ_{eq} is found with the bisection algorithm [158] by finding the root of Eq. (3.5.6). The obtained values of μ_{eq} along with the corresponding phase diagram in ϕ versus J space obtained through simulations in the $\mathrm{L_d}$ and $\mathrm{L_o}$ phases at μ_{eq} are shown in Figs. 3.15(a,b). Similar to the results in Figs. 3.4(a,c), the phase diagram in ϕ versus J space is symmetric and the value of μ_{eq} has little dependence on J.

Tuning the value of the protein vertex spontaneous curvature, C, in the spherical systems yields similar behavior to that of the planar systems, with the protein vertices separating macroscopically at low curvature and microscopically at higher curvatures (Fig. 3.15(c)). This is quantified using measurements of the weight averaged cluster size in Figs. 3.15(d,e) and the other metrics in Fig. A.7, where identical trends to the planar case are found. This demonstrates that the background curvature of a sphere has minimal effect on the observed planar results under the conditions considered.

3.6 Conclusions and Discussions

We have looked at the assembly of $\mathrm{L_o}$ inducing protein domains in a bulk $\mathrm{L_d}$ membrane phase using a model commensurate with the relevant biological length and time scales. Line tension drives the assembly of these domains until elastic energetics between domains prevent further assembly. The elastic energetics are due to the spontaneous curvature of the protein domains, and the equilibrium domain size is controlled by the line tension, bending rigidity which enhances elastic interactions, and surface tension of the $\mathrm{L_o}$–$\mathrm{L_d}$ phases. Experimental work in giant unilamellar vesicles and giant plasma membrane vesicles with $\mathrm{L_o}$–$\mathrm{L_d}$ phase separation has observed the trend of domain size increasing with line tension while trends with surface tension remain controversial [159–163], though to our knowledge no experimental study has analyzed the trends in bending rigidity.

The assembly of two solute domains with a barrier is in contrast with processes driven solely by pre-transition effects such as the hydrophobic effect [104, 164] and the prior orderphobic effect study [7]. In absence of spontaneous curvature, the assembly is energetically favorable under all cases considered. Some concerns could arise that this process is not necessarily spontaneous. The formation of the solvation bridge leads to the generation of the thermodynamically disfavored phase, which intuitively leads to the process having a barrier. It has been shown that a similar Ising model-based energy leads to spontaneous assembly of two hydrophobic solutes via a committor analysis to determine good collective variables [148]. Below a certain distance between the solutes, the formation of the solvation bridge was found to lower the free energy with subsequent reduction in the solute distance being barrier-less. Above this solute distance, the process is not spontaneous as solvation bridge formation does not lower the free energy of the system. Though similar kinetic studies are not considered here, it is likely the case due to similarities in the energetic potential.

Our work on understanding the generation of curvature in a single $\mathrm{L_o}$ induced protein domain follows the theory of Lipowsky [152] for the tensionless case. Non-zero spontaneous curvature is found to be needed to generate a dimpled state to drive microphase behavior. Surface tension regulates complete budding while having minimal effect on the curvature of slightly dimpled states. Reference [154] has speculated a mechanism for which competition between surface tension and line energies leads to stable dimpled states, but this behavior is not observed in our computational model. This is in agreement with the work of Lipowsky [152], and other work theoretical work in the context of budded domains acting as tension regulators [165] and budding in clathrin-mediated endocytosis [166, 167] that demonstrated tension serves to regulate complete budding. Additionally, the line tension needed to facilitate the budding of $\mathrm{L_o}$ domains is nontrivial due to the large bending rigidity of the $\mathrm{L_o}$ phase. This necessitates large domain sizes and line tensions which is not compatible with the hypothesis of lipid rafts being nanoscopic domains and existing near physiological critical points where line tension is low.

Recent theoretical and computational work has considered similar mechanisms for the generation of multiple domains. This work has focused on spontaneous curvature causing microphase separation on spherical systems with mass conservation [157, 168–172]. Our work differs by focusing on a flat surface with non-mass conserving moves, though the final result has been extended to spherical systems, along with showing the importance of protein domains. Prior studies [157, 169] have utilized the assumption that large regions with spontaneous curvature can be unstable, leading to the formation of smaller regions at low line tension that has been quantified here in the case of two domains merging into one. Some of this prior work better describes the microemulsion behavior some expect of lipid rafts which have not been fully explored within our framework yet [157, 173, 174], along with the effect of Gaussian curvature for different Gaussian moduli [169, 171, 172]. Prior theoretical work for flat systems has explored the energetics of curved domains [154, 155], though our computational results also incorporate effects due to line tension along with allowing for the merging of domains.

Besides the coarse-grained description of lipids and proteins considered in this work, live cells have other phenomena that may impact the formation of lipid rafts. For example, biological membranes in live cells are coupled with the cytoskeleton. This interaction has important effects on both membrane shape through extensive pushing and pulling by actin [175, 176], and membrane composition, through mechanisms such as pinning from the cytoskeleton which causes fencing of domains and limits diffusion [177, 178]. L_o and L_d domains can also be generated through organelle contact [179]. Live cell membranes also experience dynamic behavior through lipid trafficking, which is hypothesized as a potential mechanism for lipid raft formation [180–183]. The model of this work uses Metropolis Monte Carlo which assumes equilibrium behavior, and we acknowledge that live cell membranes are in a non-equilibrium state with many moving parts and do not dismiss that the inclusion of such phenomena could lead to different results.

There are many possible directions for future studies. As previously noted the effect of Gaussian moduli differences between L_o–L_d phases and the incorporation of nonequilibrium phenomena such as membrane recycling and coupling with the cytoskeleton has not been tested. Ideas from our model can be applied to other simulation models of biological membranes, such as atomistic and coarse-grained simulation models, though difficulties remain due to the necessary time and length scales needed to study lipid rafts. Non-mass conserving moves can be complicated as the identity swapping of lipids and proteins in these models are nontrivial, though recent schemes have begun to address this issue [184–186]. Nonetheless, by modulating protein length and curvature of proteins, the effects demonstrated in this chapter should be reproducible and able to explain a wide range of biological behaviors.

Part B

Data-Driven Reaction Learning

Chapter 4

Introduction to Transition Path Theory

4.1 Introduction

A fundamental problem in chemistry is the discovery of mechanistic pathways governing kinetic processes at the microscopic level. These processes include phase transitions in colloidal systems [187], chemical reactions at aqueous interfaces [188], and protein folding [189]. While diverse in context, they exhibit a common bottleneck in the form of high-energy barriers, which separate the reactant and product states of the pathway. Despite remarkable progress in high-performance molecular simulations [51, 190, 191], finding these pathways is difficult due to the rarity of barrier-crossing events at timescales achievable by current computational resources. Studying these rare events constitute identifying the transition pathways, and sampling them is an important part of obtaining a mechanistic understanding of the problem.

A common approach in mechanistic pathways with high-energy barriers is the identification of a low-dimensional set of collective variables (CVs). If the set of collective variables is sufficient to predict dynamical progress along the mechanistic pathway, it is termed a reaction coordinate. With a reaction coordinate, integration of all other degrees of freedom in the system typically yields local free energy minima corresponding to the reactant and product states with a local free energy maximum corresponding to the transition state (see Fig. 4.1 for a schematic). While idealistic, many processes such as water autoionization [192], crystallization [193, 194], and protein folding [189] are amendable to this analysis. The use of good collective variables allows for efficient calculation of reaction rates using methods such as transition state theory and reactive flux calculations [195–198]. However, collective variables are typically not known *a priori* and require exhaustive trial-and-error to obtain ones that best describe a reaction pathway [199].

Several strategies exist for capturing rare barrier-crossing events without good CVs, one of which is transition path sampling (TPS) [200, 201]; an importance sampling technique for generating an ensemble of transition pathways. An alternative strategy is to rely on transition path theory (TPT) [202, 203], which can outline various computational methods to obtain an

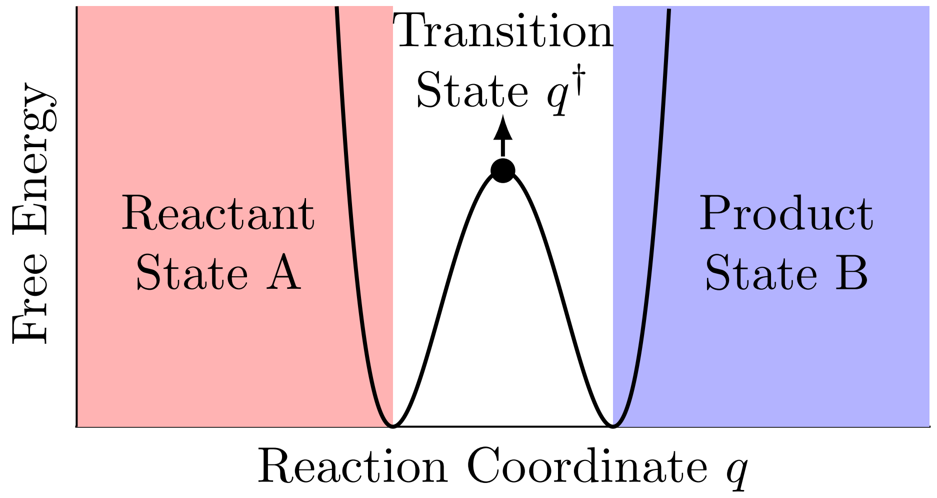


Figure 4.1: Cartoon of a potential a reaction undergoes with a one-dimensional reaction coordinate q having a reactant state A, transition state, and product state B.

average characteristic pathway, e.g., the finite-temperature string (FTS) method [204, 205]. Both strategies involve the calculation of the committor function $q(\mathbf{x})$; the probability that a trajectory starting from some initial configuration $\mathbf{x}$ enters the product state before the reactant state. The committor function can be further used to obtain reaction rates and transition-state ensembles. Its standard computation entails generating many trajectories for every initial configuration $\mathbf{x}$, which may become prohibitively expensive [206].

In this chapter, we outline the main results of transition path theory in Section 4.2, in which a framework to obtain the committor function by solving a partial differential equation is developed. The finite-temperature string method will then be derived in Section 4.3 using approximations to transition path theory. These ideas are then applied to a 2D nonconvex potential, the Müller-Brown potential, to demonstrate the ideas of transition path theory and the finite-temperature string method in Section 4.4. We conclude with some remarks on the limitations of the finite-temperature string method for higher dimensional systems in Section 4.5. These results will be used in Chapter 5 in order to obtain methods to evaluate the committor function in an efficient manner using machine learning techniques.

4.2 The Committor, Backward Kolmogorov Equation, and Reaction Quantities

To review TPT, consider a d-dimensional system with N-many particles at equilibrium that interact with a potential energy function $V(\mathbf{x})$, where $\mathbf{x} \in \Omega$ is a configuration of the system and $\Omega \subset \mathbb{R}^{Nd}$ is the configuration space. For now, we also consider the particles to have

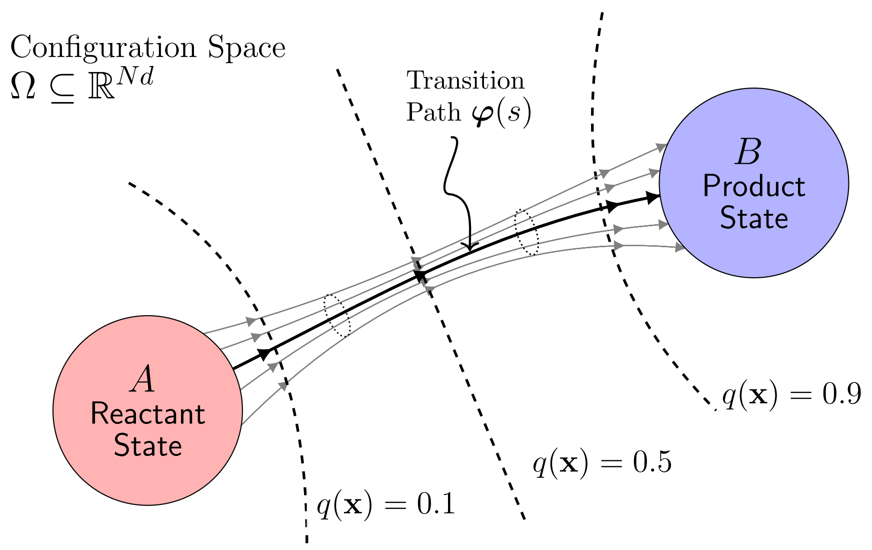


Figure 4.2: A schematic of transition path theory (TPT). Gray lines are flow lines of the probability flux $\mathbf{J}(\mathbf{x})$, and the transition tube, i.e., the region of high flux, is localized around the transition path $\boldsymbol{\varphi}(s)$. Dashed lines are isocommittor surfaces, with the middle dashed line defining the transition-state ensemble where $q(\mathbf{x}) = 0.5$.

velocities $\mathbf{v} \in \mathbb{R}^{Nd}$. Given this model system, TPT can be used to analyze the system's transition from a reactant state $A \subset \Omega$ to a product state $B \subset \Omega$ [202, 203, 207]; see Fig. 4.2 for a schematic of the problem. Central to TPT is the calculation of the committor function $q(\mathbf{x}, \mathbf{v})$, which is defined as the probability to first reach B before A given that the system initially starts at $\mathbf{x}_0 = \mathbf{x}$ and $\mathbf{v}_0 = \mathbf{v}$. The formula for $q(\mathbf{x}, \mathbf{v})$ is given by

$$q(\mathbf{x}, \mathbf{v}) = \mathbb{E}\left[h_B\left(\mathbf{x}_\tau\right) \mid \mathbf{x}_0 = \mathbf{x}, \mathbf{v}_0 = \mathbf{v}\right]; \quad \tau = \underset{t \in [0,+\infty)}{\arg\min}\{\mathbf{x}_t \in A \cup B : \mathbf{x}_0 = \mathbf{x}\}, \tag{4.2.1}$$

where $\mathbb{E}[\ldots \mid \mathbf{x}_0 = \mathbf{x}, \mathbf{v}_0 = \mathbf{v}] = \mathbb{E}^{(\mathbf{x},\mathbf{v})}$ is an average over all trajectories starting from $\mathbf{x}$ and $\mathbf{v}$, τ is the first-passage time, and the indicator function $h_C(\mathbf{x}) = 1$ gives if $\mathbf{x} \in C$ and zero otherwise.

Using stochastic calculus [208, 209], one may compute the committor as a solution to the steady-state backward Kolmogorov equation (BKE). To do so, we specify that the system is governed by the Langevin equation

$$\dot{\mathbf{x}} = \mathbf{v}\,, \tag{4.2.2}$$

$$\dot{\mathbf{v}} = -\nabla V(\mathbf{x}) - \gamma \mathbf{v} + \sqrt{2\, k_{\mathrm{B}} T \gamma}\, \boldsymbol{\eta}\,, \tag{4.2.3}$$

where γ is the friction coefficient and $\boldsymbol{\eta}$ is Gaussian white noise. Under the system of equations given by Eqs. (4.2.2) and (4.2.3), one can derive a general partial differential equation that all observables obey. To see this, let $f(\mathbf{x}(t), \mathbf{v}(t))$ be some observable computed at time t. By Itô's formula [209], we obtain

$$\mathrm{d}f = \left[\nabla_{\mathbf{x}} f \cdot \dot{\mathbf{x}} + \nabla_{\mathbf{v}} f \cdot \dot{\mathbf{v}} + \ k_{\mathrm{B}} T \nabla_{\mathbf{v}}^2 f\right] \mathrm{d}t + \sqrt{2\ k_{\mathrm{B}} T \gamma} \nabla_{\mathbf{v}} f \cdot \mathrm{d}\mathbf{B}_t\,, \tag{4.2.4}$$

where $\mathbf{B}_t$ is a Wiener process. Substitution of Eq. (4.2.3) yields

$$\mathrm{d}f = \left[\nabla_{\mathbf{x}} f \cdot \mathbf{v} + \nabla_{\mathbf{v}} f \cdot [-\nabla_{\mathbf{x}} V(\mathbf{x}) - \gamma \mathbf{v}] + \ k_{\mathrm{B}} T \nabla_{\mathbf{v}}^2 f\right] \mathrm{d}t + \sqrt{2\ k_{\mathrm{B}} T \gamma} \nabla_{\mathbf{v}} f \cdot \mathrm{d}\mathbf{B}_t \tag{4.2.5}$$

$$= \hat{L} f \,\mathrm{d}t + \sqrt{2\ k_{\mathrm{B}} T \gamma} \nabla_{\mathbf{v}} f \cdot \mathrm{d}\mathbf{B}_t\,. \tag{4.2.6}$$

Integration over time and averaging over possible trajectories yield

$$\mathbb{E}^{(\mathbf{x},\mathbf{v})}[f(\mathbf{x}(t), \mathbf{v}(t))] - f(\mathbf{x}, \mathbf{v}) = \mathbb{E}^{(\mathbf{x},\mathbf{v})}\left[\int_0^t \hat{L} f \,\mathrm{d}s\right] + \mathbb{E}^{(\mathbf{x},\mathbf{v})}\left[\int_0^t \sqrt{2\ k_{\mathrm{B}} T \gamma} \nabla_{\mathbf{v}} f \cdot \mathrm{d}\mathbf{B}_s\right]. \tag{4.2.7}$$

We will evaluate the average value of the stochastic integral first. To do so, we evaluate the general case of a function $g(\mathbf{x}(t), \mathbf{v}(t))$. Using Itô calculus and the knowledge that individual Brownian noise increments are independent with mean zero yields

$$\mathbb{E}^{(\mathbf{x},\mathbf{v})}\left[\int_0^t g(\mathbf{x}(t), \mathbf{v}(t)) dB_s\right] \approx \mathbb{E}^{(\mathbf{x},\mathbf{v})}\left[\sum_i g(\mathbf{x}(t_i), \mathbf{v}(t_i)) \left[B_{t_{i+1}} - B_{t_i}\right]\right] \tag{4.2.8}$$

$$= \sum_i \mathbb{E}^{(\mathbf{x},\mathbf{v})}\left[g(\mathbf{x}(t_i), \mathbf{v}(t_i)) \Delta B_{t_{i+1}}\right] \tag{4.2.9}$$

$$= \sum_i \mathbb{E}^{(\mathbf{x},\mathbf{v})}\left[g(\mathbf{x}(t_i), \mathbf{v}(t_i))\right] \mathbb{E}\left[\Delta B_{t_{i+1}}\right] \tag{4.2.10}$$

$$= \sum_i \mathbb{E}^{(\mathbf{x},\mathbf{v})}\left[g(\mathbf{x}(t_i), \mathbf{v}(t_i))\right] \cdot 0 = 0\,. \tag{4.2.11}$$

Hence the stochastic term in Eq. (4.2.7) evaluates to 0. Using Fubini's theorem to switch the order of time integration and averaging for the remaining terms yields

$$\mathbb{E}^{(\mathbf{x},\mathbf{v})}[f(\mathbf{x}(t), \mathbf{v}(t))] - f(\mathbf{x}, \mathbf{v}) = \int_0^t \mathbb{E}^{(\mathbf{x},\mathbf{v})}\left[\hat{L} f(\mathbf{x}(s), \mathbf{v}(s))\right] \mathrm{d}s\,. \tag{4.2.12}$$

Dividing by t and taking the limit of $t \to 0$ obtains

$$\lim_{t \to 0} \frac{\mathbb{E}^{(\mathbf{x},\mathbf{v})}[f(\mathbf{x}(t), \mathbf{v}(t))] - f(\mathbf{x}, \mathbf{v})}{t} = \lim_{t \to 0} \frac{1}{t} \int_0^t \mathbb{E}^{(\mathbf{x},\mathbf{v})}\left[\hat{L} f\right] \mathrm{d}t = \hat{L} f(\mathbf{x}, \mathbf{v})\,. \tag{4.2.13}$$

Next, we consider a function $u(t, \mathbf{x}(r), \mathbf{v}(r)) = \mathbb{E}^{(\mathbf{x}(r),\mathbf{v}(r))}[g(\mathbf{x}(t), \mathbf{v}(t)]$. By the Markov property, the expectation value at time $t + h$ is given by an iterated expectation value at time h followed by time t

$$\mathbb{E}^{(\mathbf{x},\mathbf{v})}[g(\mathbf{x}(t+h), \mathbf{v}(t+h)] = \mathbb{E}^{(\mathbf{x},\mathbf{v})}\left[\mathbb{E}^{(\mathbf{x}(h),\mathbf{y}(h))}\left[g(\mathbf{x}(t+h), \mathbf{v}(t+h)\right]\right]\,. \tag{4.2.14}$$

The time evolution of $u(t,\mathbf{x},\mathbf{v})$ can then be found as

$$\frac{\partial u(t,\mathbf{x},\mathbf{v})}{\partial t} = \lim_{h\to 0} \frac{\mathbb{E}^{(\mathbf{x},\mathbf{v})}[g(x(t+h),v(t+h))] - \mathbb{E}^{(\mathbf{x},\mathbf{v})}[g(x(t),v(t))]}{h} \tag{4.2.15}$$

$$= \lim_{h\to 0} \frac{\mathbb{E}^{(\mathbf{x},\mathbf{v})}\left[\mathbb{E}^{(\mathbf{x}(h),\mathbf{y}(h))}\left[g(\mathbf{x}(t+h),\mathbf{v}(t+h)\right]\right] - \mathbb{E}^{(\mathbf{x},\mathbf{v})}[g(x(t),v(t))]}{h} \tag{4.2.16}$$

$$= \lim_{h\to 0} \frac{\mathbb{E}^{(\mathbf{x},\mathbf{v})}\left[u(t+h,\mathbf{x}(h),\mathbf{v}(h))\right] - u(t,\mathbf{x},\mathbf{v})}{h} \tag{4.2.17}$$

$$= \hat{L}u(t,\mathbf{x},\mathbf{v})\,. \tag{4.2.18}$$

Equation (4.2.18) is known as the backward Kolmogorov equation (BKE). At steady-state, the equation reduces to

$$\hat{L}u(\mathbf{x},\mathbf{v}) = 0\,. \tag{4.2.19}$$

Using the operator $\hat{L}$ associated with the Langevin equation and letting $u(\mathbf{x},\mathbf{v}) = q(\mathbf{x},\mathbf{v})$, we obtain

$$\mathbf{v}\cdot\nabla_{\mathbf{x}}q + [-\nabla_{\mathbf{x}}V(\mathbf{x}) - \gamma\mathbf{v}]\cdot\nabla_{\mathbf{v}}q + \ k_{\mathrm{B}}T\nabla^2_{\mathbf{v}}q = 0\,. \tag{4.2.20}$$

In the overdamped limit, we can neglect $\mathbf{v}$ and obtain the commitor function in terms of $\mathbf{x}$ alone. The equations of motion in the overdamped limit are given by

$$\gamma\dot{\mathbf{x}} = -\nabla_{\mathbf{x}}V(\mathbf{x}) + \sqrt{2\ k_{\mathrm{B}}T\gamma}\boldsymbol{\eta}\,. \tag{4.2.21}$$

We can derive the corresponding BKE for the overdamped limit by either re-deriving the BKE for the overdamped Langevin equation, or in the high friction limit of the BKE for the Langevin equation. Either method will result in [210]

$$-\nabla^2_{\mathbf{x}}q(\mathbf{x}) + \beta\nabla_{\mathbf{x}}V(\mathbf{x})\cdot\nabla_{\mathbf{x}}q(\mathbf{x}) = 0\,, \tag{4.2.22}$$

subjected to the boundary conditions

$$q(\mathbf{x}) = 0,\ \mathbf{x}\in\partial A;\quad q(\mathbf{x}) = 1,\ \mathbf{x}\in\partial B\,, \tag{4.2.23}$$

where ∂A and ∂B are the boundaries of A and B respectively.

Equation (4.2.22) can be simplified by multiplication with the Boltzmann distribution $\rho(\mathbf{x}) = e^{-\beta V(\mathbf{x})}/Q$. Doing so yields

$$-\frac{e^{-\beta V(\mathbf{x})}}{Q}\nabla^2_{\mathbf{x}}q(\mathbf{x}) + \beta\frac{e^{-\beta V(\mathbf{x})}}{Q}\nabla_{\mathbf{x}}V(\mathbf{x})\cdot\nabla_{\mathbf{x}}q(\mathbf{x}) = -\nabla_{\mathbf{x}}\cdot\left[\frac{e^{-\beta V(\mathbf{x})}}{Q}\nabla_{\mathbf{x}}q(\mathbf{x})\right] = 0\,, \tag{4.2.24}$$

where one can identify

$$J(\mathbf{x}) = \frac{k_{\mathrm{B}}T}{\gamma}\rho(\mathbf{x})\nabla_{\mathbf{x}}q(\mathbf{x}) \tag{4.2.25}$$

as the probability flux of reactive trajectories at $\mathbf{x}$.

Solving Eq. (4.2.24) for the committor function allows us to evaluate quantities such as the transition path, transition-state ensembles, and reaction rates. The transition path is a curve $\boldsymbol{\varphi}(s)$ that encodes how the system on average moves from A to B in the configuration space. For every value of s, one can compute $\boldsymbol{\varphi}(s)$ self-consistently as the average configuration weighted by the flux $|\mathbf{J}(\mathbf{x})| = \frac{k_{\mathrm{B}}T}{\gamma}\rho(\mathbf{x})|\nabla_{\mathbf{x}}q(\mathbf{x})|$ at a chosen level set of the committor function $q(\mathbf{x})$, i.e.,

$$\boldsymbol{\varphi}(s) = \frac{\int_P \mathrm{d}S|\mathbf{J}(\mathbf{x})|\mathbf{x}}{\int_P \mathrm{d}S|\mathbf{J}(\mathbf{x})|} = \frac{\int_P \mathrm{d}S\rho(\mathbf{x})|\nabla_{\mathbf{x}}q(\mathbf{x})|\mathbf{x}}{\int_P \mathrm{d}S\rho(\mathbf{x})|\nabla_{\mathbf{x}}q(\mathbf{x})|}\,, \tag{4.2.26}$$

where $\int_P \mathrm{d}S$ is a surface integral over the level set $P = \{\mathbf{x} \in \Omega : q(\mathbf{x}) = q(\boldsymbol{\varphi}(s))\}$ [202, 203]. Note that for processes involving high-energy barriers the region of high flux typically forms a tubular region called the transition tube, which is localized around $\boldsymbol{\varphi}(s)$; see Fig. 4.2. The level sets of $q(\mathbf{x})$ are also referred to as the isocommittor surfaces, where the isocommittor surface corresponding to the level set $\{\mathbf{x} \in \Omega : q(\mathbf{x}) = \frac{1}{2}\}$ defines the transition-state ensemble. The reaction rate ν_R, defined as the frequency with which a system transitions from A to B, can also be evaluated from $q(\mathbf{x})$. We can evaluate ν_R as the flux through a surface in configuration space per [203, 207, 210]

$$\nu_R = \int_\Gamma \mathrm{d}S\ \mathbf{n}(\mathbf{x}) \cdot \mathbf{J}(\mathbf{x})\,, \tag{4.2.27}$$

where Γ is an arbitrary dividing surface and $\mathbf{n}(\mathbf{x})$ is the unit normal of the surface. Taking Γ to be an isocommittor surface P, one identifies the normal as

$$\mathbf{n}(\mathbf{x}) = \frac{\nabla_{\mathbf{x}}q(\mathbf{x})}{|\nabla_{\mathbf{x}}q(\mathbf{x})|}\,. \tag{4.2.28}$$

We then use the identity [211]

$$\int_P \mathrm{d}S = \int_\Omega \mathrm{d}\mathbf{x}\ \delta(q(\mathbf{x}) - q)|\nabla_{\mathbf{x}}q(\mathbf{x})|\,, \tag{4.2.29}$$

to evaluate Eq. (4.2.27) as

$$\nu_R = \int_\Omega \mathrm{d}\mathbf{x}\ \delta(q(\mathbf{x}) - q)\nabla_{\mathbf{x}}q(\mathbf{x}) \cdot \mathbf{J}\,. \tag{4.2.30}$$

To proceed, we will demonstrate that Eq. (4.2.30) is independent of the choice of q and then integrate over all possible values of q. This is seen from evaluating the partial derivative of ν_R with respect to q to yield [207, 210]

$$\frac{\partial \nu_R}{\partial q} = \int_\Omega \mathrm{d}\mathbf{x}\ \frac{\partial \delta(q(\mathbf{x}) - q)}{\partial q}\nabla_{\mathbf{x}}q(\mathbf{x}) \cdot \mathbf{J} \tag{4.2.31}$$

$$= -\int_\Omega \mathrm{d}\mathbf{x}\ \nabla_{\mathbf{x}}\delta(q(\mathbf{x}) - q) \cdot \mathbf{J} \tag{4.2.32}$$

$$= \int_\Omega \mathrm{d}\mathbf{x}\ \delta(q(\mathbf{x}) - q)\ \nabla_{\mathbf{x}} \cdot \mathbf{J} = 0\,. \tag{4.2.33}$$

The reaction rate is then evaluated as

$$\nu_R = \int_0^1 \mathrm{d}q \; \nu_R = \int_\Omega \mathrm{d}\mathbf{x} \; \nabla_\mathbf{x} q(\mathbf{x}) \cdot \mathbf{J} \tag{4.2.34}$$

$$= \frac{k_\mathrm{B}T}{\gamma} \int_\Omega \mathrm{d}\mathbf{x} \; \rho(\mathbf{x}) |\nabla_\mathbf{x} q(\mathbf{x})|^2 = \frac{k_\mathrm{B}T}{\gamma} \left\langle |\nabla_\mathbf{x} q(\mathbf{x})|^2 \right\rangle \, . \tag{4.2.35}$$

4.3 The Finite-Temperature String Method

The BKE given by Eq. (4.2.24) is difficult to solve as it corresponds to a partial differential equation in high dimensional space Nd. While the committor function is important on its own, information about the kinetic process of interest can be gained by only looking at quantities defined by $q(\mathbf{x})$ and not $q(\mathbf{x})$ itself. To do so, we will devise a numerical method to obtain the transition path $\boldsymbol{\varphi}(s)$ directly from an approximation of the committor function. One of the most popular methods to do so is the finite-temperature string (FTS) method [204, 205]. The FTS method is a numerical method to obtain transition paths, assuming that the probability flux across region A to B attains a highly-peaked and unique transition tube across different committor values. In this section, we will review the FTS method [204, 205].

The FTS method is an algorithm for obtaining a transition path $\boldsymbol{\varphi}(s)$, as defined in Eq. (4.2.26), using sampling and optimization techniques. It emerges from an approximation of the committor function $q(\mathbf{x})$ that is locally built around the transition path $\boldsymbol{\varphi}(s)$. This local approximation is achieved by constructing suitable functions $s_\gamma(\mathbf{x})$, which represent isocommittor surfaces as hyperplanes centered around $\boldsymbol{\varphi}(s)$. If $\boldsymbol{\varphi}(s)$ follows an arc-length parameterization, where s is the arc-length, the approximation for $q(\mathbf{x})$ and the formula for $s_\gamma(\mathbf{x})$ can be written as

$$q(\mathbf{x}) \approx f(s_\gamma(\mathbf{x})) \, , \tag{4.3.1}$$

$$s_\gamma(\mathbf{x}) \equiv \underset{s\in[0,L]}{\arg\min} \, \frac{1}{2} |\mathbf{x} - \boldsymbol{\varphi}(s)|^2 \, , \tag{4.3.2}$$

where L is the total arc-length of the path, and $f : [0, L] \to [0, 1]$ is an invertible scalar function. To see that the function $s_\gamma(\mathbf{x})$ approximates isocommittor surfaces as hyper-planes, one may perform the minimization in Eq. (4.3.2) to obtain the following equation:

$$\frac{\mathrm{d}\boldsymbol{\varphi}(s)}{\mathrm{d}s} \cdot (\mathbf{x} - \boldsymbol{\varphi}(s)) = 0 \, , \tag{4.3.3}$$

which is a linear equation in $\mathbf{x}$, indicating the set of all configurations satisfying Eq. (4.3.3) for fixed value of $s \in [0, L]$ is a hyperplane; see Fig. 4.3 for illustration. On the other hand, the operation of fixing a configuration $\mathbf{x}$, and finding s that satisfies Eq. (4.3.3) defines a mapping between configurations $\mathbf{x} \in \Omega$ and the variable $s \in [0, L]$. This mapping is what we denote as $s_\gamma(\mathbf{x})$.

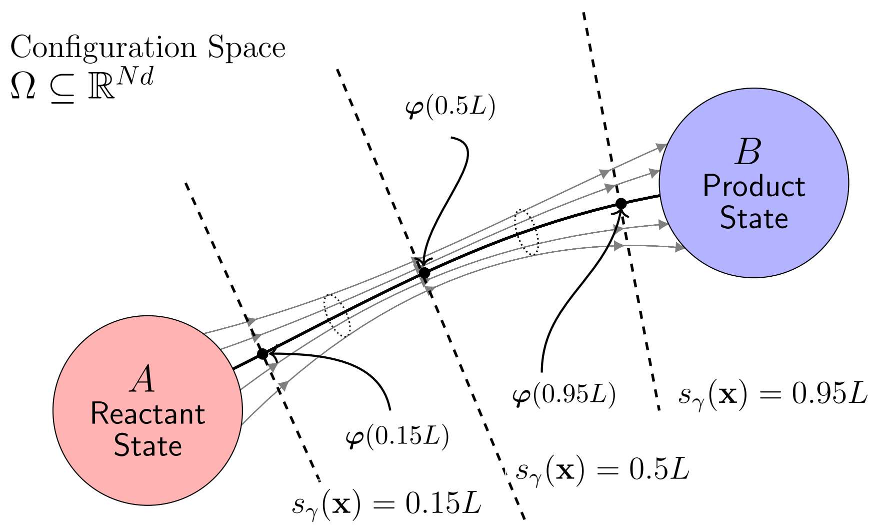

Configuration Space
$\Omega \subseteq \mathbb{R}^{Nd}$
A
Reactant State
B
Product State
$\boldsymbol{\varphi}(0.5L)$
$\boldsymbol{\varphi}(0.15L)$
$\boldsymbol{\varphi}(0.95L)$
$s_\gamma(\mathbf{x}) = 0.95L$
$s_\gamma(\mathbf{x}) = 0.5L$
$s_\gamma(\mathbf{x}) = 0.15L$

Figure 4.3: The local approximation of isocommittor surfaces as hyper-planes, which also correspond to the level sets of $s_\gamma(\mathbf{x})$. The normal vector of each hyper-plane is the tangent vector $\frac{\mathrm{d}\boldsymbol{\varphi}(s)}{\mathrm{d}s}$.

Given $s_\gamma(\mathbf{x})$ in Eq. (4.3.2), the problem of finding $\boldsymbol{\varphi}(s)$ can be posed as an optimization problem. To this end, using Eq. (4.3.1), Eq. (4.2.26) can be approximated as an integral over the hyperplane defined by $s_\gamma(\mathbf{x})$:

$$\boldsymbol{\varphi}(s) \approx \frac{\int_{\tilde{P}} \mathrm{d}S \ \rho(\mathbf{x}) f'(s_\gamma(\mathbf{x}))|\nabla_{\mathbf{x}} s_\gamma(\mathbf{x})|\mathbf{x}}{\int_{\tilde{P}} \mathrm{d}S \ \rho(\mathbf{x}) f'(s_\gamma(\mathbf{x}))|\nabla_{\mathbf{x}} s_\gamma(\mathbf{x})|}\,, \tag{4.3.4}$$

where $\tilde{P}$ is a level set of the function $s_\gamma(\mathbf{x})$ given by $\tilde{P} = \{\mathbf{x} \in \Omega : s_\gamma(\mathbf{x}) = s\}$. Since $f'(s_\gamma(\mathbf{x}))$ is constant over the level set $\tilde{P}$, Eq. (4.3.4) can be rewritten as

$$\boldsymbol{\varphi}(s) \approx \frac{\int_{\tilde{P}} \mathrm{d}S \ \rho(\mathbf{x})|\nabla_{\mathbf{x}} s_\gamma(\mathbf{x})|\mathbf{x}}{\int_{\tilde{P}} \mathrm{d}S \ \rho(\mathbf{x})|\nabla_{\mathbf{x}} s_\gamma(\mathbf{x})|}\,. \tag{4.3.5}$$

Using Eq. (4.2.29), Eq. (4.3.5) can be rewritten as

$$\boldsymbol{\varphi}(s) \approx \frac{\int_{\Omega} \mathrm{d}\mathbf{x} \ \rho(\mathbf{x})\delta(s_\gamma(\mathbf{x}) - s)|\nabla_{\mathbf{x}} s_\gamma(\mathbf{x})|^2\mathbf{x}}{\int_{\Omega} \mathrm{d}\mathbf{x} \ \rho(\mathbf{x})\delta(s_\gamma(\mathbf{x}) - s)|\nabla_{\mathbf{x}} s_\gamma(\mathbf{x})|^2} = \frac{\langle \delta(s_\gamma(\mathbf{x}) - s)|\nabla_{\mathbf{x}} s_\gamma(\mathbf{x})|^2\mathbf{x}\rangle}{\langle \delta(s_\gamma(\mathbf{x}) - s)|\nabla_{\mathbf{x}} s_\gamma(\mathbf{x})|^2\rangle}\,. \tag{4.3.6}$$

Furthermore, assuming the path's curvature to be small, which implies that $|\nabla_{\mathbf{x}} s_\gamma(\mathbf{x})|^2 \approx 1$ (see Appendix A of Ref. [205] for a proof), Eq. (4.3.6) can be simplified into a conditional average given by

$$\boldsymbol{\varphi}(s) \approx \frac{\langle \delta(s_\gamma(\mathbf{x}) - s)\mathbf{x}\rangle}{\langle \delta(s_\gamma(\mathbf{x}) - s)\rangle} = \langle \mathbf{x} \mid s_\gamma(\mathbf{x}) = s\rangle\,. \tag{4.3.7}$$

Lastly, one may use variational techniques to show that Eq. (4.3.7) is the result of extremizing the following functional [205, 212]:

$$C[\boldsymbol{\varphi}] = \int_0^L \mathrm{d}s \, \left\langle \frac{1}{2} |\boldsymbol{\varphi}(s) - \mathbf{x}|^2 \delta(s_\gamma(\mathbf{x}) - s) \right\rangle \tag{4.3.8}$$

such that

$$\left| \frac{\mathrm{d}\boldsymbol{\varphi}(s)}{\mathrm{d}s} \right| = 1 \,. \tag{4.3.9}$$

Equation (4.3.9) is the definition of arc-length parameterization, which sets a constraint on the possible paths that extremize Eq. (4.3.8).

Equations (4.3.8) and (4.3.9) form the starting points for developing the FTS method, with several discretization and approximation steps leading to a solvable optimization problem. To this end, discretizing $\boldsymbol{\varphi}(s)$ into a set of equidistant nodal points $\{\boldsymbol{\varphi}^\alpha\}_{\alpha=1}^M$, satisfying Eq. (4.3.9), i.e., $|\boldsymbol{\varphi}^{\alpha+1} - \boldsymbol{\varphi}^\alpha| = |\boldsymbol{\varphi}^\alpha - \boldsymbol{\varphi}^{\alpha-1}|$, $\forall \alpha \in \{1, \ldots, M\}$, Eq. (4.3.8) can be approximated as

$$C(\{\boldsymbol{\varphi}^\alpha\}) = \sum_{\alpha=1}^M \Delta s \left\langle \frac{1}{2} |\boldsymbol{\varphi}^\alpha - \mathbf{x}|^2 \delta(s_\gamma(\mathbf{x}) - s_\alpha) \right\rangle \,, \tag{4.3.10}$$

where $s_\alpha = \left(\frac{\alpha-1}{M-1}\right) L$ is the arc-length of the path up to node $\boldsymbol{\varphi}^\alpha$, and Δs is the arc-length between any two nodes. Furthermore, the Dirac delta function $\delta(s_\gamma(\mathbf{x}) - s_\alpha)$ can be approximated with an indicator function (see Appendix B of Ref. [205]):

$$h_{R_\alpha}(\mathbf{x}) = \begin{cases} \frac{1}{\Delta s} & \mathbf{x} \in R_\alpha(\{\boldsymbol{\varphi}^\alpha\}) = \{\mathbf{x} \in \Omega : |\mathbf{x} - \boldsymbol{\varphi}^\alpha| < |\mathbf{x} - \boldsymbol{\varphi}^{\alpha'}| \;\; \forall \alpha' \neq \alpha\} \\ 0 & \text{otherwise} \end{cases} \,, \tag{4.3.11}$$

where R_α denotes a Voronoi cell centered at node $\boldsymbol{\varphi}^\alpha$. With these steps, Eq. (4.3.10) can then be expressed as a least-squares function:

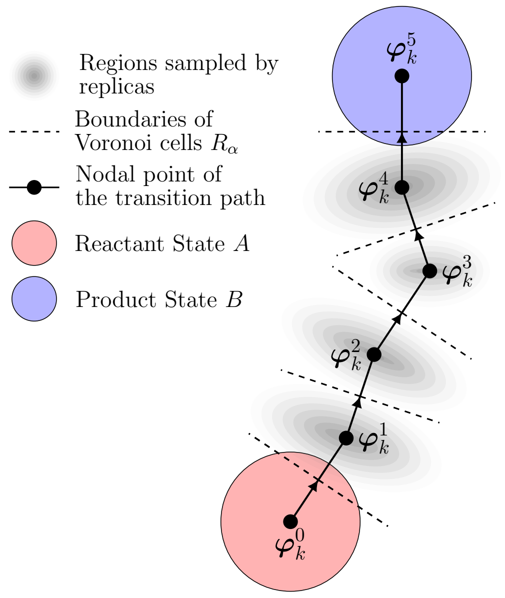
$$C(\{\boldsymbol{\varphi}^\alpha\}) = \sum_{\alpha=1}^M \Delta s \left\langle \frac{1}{2} |\boldsymbol{\varphi}^\alpha - \mathbf{x}|^2 h_{R_\alpha}(\mathbf{x}) \right\rangle = \sum_{\alpha=1}^M \left\langle \frac{1}{2} |\boldsymbol{\varphi}^\alpha - \mathbf{x}|^2 \right\rangle_{R_\alpha(\{\boldsymbol{\varphi}^\alpha\})} \tag{4.3.12}$$

where $\langle \ldots \rangle_{R_\alpha(\{\boldsymbol{\varphi}^\alpha\})}$ is an ensemble average constrained inside a Voronoi cell.

The ensemble averages in Eq. (4.3.12) can be estimated as averages over samples obtained from molecular simulations, which are constrained to be inside the Voronoi cells and are initiated with the configuration of the corresponding node. As illustrated in Fig. 4.4(left), this step involves introducing M-many replicas of the system to sample configurations within each of the M-many Voronoi cells, where each replica can evolve according to discrete overdamped Langevin dynamics with a rejection rule:

$$\mathbf{x}_\star^\alpha = \mathbf{x}_t^\alpha - \gamma^{-1} \nabla_{\mathbf{x}} V(\mathbf{x}_t^\alpha) \Delta t + \sqrt{2\Delta t \, k_\mathrm{B} T \gamma^{-1}} \mathbf{w}_t^\alpha \,, \tag{4.3.13}$$

$$\mathbf{x}_{t+1}^\alpha = \begin{cases} \mathbf{x}_\star^\alpha & \text{if } \mathbf{x}_\star^\alpha \in R_\alpha \\ \mathbf{x}_t^\alpha & \text{otherwise} \end{cases} \,, \tag{4.3.14}$$


Regions sampled by replicas
Boundaries of Voronoi cells R_α
Nodal point of the transition path
Reactant State A
Product State B
$\boldsymbol{\varphi}_k^0$
$\boldsymbol{\varphi}_k^1$
$\boldsymbol{\varphi}_k^2$
$\boldsymbol{\varphi}_k^3$
$\boldsymbol{\varphi}_k^4$
$\boldsymbol{\varphi}_k^5$

Algorithm 3: The FTS Method

```
Data: Initial conditions $\{\boldsymbol{\varphi}_0^\alpha\}$. The FTS Method step size $\Delta\tau$ and penalty strength $\lambda_S$.
1 for $k = 0, \dots, K$ do
2     for $\alpha = 1, \dots, M$ in parallel do
3         for $m = 1, \dots, |\mathcal{R}_k^\alpha|$ do
4             Sample $\mathbf{x}_m^\alpha$ with MD/MC simulation constrained in the Voronoi cell $R_\alpha(\{\boldsymbol{\varphi}_k^\alpha\})$, e.g., Eqs. (4.3.13) and (4.3.14).
5             Store $\mathbf{x}_m^\alpha$ in batch $\mathcal{R}_k^\alpha$.
6     $\boldsymbol{\varphi}_\alpha^{k+1} \leftarrow$ Eqs. (4.3.18) and (4.3.19).
```

Figure 4.4: (Left) An illustration of the FTS method, where each replica samples configurations inside a Voronoi cell. (Right) Pseudo-code for the FTS method.

where $\mathbf{w}_t^\alpha$ is a random variable with zero-mean and unit variance. Note that Eq. (4.3.13) can be replaced with an MC step. Introducing $\mathcal{R}^\alpha$ as the batch of samples obtained from the α-th replica, Eq. (4.3.12) can be estimated as

$$\hat{C}(\{\boldsymbol{\varphi}^\alpha\};\{\mathcal{R}^\alpha\}) = \sum_{\alpha=1}^{M} \frac{1}{|\mathcal{R}^\alpha|} \sum_{\mathbf{x}\in\mathcal{R}^\alpha} \frac{1}{2}|\boldsymbol{\varphi}^\alpha - \mathbf{x}|^2 \,. \tag{4.3.15}$$

To avoid large displacements in neighboring nodal points, a penalty function is added to Eq. (4.3.15), which yields

$$\hat{C}(\{\boldsymbol{\varphi}^\alpha\};\{\mathcal{R}^\alpha\}) = \sum_{\alpha=1}^{M} \frac{1}{|\mathcal{R}^\alpha|} \sum_{\mathbf{x}\in\mathcal{R}^\alpha} \frac{1}{2}|\boldsymbol{\varphi}^\alpha - \mathbf{x}|^2 + \frac{\lambda_S}{2}\sum_{\alpha=1}^{M-1} |\boldsymbol{\varphi}^{\alpha+1} - \boldsymbol{\varphi}^\alpha|^2 \,, \tag{4.3.16}$$

$$\text{s.t.} \quad |\boldsymbol{\varphi}^{\alpha+1} - \boldsymbol{\varphi}^\alpha| = |\boldsymbol{\varphi}^{\alpha} - \boldsymbol{\varphi}^{\alpha-1}| \,, \tag{4.3.17}$$

where λ_S is the penalty strength.

The FTS method minimizes Eq. (4.3.16) using a closed feedback loop between the replica dynamics, e.g., Eqs. (4.3.13) and (4.3.14), and a modified gradient-descent step. At the k-th iteration of the loop, replicas generate a collection of batches $\{\mathcal{R}_k^\alpha\}_{\alpha=1}^M$, where the batch $\mathcal{R}_k^\alpha$ consists of a short molecular dynamics (MD)/Monte Carlo (MC) trajectory run from the α-th replica. This data is then used in a two-part gradient descent update, where the first part corresponds to the following update:

$$\boldsymbol{\varphi}_\star^\alpha = \boldsymbol{\varphi}_k^\alpha - \Delta\tau\nabla_{\boldsymbol{\varphi}^\alpha}\hat{C}(\{\boldsymbol{\varphi}_k^\alpha\};\{\mathcal{R}_k^\alpha\}) \,, \tag{4.3.18}$$

with $\Delta\tau$ the step size. Note that one can replace Eq. (4.3.18) with an implicit update for increased stability or a momentum-variant, such as the Heavy-Ball [213] and the Nesterov method [214], for accelerated convergence. The second part enforces the constraint Eq. (4.3.17) with a reparameterization of the path using linear interpolation:

$$\boldsymbol{\varphi}_{k+1}^{\alpha} = \boldsymbol{\varphi}_{\star}^{a(\alpha)-1} + \left(L_M \frac{\alpha-1}{M-1} - L_{a(\alpha)-1}\right) \frac{\boldsymbol{\varphi}_{\star}^{a(\alpha)} - \boldsymbol{\varphi}_{\star}^{a(\alpha)-1}}{\left|\boldsymbol{\varphi}_{\star}^{a(\alpha)} - \boldsymbol{\varphi}_{\star}^{a(\alpha)-1}\right|}, \tag{4.3.19}$$

where $L_\alpha = \sum_{\alpha'=2}^{\alpha} |\boldsymbol{\varphi}_{\star}^{\alpha'} - \boldsymbol{\varphi}_{\star}^{\alpha'-1}|$ is the length of the path up to node $\boldsymbol{\varphi}_{\star}^{\alpha}$, and $a(\alpha) \in \{1, \ldots, M\}$ is an index such that $L_{a(\alpha)-1} < \left(\frac{\alpha-1}{M-1}\right) L_M < L_{a(\alpha)}$. This process is repeated until convergence is achieved, yielding the transition path $\boldsymbol{\varphi}(s)$. The pseudocode for the FTS method in its entirety is given in Algorithm 3.

4.4 Numerical Examples

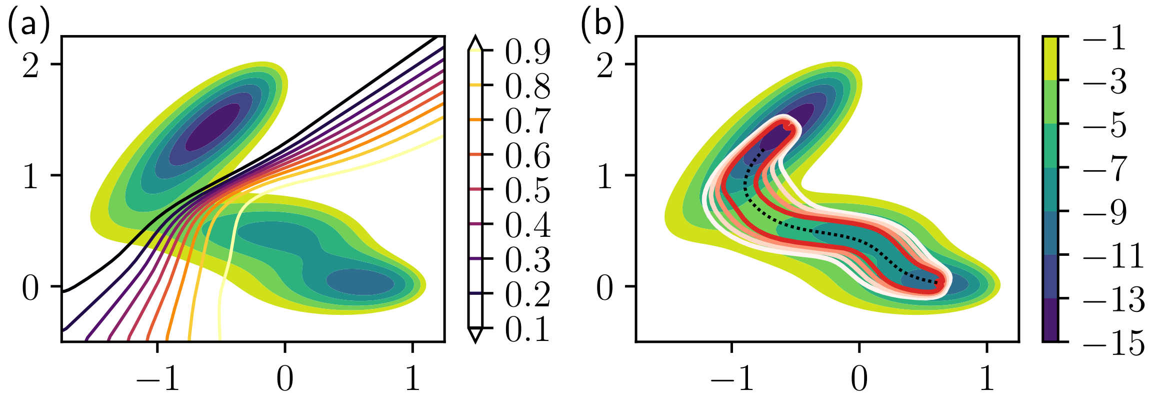


Figure 4.5: (a) MB potential along with isocommittor lines from the FEM solution. Note the MB contours correspond to βV_{MB}. (b) MB potential along with contours indicating the lines of increasing flux $\mathbf{J}(\mathbf{x})$ from white to red along with the transition path in black, computed using the FEM solution via Eq. (4.2.26).

In this section, we perform demonstrative calculations on a 2D single particle system in which the committor function, reaction rate, and transition path can be computed numerically to high precision. We will use a particle subject to the 2D Müller-Brown (MB) potential

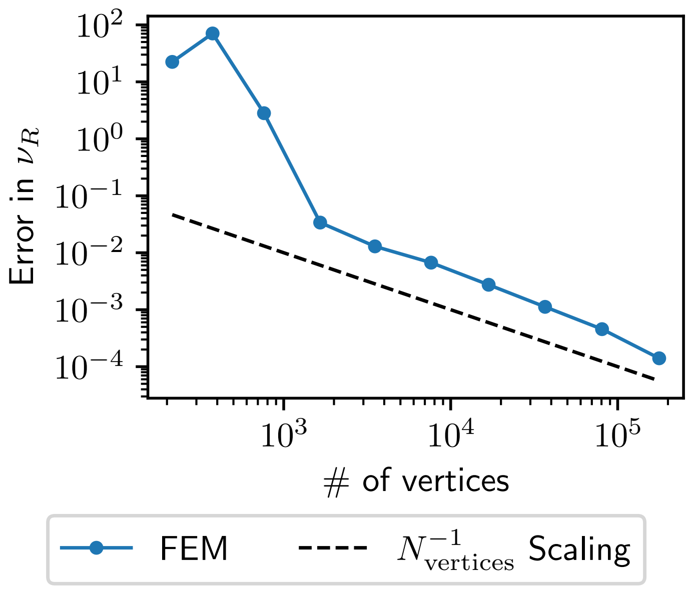


Figure 4.6: The relative error in the reaction rate of the MB potential compared to a solution with 386671 vertices for FEM solutions with a varying number of vertices.

[215], which is a Gaussian mixture potential given by

$$V_{\text{MB}}(\mathbf{x}) = \sum_{k=1}^{4} A_k \exp\left(a_i\,(x-\bar{x}_i)^2 + b_i\,(x-\bar{x}_i)\,(y-\bar{y}_i) + c_i\,(y-\bar{y})^2\right)\,, \tag{4.4.1}$$
$$A = (-200, -100, -170, 15), \quad a = (-1, -1, -6.5, -0.7)\,,$$
$$b = (0, 0, 11, 0.6), \quad c = (-10, -10, -6.5, 0.7)\,,$$
$$\bar{x} = (1, 0, -0.5, -1), \quad \bar{y} = (0, 0.5, 1.5, 1)\,.$$

It has two minima at $\mathbf{x}_0^{\text{A}} \approx (-0.558, 1.442)$ and $\mathbf{x}_0^{\text{B}} \approx (0.623, 0.028)$. In what follows, we study this model at a temperature where $k_{\text{B}}T = 10$ and friction coefficient $\gamma = 1$. While an analytical form of $q(\mathbf{x})$ for the MB potential is unknown, we use the finite element method (FEM) to numerically solve the BKE (Eq. (4.2.24)) via FEniCS [216, 217], and obtain a solution to the committor function $q_{\text{FEM}}(\mathbf{x})$ with first-order Lagrange elements. This is done on the domain $\Omega = [-1.75, 1.25] \times [-0.5, 2.25]$, with the reactant and product states defined by $A = \{\mathbf{x} \in \Omega : |\mathbf{x} - \mathbf{x}_0^{\text{A}}| < 0.025\}$ and $B = \{\mathbf{x} \in \Omega : |\mathbf{x} - \mathbf{x}_0^{\text{B}}| < 0.025\}$, respectively. The FEM solution is obtained by applying Dirichlet boundary conditions as per Eq. (4.2.23) along with a zero-flux Neumann boundary condition on $\partial\Omega$, and on a mesh with 386671 vertices. Contours of the MB potential along with isocommittor lines of $q_{\text{FEM}}(\mathbf{x})$ are shown in Fig. 4.5(a), along with contours of increasing flux and the transition path in Fig. 4.5(b).

An estimate of the reaction rate ν_R over Ω is obtained by evaluating Eq. (4.2.35) with

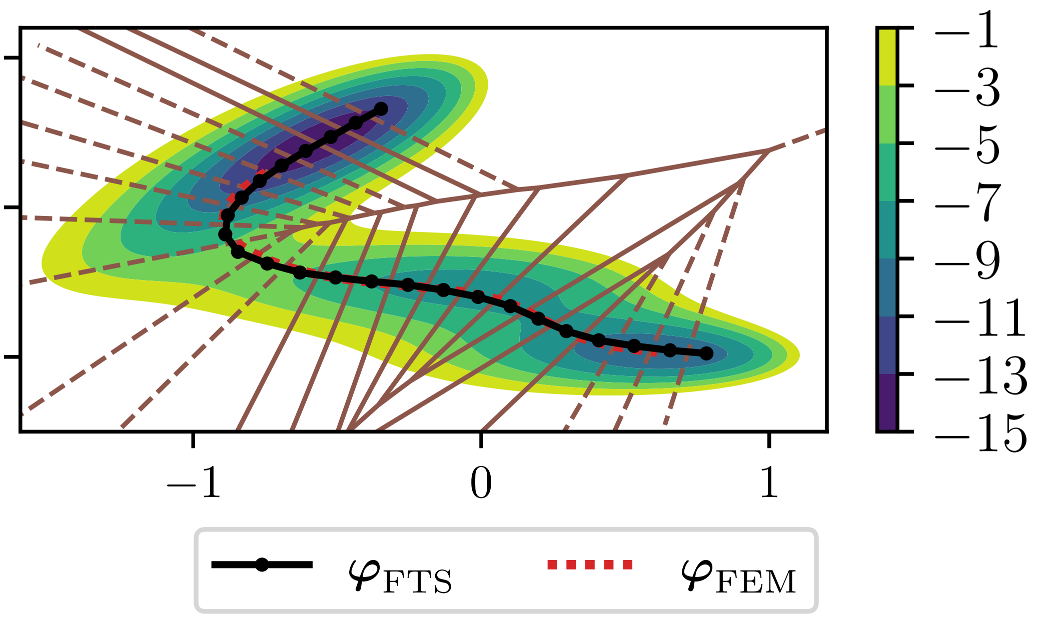


Figure 4.7: Transition paths obtained using FEM and the FTS method for the MB potential, with the boundaries of the Voronoi cells corresponding to $\boldsymbol{\varphi}_{\text{FTS}}$ given in brown.

$q_{\text{FEM}}(\mathbf{x})$ to obtain

$$\nu_{R,\text{FEM}} = \left\langle \frac{k_{\text{B}}T}{\gamma} |\nabla_{\mathbf{x}} q_{\text{FEM}}|^2 \right\rangle \approx 4.91 \cdot 10^{-3} \,. \tag{4.4.2}$$

The relative error of the solution with 386671 vertices to solutions of lower quality is shown in Fig. 4.6, demonstrating the expected linear convergence as the number of elements increases [3]. We obtain a comparison to direct simulation by running unbiased trajectories of the MB potential under discrete overdamped Langevin dynamics of the form

$$\mathbf{x}_{t+1} = \mathbf{x}_t - \gamma^{-1} \nabla_{\mathbf{x}} V(\mathbf{x}_t) \Delta t + \sqrt{2\Delta t \; k_{\text{B}} T \gamma^{-1}} \mathbf{w}_t \,. \tag{4.4.3}$$

The reaction rate is measured by averaging the inverse of the time it takes for 16000 trajectories to go from $\mathbf{x}_0^{\text{A}}$ to $\mathbf{x}_0^{\text{B}}$ and back to $\mathbf{x}_0^{\text{A}}$. This obtains

$$\nu_{R,\text{DS}} = (5.38 \pm 0.08) \cdot 10^{-3} \,, \tag{4.4.4}$$

in close agreement with Eq. (4.4.2).

We now evaluate the transition path for the MB potential. The transition path for the FEM solution, $\boldsymbol{\varphi}_{\text{FEM}}(s)$ is obtained through Eq. (4.2.26). The transition path from the FTS method, $\boldsymbol{\varphi}_{\text{FTS}}$, is obtained using 24 replicas with $|\mathcal{R}_k^{\alpha}| = 64$. With the FTS method, the penalty strength is set to $\lambda_{\text{S}} = 0.1M$, and Eq. (4.3.18) is done using the Nesterov method

[214] with the step size $\Delta\tau = 0.05$ and momentum coefficient $\mu = 0.9$ (see Eqs. (B.2.3) and (B.2.4) for the implementation of the Nesterov method). Figure 4.7 shows the transition paths, which agree closely with each other. The major observed difference is in the ends of $\boldsymbol{\varphi}_{\mathrm{FTS}}$ extending beyond $\boldsymbol{\varphi}_{\mathrm{FEM}}$ and the corresponding $q(\mathbf{x}) = 0$ and 1 isocommittor surfaces, which is allowed in the FTS method.

4.5 Limitations of the FTS Method

Though the FTS method is a strong framework to evaluate transition paths, it does have limitations. For instance, the application of the FTS method to molecular systems may fail since the distance metrics defining the Voronoi cells are not invariant with respect to rigid-body transformations. As a result, replicas can escape from their respective Voronoi cells without any structural change via rotations and/or translations alone. To resolve this issue, the FTS method is typically applied in the space of collective variables (CVs), which are invariant under translation and rotation by construction. While a solution independent of CVs remains an open problem, the work in Ref. [205] proposes a sufficiently general CV, denoted as Θ, if the system configuration $\mathbf{x}$ can be divided into a sub-system configuration $\mathbf{x}_{\mathrm{S}}$ that undergoes the structural change and solvent degrees of freedom $\mathbf{x}_{\mathrm{E}}$ that make up the surrounding environment. This CV takes $\mathbf{x}_{\mathrm{S}}$ and a string nodal point $\boldsymbol{\varphi}^{\alpha}$ as input, and it can be written as

$$\Theta(\mathbf{x}_{\mathrm{S}}; \mathbf{R}^*, \mathbf{b}^*) = \mathbf{R}^* \left(\mathbf{x}_{\mathrm{S}} - \mathbf{b}^*\right) , \tag{4.5.1}$$

$$(\mathbf{R}^*, \mathbf{b}^*) = \underset{(\mathbf{R},\mathbf{b})}{\arg\min} \left|\mathbf{R}\left(\mathbf{x}_{\mathrm{S}} - \mathbf{b}\right) - \boldsymbol{\varphi}^{\alpha}\right| , \tag{4.5.2}$$

where $\mathbf{R}^*$ and $\mathbf{b}^*$ are a rotation matrix and translation vector, respectively, that form a rigid body transformation of the sub-system. By minimizing the distance metric in Eq. (4.5.2), the chosen rigid transformation has the effect of matching the center-of-mass and orientation axis of $\mathbf{x}_{\mathrm{S}}$ to that of $\boldsymbol{\varphi}^{\alpha}$. This results in a CV that not only retains some of the original molecular degrees of freedom but also removes the degeneracy due to translations and rotations. The transformation defined by Eq. (4.5.2) can also be done at a relatively low computational cost by translating the sub-system to match its center of mass with the center of mass of $\boldsymbol{\varphi}^{\alpha}$ and subsequently rotating the sub-system via the Kabsch algorithm [218]. Other CVs are also possible and may be needed when dealing with rare-event problems where the system cannot be subdivided, e.g., nucleation and self-assembly.

Despite the generality of Eq. (4.5.1), it may not be sufficient in cases such as having high solvent densities where the solvent molecules/particles move in a highly correlated fashion during the transition, i.e., solvent reorganization, or where additional CVs are required to resolve the transition. In such cases, we note the committor values as a function of $\mathbf{x}_{\mathrm{S}}$ alone are insufficient as the solvent degrees of freedom have significant influence over $q(\mathbf{x})$. In the following chapter, we will demonstrate a method capable of solving the BKE in high-dimensional systems using sampling and machine learning techniques. This will allow us to resolve the issues associated with CVs as mentioned in this section, as the method will be capable of generating a CV-independent representation of the committor function that can use data collected via sampling methods like the FTS method that may rely on collective variables.

Chapter 5

Committor Learning

5.1 Introduction

Having reviewed transition path theory (TPT) in Chapter 4, we will now consider methods for solving the backward Kolmogorov equation using machine learning tools. In the framework of TPT, the committor function can be computed by solving a high-dimensional partial differential equation (PDE) in configuration space, called the backward Kolmogorov equation (BKE) [202, 203, 219]. The complexity in solving the high-dimensional BKE may be reduced by constructing a low-dimensional set of collective variables (CVs) [220], but they are not known *a priori* and require exhaustive trial-and-error to obtain ones that best describe a reaction pathway [199]. On the other hand, one does not need to solve the BKE over the entire configuration space to obtain reaction rates and transition-state ensembles by focusing on the important regions across the transition path. One way to target these regions is importance sampling [39] where molecular simulations are biased to generate configurations according to target values of the committor function in regions across the transition path. However, since the committor function has no closed-form expression as a function of configuration $\mathbf{x}$ and intrinsically involves averages over finite-time trajectories, it is impractical to use it in conjunction with existing importance sampling techniques. Modern machine learning (ML) approaches can alleviate this issue by representing committor functions via artificial neural networks. This is the strategy used in recent work [10] to create an ML algorithm that adopts a feedback loop between importance sampling and neural network training, which involves minimizing a loss function derived from the BKE. The feedback loop uses the neural network to acquire high-quality data from short molecular dynamics (MD) or Monte Carlo (MC) simulations via umbrella sampling [129] where a bias potential built from the neural network enhances sampling of the transition state. However, as will be shown in this work, umbrella sampling poorly explores regions across the transition path, which may result in an inaccurate computation of committor functions and thereby inaccurate, high-variance estimates of the reaction rates. This issue may be mitigated by a careful fine-tuning of the parameters used in umbrella sampling, which is a non-trivial task, or increasing the number of samples used during training, which may require long molecular simulations to reach the desired accuracy. Furthermore, the bias potential built from the neural network can lead to prohibitively expensive simulation due to the non-local many-body nature and size of the neural network.

In this chapter, we improve the algorithm in Ref. [10] to increase its accuracy. The accuracy is evaluated by computing the error in the committor function and reaction rate, with both errors evaluated between the neural network and a solution of the BKE computed either using analytical methods or the finite element method with a fine resolution for low-dimensional problems. We show that accuracy in committor functions can be improved by adding elements of supervised learning, where the neural network is trained on estimates of committor values generated via short trajectories. Accuracy in reaction rates can be improved by replacing the committor-based umbrella sampling with the FTS method [205], which samples configurations homogeneously across the transition path, and enables accurate low-variance on-the-fly estimation of reaction rates. The resulting algorithm with the FTS method is also amenable to error analysis, enabling accurate estimation of reaction rates with a lower number of samples. We also demonstrate the applicability of this method to a molecular system with a high-dimensional configuration space and demonstrate that accurate computations of the committor function and reaction rate can be obtained.

This chapter is organized as follows: in Section 5.2, we show how the BKE can be cast as an optimization problem amendable to machine learning methods. In Section 5.3, we review the ML algorithm proposed in Ref. [10] and describe how it uses umbrella sampling with feedback loops. We propose modifications to this algorithm starting with the addition of supervised learning elements in Section 5.4 and ending and use of the FTS method for importance sampling in Section 5.5. In Sections 5.6.1 and 5.6.2, we test all algorithms to problems corresponding to a particle diffusing in non-convex potential energies, showcasing how our modifications lead to a more accurate low-variance computation of the committor function and reaction rates. In Section 5.6.3, we provide an error analysis for algorithms that use the FTS method, demonstrating that the sampling distribution of the estimated reaction rates obeys a log-normal distribution, which can be used to remove the sampling error in these estimates. In Section 5.7, we apply the algorithms to a molecular system, i.e., a solvated dimer undergoing a transition between a compact to an extended state, and find the previously seen trends in low-dimensional systems to be applicable to such a high-dimensional system.

5.2 From Transition Path Theory to Machine Learning

Having reviewed TPT in Section 4.2, we will now use variational calculus to derive a form amendable to machine learning techniques. The BKE, which is a high-dimensional PDE, is infeasible to solve via standard finite difference/elements for large molecular systems, as the number of grid points/elements grows exponentially with system size N. However, it is in these situations that methods inspired by ML may hold a feasible alternative, where the committor function can be approximated by a neural network whose model parameters can be solved by transforming the BKE into an optimization problem [8–10, 221]. To this end, we begin by constructing a variational form of the BKE. Following the standard procedure for elliptic PDEs [57], we consider a variation of the committor function $\delta q(\mathbf{x})$, which obeys the constraints $\delta q(\mathbf{x}) = 0$ for $\mathbf{x} \in \partial A$ and $\mathbf{x} \in \partial B$ to satisfy the boundary conditions in Eq. (4.2.23). Multiplying Eq. (4.2.24) by $\delta q(\mathbf{x})$, integrating over $\Omega \setminus A \cup B$, and then integrating by parts yields

$$\int_{\Omega\setminus A\cup B} \mathrm{d}\mathbf{x}\; \delta q(\mathbf{x}) \nabla_{\mathbf{x}} \left[\rho(\mathbf{x}) \nabla_{\mathbf{x}} q(\mathbf{x})\right] = -\int_{\Omega\setminus A\cup B} \mathrm{d}\mathbf{x}\; \rho(\mathbf{x})\; \nabla_{\mathbf{x}} \delta q(\mathbf{x}) \cdot \nabla_{\mathbf{x}} q(\mathbf{x}) = 0\,. \tag{5.2.1}$$

Applying Vainberg's theorem [57] to Eq. (5.2.1) leads to the following functional:

$$L\left[\tilde{q}\right] = \frac{1}{2} \int_{\Omega\setminus A\cup B} \mathrm{d}\mathbf{x} \rho(\mathbf{x}) |\nabla_{\mathbf{x}} \tilde{q}(\mathbf{x})|^2 = \frac{1}{2} \left\langle |\nabla_{\mathbf{x}} \tilde{q}(\mathbf{x})|^2 \right\rangle_{\Omega\setminus A\cup B} \tag{5.2.2}$$

whose extremization over the space of admissible functions $\tilde{q}(\mathbf{x})$ subject to boundary conditions Eq. (4.2.23) leads to the solution of the BKE. The variational form in Eq. (5.2.2) therefore transforms the strong form of BKE into a problem of functional optimization, where the committor function satisfies

$$q(\mathbf{x}) = \arg\min_{\tilde{q}} L\left[\tilde{q}\right] \quad \text{s.t.} \quad \tilde{q}(\mathbf{x}) = 0,\ \mathbf{x} \in \partial A; \quad \tilde{q}(\mathbf{x}) = 1,\ \mathbf{x} \in \partial B\,. \tag{5.2.3}$$

Equation (5.2.3) guides a new ML-based optimization problem, where we may approximate the committor function with a neural network model $q(\mathbf{x}) \approx \hat{q}(\mathbf{x}; \boldsymbol{\theta})$ with the model parameters $\boldsymbol{\theta}$. Introducing the BKE loss function as

$$\ell(\mathbf{x}; \boldsymbol{\theta}) = \frac{1}{2} |\nabla_{\mathbf{x}} \hat{q}(\mathbf{x}; \boldsymbol{\theta})|^2 \tag{5.2.4}$$

and imposing boundary conditions in Eq. (4.2.23) by the penalty method [222] with the loss functions

$$\ell_{\mathrm{A}}(\mathbf{x}_A; \boldsymbol{\theta}) = \frac{1}{2} (\hat{q}(\mathbf{x}_A; \boldsymbol{\theta}))^2\,, \tag{5.2.5}$$

$$\ell_{\mathrm{B}}(\mathbf{x}_B; \boldsymbol{\theta}) = \frac{1}{2} (\hat{q}(\mathbf{x}_B; \boldsymbol{\theta}) - 1)^2\,, \tag{5.2.6}$$

where $\mathbf{x}_A \in A$ and $\mathbf{x}_B \in B$, the model parameters $\boldsymbol{\theta}$ can be obtained by extremizing the following objective function:

$$L(\boldsymbol{\theta}) = \langle \ell(\mathbf{x}; \boldsymbol{\theta}) \rangle + \lambda_{\mathrm{A}} \left\langle \ell_{\mathrm{A}}(\mathbf{x}; \boldsymbol{\theta}) \right\rangle_A + \lambda_{\mathrm{B}} \left\langle \ell_{\mathrm{B}}(\mathbf{x}; \boldsymbol{\theta}) \right\rangle_B\,. \tag{5.2.7}$$

Here, $\langle \ldots \rangle_C$ denotes ensemble averaging constrained in a region $C \subset \Omega$, and λ_{A} and λ_{B} control the penalty strengths that enforce boundary conditions at A and B, respectively.

Note that the ensemble average of the BKE loss function $\langle \ell(\mathbf{x};\boldsymbol{\theta})\rangle$ is proportional to the reaction rate in Eq. (4.2.35) up to a constant factor $2k_{\mathrm{B}}T/\gamma$, and thus it is crucial for any ML approach that solves the BKE to be able to compute $\langle \ell(\mathbf{x};\boldsymbol{\theta})\rangle$ accurately.

The task of minimizing Eq. (5.2.7) may not yet be feasible in large system sizes, since the ensemble averages involve high-dimensional integrals, which may be evaluated via standard quadrature but their computational cost grows exponentially with system size. To resolve this issue, one may approximate the ensemble averages in Eq. (5.2.7) with averages over samples obtained via molecular dynamics (MD) or Monte Carlo (MC) simulations. In this case, Eq. (5.2.7) can be evaluated as

$$\hat{L}(\boldsymbol{\theta};\mathcal{S},\mathcal{A},\mathcal{B}) = \frac{1}{|\mathcal{S}|}\sum_{\mathbf{x}\in\mathcal{S}}\ell(\mathbf{x};\boldsymbol{\theta}) + \frac{\lambda_{\mathrm{A}}}{|\mathcal{A}|}\sum_{\mathbf{x}\in\mathcal{A}}\ell_{\mathrm{A}}(\mathbf{x};\boldsymbol{\theta}) + \frac{\lambda_{\mathrm{B}}}{|\mathcal{B}|}\sum_{\mathbf{x}\in\mathcal{B}}\ell_{\mathrm{B}}(\mathbf{x};\boldsymbol{\theta})\,, \tag{5.2.8}$$

where $\mathcal{A}$, $\mathcal{B}$ and $\mathcal{S}$ are batches of samples obtained in the reactant state A, product state B and configuration space Ω, respectively, and the operator $|\cdot|$ denotes the size of each batch. The outlined strategy is the basis behind some of the recent ML approaches for solving the BKE [8–10, 221] though earlier works can be found that utilize a different objective function to train a neural network that takes collective variables as input and is trained on data obtained from transition path sampling [223, 224]. The main challenge inherent in these approaches is sampling; since the first term in Eq. (5.2.7) is proportional to the magnitude of the flux $|\mathbf{J}(\mathbf{x};\boldsymbol{\theta})| = \rho(\mathbf{x})\frac{k_{\mathrm{B}}T}{\gamma}|\nabla_{\mathbf{x}}\hat{q}(\mathbf{x};\boldsymbol{\theta})|$, the optimization problem is dominated by the rare configurations found in regions of high flux, e.g. the transition-state ensemble. Inadequate sampling of the transition-state ensemble may lead to poor estimates of the average BKE loss function in Eq. (5.2.4), resulting in an inaccurate computation of committor functions and reaction rates in Eq. (4.2.35). Inadequate sampling may also lead to poor estimates of the gradient $\nabla_{\boldsymbol{\theta}}L$, which may negatively impact the performance of the neural network training. In Ref. [10], this sampling problem is partially resolved via an importance sampling technique, namely umbrella sampling, that is coupled with the neural network model in a feedback loop.

5.3 Solving the BKE with Umbrella Sampling and Feedback Loops

In this section, we review the algorithm in Ref. [10] that utilizes umbrella sampling for obtaining the committor functions. To this end, consider a system that evolves via discrete overdamped Langevin dynamics with noise $\mathbf{w}_t$ that has zero mean and unit variance. As discussed in Section 3.3, umbrella sampling biases the system's dynamics by adding a potential of the form $W(\mathbf{x};\boldsymbol{\theta}) = \frac{1}{2}\kappa(\hat{q}(\mathbf{x};\boldsymbol{\theta}) - q_0)^2$ to the potential energy function $V(\mathbf{x})$, where q_0 is the target committor value and κ is the bias strength. This bias leads to modified equations of motion

$$\mathbf{x}_{t+1} = \mathbf{x}_t - \gamma^{-1}\nabla_{\mathbf{x}}\left[V(\mathbf{x}_t) + W(\mathbf{x}_t;\boldsymbol{\theta})\right]\Delta t + \sqrt{2k_{\mathrm{B}}T\Delta t\gamma^{-1}}\mathbf{w}_t\,, \tag{5.3.1}$$

which sample a target distribution given by $\rho(\mathbf{x};\boldsymbol{\theta}) \propto e^{-\beta[V(\mathbf{x})+W(\mathbf{x};\boldsymbol{\theta})]}$ as $\Delta t \to 0$. With a suitable choice of q_0 and κ, the system may explore configurations $\mathbf{x}$ and values of $\hat{q}(\mathbf{x};\boldsymbol{\theta})$ that are rare according to the unbiased equilibrium distribution $\rho(\mathbf{x}) \sim e^{-\beta V(\mathbf{x})}$. In Ref. [10], this strategy is expanded to target a range of $\hat{q}(\mathbf{x};\boldsymbol{\theta})$ values between zero and one by introducing M-many simulation systems, each of which uses a biasing potential with a unique target value and biasing strength. Referring to these simulation systems as replicas and enumerating them via an indexing variable $\alpha \in \{1,\dots,M\}$, the bias potential for each replica can be written as $W_\alpha(\mathbf{x};\boldsymbol{\theta}) = \frac{1}{2}\kappa_\alpha(\hat{q}(\mathbf{x};\boldsymbol{\theta})-q_\alpha)^2$, which induces a biased distribution $\rho_\alpha(\mathbf{x};\boldsymbol{\theta}) \propto e^{-\beta[V(\mathbf{x})+W_\alpha(\mathbf{x};\boldsymbol{\theta})]}$. The set of target committor values and biasing strengths is denoted as $\{(\kappa_\alpha, q_\alpha)\}_{\alpha=1}^{M}$. Note that the configurations corresponding to the target distributions can also be generated via MC or other MD methods instead of Eq. (5.3.1).

The algorithm for solving the BKE is a closed feedback loop between the replica dynamics and any chosen optimizer, such as stochastic gradient descent (SGD) [225], Heavy-Ball [213], or Adam [226], to obtain model parameters $\boldsymbol{\theta}$ that extremize Eq. (5.2.8). At the k-th iteration, replicas generate samples that are stored into a collection of batches $\{\mathcal{M}_k^\alpha\}_{\alpha=1}^{M}$, where the α-th batch $\mathcal{M}_k^\alpha$ consists of samples obtained from a short MD/MC trajectory run of the α-th replica. This data is then used to compute the gradient $\nabla_{\boldsymbol{\theta}}\hat{L}$ in order to update the model parameters $\boldsymbol{\theta}_k \to \boldsymbol{\theta}_{k+1}$. At the $(k+1)$-th iteration, the process repeats by using $\hat{q}(\mathbf{x};\boldsymbol{\theta}_{k+1})$ to obtain new samples for further optimization.

The algorithm requires two additional components. First, the reactant and product batches $\mathcal{A}$ and $\mathcal{B}$ are generated using short MD/MC trajectories constrained in the reactant and product states, respectively. Second, a formula for $\nabla_{\boldsymbol{\theta}}\hat{L}$ is needed for the optimizer and is obtained using a reweighting procedure [137] to compute the unbiased sample averages from biased samples. This yields

$$\begin{aligned}\nabla_{\boldsymbol{\theta}}\hat{L}\left(\boldsymbol{\theta}_k;\{(\mathcal{M}_k^\alpha, z_\alpha)\},\mathcal{A}_k,\mathcal{B}_k\right) = &\frac{\sum_{\alpha=1}^{M}\frac{z_\alpha}{|\mathcal{M}_k^\alpha|}\sum_{\mathbf{x}\in\mathcal{M}_k^\alpha}\left[\frac{\nabla_{\boldsymbol{\theta}}\ell(\mathbf{x};\boldsymbol{\theta}_k)}{c(\mathbf{x};\boldsymbol{\theta}_k)}\right]}{\sum_{\alpha=1}^{M}\frac{z_\alpha}{|\mathcal{M}_k^\alpha|}\sum_{\mathbf{x}\in\mathcal{M}_k^\alpha}\left[\frac{1}{c(\mathbf{x};\boldsymbol{\theta}_k)}\right]} + \frac{\lambda_{\mathrm{A}}}{|\mathcal{A}_k|}\sum_{\mathbf{x}\in\mathcal{A}_k}\nabla_{\boldsymbol{\theta}}\ell_{\mathrm{A}}(\mathbf{x};\boldsymbol{\theta}_k)\\ &+\frac{\lambda_{\mathrm{B}}}{|\mathcal{B}_k|}\sum_{\mathbf{x}\in\mathcal{B}_k}\nabla_{\boldsymbol{\theta}}\ell_{\mathrm{B}}(\mathbf{x};\boldsymbol{\theta}_k)\,,\end{aligned} \tag{5.3.2}$$

where $\mathcal{A}_k \subset \mathcal{A}$ and $\mathcal{B}_k \subset \mathcal{B}$ are mini-batches obtained from random sub-sampling of the reactant and product batches, respectively, and $c(\mathbf{x};\boldsymbol{\theta}) = \sum_{\alpha=1}^{M} e^{-\beta W_\alpha(\mathbf{x};\boldsymbol{\theta})}$. Here, z_α is a reweighting factor given by the relative partition function

$$z_\alpha = \frac{Q_\alpha}{\sum_{\alpha'=1}^{M} Q_{\alpha'}} = \frac{\int \mathrm{d}\mathbf{x}\ e^{-\beta[V(\mathbf{x})+W_\alpha(\mathbf{x};\boldsymbol{\theta})]}}{\sum_{\alpha'=1}^{M}\int \mathrm{d}\mathbf{x}\ e^{-\beta[V(\mathbf{x})+W_{\alpha'}(\mathbf{x};\boldsymbol{\theta})]}}\,, \tag{5.3.3}$$

where Q_α is the partition function of the α-th replica. Given the batches of samples $\{\mathcal{M}_k^\alpha\}_{\alpha=1}^{M}$, various free-energy methods [227] can be used to compute Q_α via the free-energy

Algorithm 4: The BKE–US Method [10]

```
Data: Initial conditions $\boldsymbol{\theta}_0$. Reactant and product batches $\mathcal{A}$ and $\mathcal{B}$. Hyperparameters for optimizer $\boldsymbol{\eta}$. Bias potential parameters $\{(\kappa_\alpha, q_\alpha)\}_{\alpha=1}^{M}$. Penalty strengths $\lambda_{\mathrm{A}}$ and $\lambda_{\mathrm{B}}$.
1 for $k = 0, \dots, K$ do
2     for $\alpha = 1, \dots, M$ in parallel do
3         for $m = 1, \dots, |\mathcal{M}_k^\alpha|$ do
4             Sample $\mathbf{x}_m^\alpha \sim \rho_\alpha(\mathbf{x}; \boldsymbol{\theta}_k)$ with MD/MC simulation, e.g., Eq. (5.3.1).
5             Store $\mathbf{x}_m^\alpha$ in batch $\mathcal{M}_k^\alpha$.
6     Sample mini-batch $\mathcal{A}_k \subset \mathcal{A}$ and $\mathcal{B}_k \subset \mathcal{B}$.
7     Compute $z_\alpha$ with a free-energy method, e.g., FEP Eq. (5.3.6).
8     Compute $\nabla_{\boldsymbol{\theta}} \hat{L}(\boldsymbol{\theta}_k; \{(\mathcal{M}_k^\alpha, z_\alpha)\}, \mathcal{A}_k, \mathcal{B}_k)$ with Eq. (5.3.2).
9     Update $\boldsymbol{\theta}_k \to \boldsymbol{\theta}_{k+1}$ with optimizer.
```

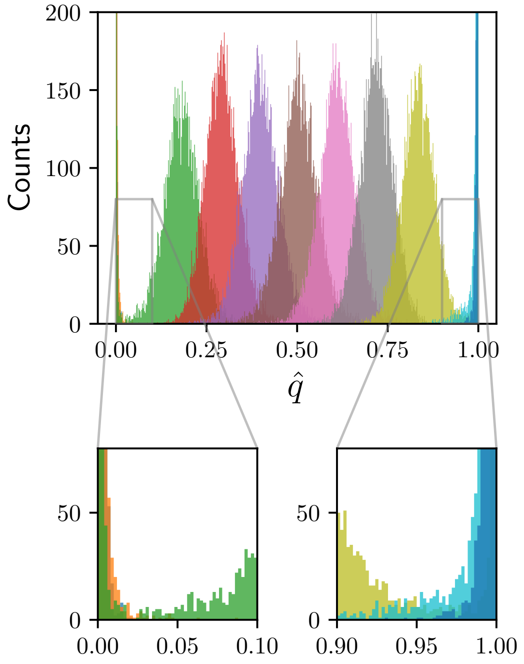


Figure 5.1: (Left) Pseudo-code corresponding to the BKE–US method. Lines 2-6 are the sampling steps, Lines 7-9 are the optimization steps, and a feedback loop couples the sampling and optimization steps together. Note that the sampling of configuration $\mathbf{x}_m^\alpha$ in Line 4 utilizes a fixed simulation length to obtain uncorrelated samples in the batch $\mathcal{M}_k^\alpha$—a convention used for all subsequent algorithms proposed in this work. (Right) Histograms of committor values from committor-based umbrella sampling. The histograms overlap near the transition state, with inset plots showing that the histograms are non-overlapping near the reactant and product states. See also Fig. 5.8(b, top) for the corresponding histograms in configuration space.

$F_\alpha = -\frac{1}{\beta} \ln Q_\alpha$. In this work, we use free-energy perturbation (FEP) [133] where the estimator for z_α is derived from the following exact identity which is a restatement of Eq. (3.3.7):

$$\frac{z_\alpha}{z_{\alpha'}} = e^{-\beta \Delta F_{\alpha,\alpha'}} = \left\langle \frac{\phi_\alpha(\mathbf{x}; \boldsymbol{\theta})}{\phi_{\alpha'}(\mathbf{x}; \boldsymbol{\theta})} \right\rangle_{\alpha'}, \tag{5.3.4}$$

where $\langle \ldots \rangle_{\alpha'}$ is an ensemble average over the distribution $\rho_{\alpha'} \propto e^{-\beta[V(\mathbf{x})+W_{\alpha'}(\mathbf{x};\boldsymbol{\theta})]}$ obeyed by the α'-th replica, $\Delta F_{\alpha,\alpha'} = F_\alpha - F_{\alpha'}$ is the relative free-energy difference, and $\phi_\alpha(\mathbf{x};\boldsymbol{\theta}) = e^{-\beta W_\alpha(\mathbf{x};\boldsymbol{\theta})}$. Given a batch $\mathcal{M}_k^{\alpha'}$ from the α'-th replica, Eq. (5.3.4) can be estimated as

$$\frac{z_\alpha}{z_{\alpha'}} \approx \frac{1}{|\mathcal{M}_k^{\alpha'}|} \sum_{\mathbf{x}\in\mathcal{M}_k^{\alpha'}} \frac{\phi_\alpha(\mathbf{x};\boldsymbol{\theta}_k)}{\phi_{\alpha'}(\mathbf{x};\boldsymbol{\theta}_k)} \,. \tag{5.3.5}$$

The accuracy of Eq. (5.3.5) quickly deteriorates if samples obtained between the α-th and α'-th replicas do not overlap [228]. To mitigate this issue, we can employ a strategy called stratification [229], where the forward and backward free-energy differences per Eq. (5.3.5) between adjacent replicas are used to compute the overall free-energy difference of replica α in reference to replica γ. This strategy yields the following formula:

$$z_\alpha = \frac{z_\alpha^\star}{\sum_{\alpha=1}^M z_\alpha^\star}; \quad z_\alpha^\star = \begin{cases} \prod_{i=\gamma}^{\alpha-1} e^{-\beta\Delta F_{(i+1),i}} \approx \prod_{i=\gamma}^{\alpha-1} \left(\frac{1}{|\mathcal{M}_k^i|} \sum_{\mathbf{x}\in\mathcal{M}_k^i} \frac{\phi_{i+1}(\mathbf{x};\boldsymbol{\theta}_k)}{\phi_i(\mathbf{x};\boldsymbol{\theta}_k)} \right) & \alpha > \gamma \\ \prod_{i=\alpha}^{\gamma-1} e^{-\beta\Delta F_{(i-1),i}} \approx \prod_{i=\alpha}^{\gamma-1} \left(\frac{1}{|\mathcal{M}_k^i|} \sum_{\mathbf{x}\in\mathcal{M}_k^i} \frac{\phi_{i-1}(\mathbf{x};\boldsymbol{\theta}_k)}{\phi_i(\mathbf{x};\boldsymbol{\theta}_k)} \right) & \alpha < \gamma \\ 1 & \alpha = \gamma \end{cases}, \tag{5.3.6}$$

where $\gamma \sim \text{unif}\{1, M\}$ is randomly chosen at every iteration. In what follows, we shall refer to this complete algorithm as the BKE–US method, whose pseudocode is described in Algorithm 4 (Fig. 5.1, left). Note that Ref. [10] recommends choosing a different set of biasing potentials such that $c(\mathbf{x};\boldsymbol{\theta}_k) \approx 1$, which corresponds to a special case of Eq. (5.3.2). Additionally, Ref. [10] uses replica exchange, where configurations are exchanged between neighboring replicas to alleviate issues with metastability, which is not used here.

The challenge in the BKE–US method lies in selecting the bias potential parameters $\{(\kappa_\alpha, q_\alpha)\}_{\alpha=1}^M$ such that the average loss functions and their gradients are accurately estimated with low variance. Since these estimates are obtained by reweighting procedures their accuracy depends severely on obtaining an accurate estimate of the free-energy differences $\Delta F_{\alpha,\alpha'}$, and hence the reweighting factors z_α. If one follows the procedures common to umbrella sampling and free-energy calculations, this is achieved by ensuring overlap in the histograms of the biased $\hat{q}(\mathbf{x};\boldsymbol{\theta})$ values [229]. One may choose as initial guess $q_\alpha = (\alpha-1)/(M-1)$ with equal biasing strengths, which is the setting recommended in Ref. [10], to obtain such overlap. However, since the committor varies rapidly near the transition state in the presence of high-energy barriers, this setting may lead to inadequate sampling of regions between the transition state and reactant/product state. This reduces the overlap between histograms, thereby reducing the accuracy as well as increasing the variance of the estimated average loss functions obtained from reweighting. Figure 5.1(right) shows such behavior in the histograms of $\hat{q}$-values, with the replicas near the edges having progressively worse overlaps than the replicas biased towards the transition state. Such a non-overlapping behavior is even more apparent in the configuration space, as shown in Fig. 5.8(b) for a one-dimensional system, where large gaps in the histograms between the reactant/product basins and the transition states can be observed. It may be plausible that further importance sampling near the edges increases the overlap, but this requires further fine-tuning of the bias parameters to focus more heavily on regions where $\hat{q}(\mathbf{x};\boldsymbol{\theta}) \approx 0$ and $\hat{q}(\mathbf{x};\boldsymbol{\theta}) \approx 1$; a non-trivial procedure to perform in high-dimensional systems. Alternatively, one may also increase the batch size to improve the chances of obtaining samples in the poorly targeted regions, but this task may require prohibitively long simulations. Altogether, these issues motivate us to construct modifications to the BKE–US method, described in the next sections.

5.4 Adding Elements of Supervised Learning

To begin with, the accuracy of the BKE–US method (Algorithm 4) can be improved by adding supervised learning elements, where one can train the neural network to fit $\hat{q}(\mathbf{x};\boldsymbol{\theta})$ to known estimates of $q(\mathbf{x})$. It has been found that supervised learning elements in the context of training neural network models achieve better performance by finding global minima in problems originally devoid of such elements [230–232]. In our case, supervised learning can be implemented by evaluating an estimate of $q(\mathbf{x})$ denoted as the empirical committor function $q_{\text{emp}}(\mathbf{x})$ using short trajectories that start from a configuration $\mathbf{x}$. The quantity $q_{\text{emp}}(\mathbf{x})$ can be obtained from a sample-mean estimator of Eq. (4.2.1):

$$q_{\text{emp}}(\mathbf{x}) = \frac{1}{H}\sum_{i=1}^{H} h_B\left(\mathbf{x}_\tau;\ \mathbf{x}_0 = \mathbf{x}\right) , \tag{5.4.1}$$

where the averaging is performed over H-many trajectories that are conditioned upon starting at $\mathbf{x}_0 = \mathbf{x}$, and ending at the first-passage time τ. This estimator obeys the binomial distribution and its variance scales as $\frac{1}{H}$ [206]. It is important to note that supervised learning of committor functions without importance sampling is ineffective since it is necessary for the neural network to be trained on empirical committor values corresponding to rare events, i.e., configurations along the transition tube including the transition state. To this end, one may use either umbrella sampling as described before or the FTS method, introduced in Section 4.3, to target the transition tube.

At this stage, an objective function must be formulated to inform $\hat{q}(\mathbf{x};\boldsymbol{\theta})$ with the empirical committor function $q_{\text{emp}}(\mathbf{x})$. To this end, a loss function in supervised learning is typically postulated as the squared error for every configuration $\mathbf{x}$:

$$\ell_{\text{MSE}}(q_{\text{emp}}, \mathbf{x};\boldsymbol{\theta}) = \frac{1}{2}(\hat{q}(\mathbf{x};\boldsymbol{\theta}) - q_{\text{emp}})^2 . \tag{5.4.2}$$

Suppose that $q_{\text{emp}}(\mathbf{x})$ is computed from configurations sampled by different replicas during importance sampling. For every α-th replica, this allows us to generate a batch of samples $\mathcal{C}^\alpha$, which is a set of pairs of empirical committor function and its corresponding configuration.

Algorithm 5: The BKE–US+SL Method

```
Data: Initial conditions $\boldsymbol{\theta}_0$. Reactant and product batches $\mathcal{A}$ and $\mathcal{B}$.
      Hyperparameters for optimizer $\boldsymbol{\eta}$. Bias potential parameters $\{(\kappa_\alpha, q_\alpha)\}_{\alpha=1}^{M}$.
      Penalty strengths $\lambda_{\mathrm{A}}$, $\lambda_{\mathrm{B}}$, and $\lambda_{\mathrm{SL}}$. Starting and ending iteration index, $k_{\mathrm{emp,s}}$
      and $k_{\mathrm{emp,e}}$, and sampling period $\tau_{\mathrm{emp}}$ for supervised learning.
1  for $k = 0, \dots, K$ do
2      for $\alpha = 1, \dots, M$ in parallel do
3          for $m = 1, \dots, |\mathcal{M}_k^\alpha|$ do
4              Sample $\mathbf{x}_m^\alpha \sim \rho_\alpha(\mathbf{x}; \boldsymbol{\theta}_k)$ with MD/MC simulation, e.g., Eq. (5.3.1).
5              Store $\mathbf{x}_m^\alpha$ in batch $\mathcal{M}_k^\alpha$.
6          if $k \geq k_{\mathrm{emp,s}}$ and $k < k_{\mathrm{emp,e}}$ and $k \pmod{\tau_{\mathrm{emp}}} = 0$ then
7              Evaluate $q_{\mathrm{emp}}$ at $\mathbf{x}^\alpha \in \mathcal{M}_k^\alpha$ with Eq. (5.4.1).
8              Store $(q_{\mathrm{emp}}, \mathbf{x}^\alpha)$ in batch $\mathcal{C}^\alpha$.
9      Sample mini-batch $\mathcal{A}_k \subset \mathcal{A}$, $\mathcal{B}_k \subset \mathcal{B}$, and $\mathcal{C}_k^\alpha \subset \mathcal{C}^\alpha$.
10     Compute $z_\alpha$ with a free-energy method, e.g., FEP Eq. (5.3.6).
11     Compute $\nabla_{\boldsymbol{\theta}} \hat{L}(\boldsymbol{\theta}_k; \{(\mathcal{M}_k^\alpha, z_\alpha)\}, \mathcal{A}_k, \mathcal{B}_k) + \nabla_{\boldsymbol{\theta}} \hat{L}_{\mathrm{SL}}(\boldsymbol{\theta}_k; \{\mathcal{C}_k^\alpha\})$ with Eqs. (5.3.2)
        and (5.4.5).
12     Update $\boldsymbol{\theta}_k \to \boldsymbol{\theta}_{k+1}$ with optimizer.
```

Figure 5.2: Pseudo-code for the BKE–US+SL method.

Denoting the collection of batches as $\{\mathcal{C}^\alpha\}_{\alpha=1}^{M}$, and given Eq. (5.4.2), the objective function as a mean-squared error has the form

$$\hat{L}_{\mathrm{MSE}}(\boldsymbol{\theta}; \{\mathcal{C}^\alpha\}) = \frac{\lambda_{\mathrm{MSE}}}{M} \sum_{\alpha=1}^{M} \frac{1}{|\mathcal{C}^\alpha|} \sum_{(q_{\mathrm{emp}}, \mathbf{x}) \in \mathcal{C}^\alpha} \ell_{\mathrm{MSE}}(q_{\mathrm{emp}}, \mathbf{x}; \boldsymbol{\theta}), \tag{5.4.3}$$

where λ_{MSE} is the penalty strength. In practice, an optimizer to train the neural network requires the gradient $\nabla_{\boldsymbol{\theta}} \hat{L}_{\mathrm{MSE}}$ as additional input, which can be computed using a collection of mini-batches $\{\mathcal{C}_k^\alpha\}$ with $\mathcal{C}_k^\alpha \subset \mathcal{C}^\alpha$ generated via random sub-sampling of the original batch $\mathcal{C}^\alpha$ similar to the sub-sampling procedure in Eq. (5.3.2).

Note that a finite number of trajectories are used to obtain estimates of committor values for each configuration $\mathbf{x}$, resulting in a statistically noisy variation of $q_{\mathrm{emp}}(\mathbf{x})$. Therefore, using the objective function Eq. (5.4.3) to train the neural network may lead to overfitting issues and loss in accuracy. To alleviate this problem, we introduce a modified form of the objective function where we first evaluate the squared mean error for a batch of samples $\mathcal{C}^\alpha$ corresponding to the α-th replica:

$$\ell_{\mathrm{ME}}(\mathcal{C}^\alpha; \boldsymbol{\theta}) = \frac{1}{2} \left[\frac{1}{|\mathcal{C}^\alpha|} \sum_{(q_{\mathrm{emp}}, \mathbf{x}) \in \mathcal{C}^\alpha} (\hat{q}(\mathbf{x}; \boldsymbol{\theta}) - q_{\mathrm{emp}}) \right]^2 . \tag{5.4.4}$$

This is then reduced across all replicas, yielding the modified supervised learning objective function

$$\hat{L}_{\mathrm{SL}}(\boldsymbol{\theta};\{\mathcal{C}^{\alpha}\}) = \frac{\lambda_{\mathrm{SL}}}{M}\sum_{\alpha=1}^{M} \ell_{\mathrm{ME}}(\mathcal{C}^{\alpha};\boldsymbol{\theta})\,, \tag{5.4.5}$$

where λ_{SL} is the penalty strength. Equation (5.4.5) indicates the neural network is trained on committor errors that are locally averaged over a single replica. Such an averaging smears out the statistical error in $q_{\mathrm{emp}}(\mathbf{x})$, alleviates the issue of overfitting, and further helps the neural network generalize to regions outside of the ones covered by sampling. A more detailed discussion, which shows results comparing the standard (Eq. (5.4.3)) and modified (Eq. (5.4.5)) objective functions for a two-dimensional system can be found in Appendix B.2.3

To incorporate the supervised learning strategy in the BKE–US method, each replica computes $q_{\mathrm{emp}}(\mathbf{x}^{\alpha})$ between the sampling and optimization steps of the algorithm, where $\mathbf{x}^{\alpha}$ is the current configuration of replica α. The committor evaluation can be initiated at a chosen iteration $k = k_{\mathrm{emp,s}}$ until $k = k_{\mathrm{emp,e}}$, after which no more $q_{\mathrm{emp}}(\mathbf{x}^{\alpha})$ values are computed. Since each $q_{\mathrm{emp}}(\mathbf{x}^{\alpha})$ requires the initiation of H-many trajectories starting at $\mathbf{x}_0 = \mathbf{x}^{\alpha}$, the committor is evaluated infrequently every τ_{emp} iterations to reduce the computational cost. The pseudocode combining supervised learning with the BKE–US method is described in Algorithm 4 (Fig. 5.2), and is herein referred to as the BKE–US+SL method.

5.5 Solving the BKE with the Finite-Temperature String Method

For methods employing umbrella sampling, it is important to ensure sufficient overlap in samples obtained from neighboring replicas, since the overlap guarantees accurate computation of reweighting factors z_{α}, and further controls the accuracy in the estimator for the average loss functions, e.g., the average BKE loss function, which sets the reaction rate. As mentioned before, this may require exhaustive fine-tuning of the algorithm parameters, or long simulations to obtain a larger number of samples. On the other hand, the framework of TPT already provides the finite-temperature string (FTS) method [204, 205], which can homogeneously sample overlapping regions across the transition tube with few control parameters. The FTS method also yields the transition path $\boldsymbol{\varphi}(s)$ without needing to compute the committor function $q(\mathbf{x})$. Therefore, if we replace the committor-based umbrella sampling with the FTS method, we eliminate the feedback loop between importance sampling and the neural network training in learning $q(\mathbf{x})$. Furthermore, it is also possible to obtain a low-variance estimate of the reaction rate due to the overlaps in samples obtained from the FTS method.

With the FTS method described in Section 4.3, we now proceed to construct new algorithms for minimizing the loss in Eq. (5.2.8). The key idea behind all subsequent new algorithms is to replace the committor-based umbrella sampling in the BKE–US method with the FTS method. This allows the replicas to generate samples that homogeneously cover the transition tube with little fine-tuning, and enables accurate low-variance estimation of the average BKE loss function and its gradient. As mentioned before, since the average BKE loss function is proportional to the chemical reaction rate, the FTS method also enables accurate estimation of reaction rates during optimization of the neural network.

5.5.1 The FTS Method with Master Equation

The first algorithm that we construct involves running the FTS method simultaneously with the neural network training. In particular, the replicas from the FTS method generate batches of sampled configurations $\{\mathcal{R}_k^\alpha\}_{\alpha=1}^M$ to update the current path $\{\boldsymbol{\varphi}_k^\alpha\}$ via Eqs. (4.3.18)–(4.3.19), as well as the neural network parameters $\boldsymbol{\theta}_k$ by computing the gradient of the loss in Eq. (5.2.8). Note that, in this algorithm, there is no feedback loop between the neural network and updates to the path. In this case, the loss gradient $\nabla_{\boldsymbol{\theta}} \hat{L}$ can be calculated using modified versions of Eqs. (5.3.2)–(5.3.3), where the bias potentials W_α are replaced with hard-wall potentials constraining each replica to its Voronoi cell, i.e.,

$$W_\alpha(\mathbf{x}; \{\boldsymbol{\varphi}^\alpha\}) = \begin{cases} 0 & \mathbf{x} \in R_\alpha \\ \infty & \text{otherwise} \end{cases} . \tag{5.5.1}$$

This yields

$$\nabla_{\boldsymbol{\theta}} \hat{L}\left(\boldsymbol{\theta}_k; \{(\mathcal{R}_k^\alpha, z_\alpha)\}, \mathcal{A}_k, \mathcal{B}_k\right) = \sum_{\alpha=1}^{M} \frac{z_\alpha}{|\mathcal{R}_k^\alpha|} \sum_{\mathbf{x} \in \mathcal{R}_k^\alpha} \nabla_{\boldsymbol{\theta}} \ell(\mathbf{x}; \boldsymbol{\theta}_k) + \frac{\lambda_{\mathrm{A}}}{|\mathcal{A}_k|} \sum_{\mathbf{x} \in \mathcal{A}_k} \nabla_{\boldsymbol{\theta}} \ell_{\mathrm{A}}(\mathbf{x}; \boldsymbol{\theta}_k) + \frac{\lambda_{\mathrm{B}}}{|\mathcal{B}_k|} \sum_{\mathbf{x} \in \mathcal{B}_k} \nabla_{\boldsymbol{\theta}} \ell_{\mathrm{B}}(\mathbf{x}; \boldsymbol{\theta}_k)\,, \tag{5.5.2}$$

where the reweighting factors z_α are

$$z_\alpha = \frac{\int_{R_\alpha} \mathrm{d}\mathbf{x}\; e^{-\beta V(\mathbf{x})}}{\int_{\bigcup_{\alpha=1}^M R_\alpha} \mathrm{d}\mathbf{x}\; e^{-\beta V(\mathbf{x})}} = \frac{\int_{R_\alpha} \mathrm{d}\mathbf{x}\; e^{-\beta V(\mathbf{x})}}{\int_{\Omega} \mathrm{d}\mathbf{x}\; e^{-\beta V(\mathbf{x})}} = \int_{R_\alpha} \mathrm{d}\mathbf{x}\; \rho(\mathbf{x})\,. \tag{5.5.3}$$

Equation (5.5.3) indicates z_α is the equilibrium probability of finding $\mathbf{x}$ to be in a Voronoi cell R_α. This set of equilibrium probabilities can be computed as a solution to a steady-state master equation, whose form is found by identifying the instantaneous rates (or fluxes) between neighboring Voronoi cells [205]. To this end, let $N_{\alpha\alpha'}$ be the number of times that the α-th replica attempts to exit its Voronoi cell R_α and enter a neighboring Voronoi cell $R_{\alpha'}$, e.g., the number of times that $\mathbf{x}_\star^\alpha \in R_{\alpha'}$ for the replica dynamics given by Eqs. (4.3.13)–(4.3.14). Let $k_{\alpha\alpha'}$ be the rate at which the system transitions between R_α to $R_{\alpha'}$. Denoting N_{steps}^α as the total simulation length of the α-th replica, the previous rate can be evaluated as $k_{\alpha\alpha'} \approx N_{\alpha\alpha'}/N_{\text{steps}}^\alpha$. The steady-state master equation is then given by a balance between the total rate of leaving and entering the Voronoi cell R_α:

$$\sum_{\alpha'=1}^{M} z_{\alpha'} k_{\alpha'\alpha} = \sum_{\alpha'=1}^{M} z_\alpha k_{\alpha\alpha'}, \quad \forall \alpha \in \{1, \ldots, M\}\,, \tag{5.5.4}$$

which can be solved to obtain z_α; see Appendix B.1 for more details, and also Section III of Ref. [233] for a more detailed discussion of Eq. (5.5.4). Equations (5.5.2) and (5.5.4) constitute the new algorithm, and will herein be referred to as the BKE–FTS(ME) method, whose pseudocode is described in Algorithm 6.

We note that higher dimensional problems will require collective variables (see Section 4.5), In this situation, the new BKE–FTS algorithms can still use the FTS method with the CV as given in Eq. (4.5.1) to train neural networks that are implicitly aware of the solvent degrees of freedom that the CV does not account for, since each replica samples the solvent configurations that participate in the transition. Such a strategy of utilizing the FTS method with the CV in Eq. (4.5.1) is used in Section 5.7 to compute committor functions and reaction rates in a solvated dimer system with relatively high accuracy.

Algorithm 6: The BKE–FTS(ME) Method

```
Data: Initial conditions $\boldsymbol{\theta}_0$, $\{\boldsymbol{\varphi}_0^\alpha\}$. Reactant and product batches $\mathcal{A}$ and $\mathcal{B}$.
      Hyperparameters for optimizers $\boldsymbol{\eta}$. The FTS Method step size $\Delta\tau$ and penalty
      strength $\lambda_{\mathrm{S}}$. Penalty strengths $\lambda_{\mathrm{A}}$ and $\lambda_{\mathrm{B}}$.
1  for $k = 0, \dots, K$ do
2      for $\alpha = 1, \dots, M$ in parallel do
3          for $m = 1, \dots, |\mathcal{R}_k^\alpha|$ do
4              Sample $\mathbf{x}_m^\alpha$ with MD/MC simulation constrained in the Voronoi cell
                 $R_\alpha(\{\boldsymbol{\varphi}_k^\alpha\})$, e.g., Eqs. (4.3.13) and (4.3.14).
5              Store $\mathbf{x}_m^\alpha$ in batch $\mathcal{R}_k^\alpha$.
6      $\boldsymbol{\varphi}_\alpha^{k+1} \leftarrow$ Eqs. (4.3.18) and (4.3.19).
7      Sample mini-batch $\mathcal{A}_k \subset \mathcal{A}$ and $\mathcal{B}_k \subset \mathcal{B}$.
8      Compute $z_\alpha$ by solving the master equation Eq. (5.5.4).
9      Compute $\nabla_{\boldsymbol{\theta}} \hat{L}(\boldsymbol{\theta}_k; \{(\mathcal{R}_k^\alpha, z_\alpha)\}, \mathcal{A}_k, \mathcal{B}_k)$ with Eq. (5.5.2).
10     Update $\boldsymbol{\theta}_k \to \boldsymbol{\theta}_{k+1}$ with optimizer.
```

Figure 5.3: (Left) An illustration of the FTS method, where each replica samples configurations inside a Voronoi cell. (Right) Pseudo-code for the BKE–FTS(ME) method. Note that the path is updated concurrently with the neural network at the k-th iteration.

5.5.2 The FTS Method with Umbrella Sampling

As mentioned before, given a sufficient number of nodes, the BKE–FTS(ME) method guarantees homogeneous sampling across the transition path (see also Fig. 5.8(b)), which better ensures low-variance estimation from reweighting. Accuracy can also be improved by running longer simulations, i.e., larger N_{steps}^α, since they lead to more accurate estimates of the rates $k_{\alpha\alpha'}$, thereby reducing the error in the estimated reweighting factor z_α. Despite this,

Algorithm 7: The BKE–FTS(US) Method

```
Data: Initial conditions $\boldsymbol{\theta}_0$. Nodal points of the transition path $\{\boldsymbol{\varphi}^\alpha\}$ obtained from
      the FTS method. Reactant and product batches $\mathcal{A}$ and $\mathcal{B}$. Hyperparameters
      for optimizers $\boldsymbol{\eta}$. Penalty strengths $\lambda_\mathrm{A}$ and $\lambda_\mathrm{B}$.
1 for $k = 0, \dots, K$ do
2     for $\alpha = 1, \dots, M$ in parallel do
3         for $m = 1, \dots, |\mathcal{M}_k^\alpha|$ do
4             Sample $\mathbf{x}_m^\alpha \sim \rho_\alpha(\mathbf{x}; \{\boldsymbol{\varphi}^\alpha\})$ with MD/MC simulation, e.g., Eq. (5.3.1) and
               Eq. (5.5.5).
5             Store $\mathbf{x}_m^\alpha$ in batch $\mathcal{M}_k^\alpha$.
6     Sample mini-batch $\mathcal{A}_k \subset \mathcal{A}$ and $\mathcal{B}_k \subset \mathcal{B}$.
7     Compute $z_\alpha$ with a free-energy method, e.g., FEP Eq. (5.3.6).
8     Compute $\nabla_{\boldsymbol{\theta}} \hat{L}(\boldsymbol{\theta}_k; \{(\mathcal{M}_k^\alpha, z_\alpha)\}, \mathcal{A}_k, \mathcal{B}_k)$ with Eq. (5.3.2).
9     Update $\boldsymbol{\theta}_k \to \boldsymbol{\theta}_{k+1}$ with optimizer.
```

Figure 5.4: Pseudo-code for the BKE–FTS(US) method.

the error in z_α is difficult to study as it involves the error propagation of $k_{\alpha\alpha'}$, which forms a random matrix in the master equation. On the other hand, z_α computed from umbrella sampling is amenable to error analysis [137, 234], which makes it feasible to determine the error in the estimates computed from reweighting as a function of batch size. This motivates us to construct a modification to the BKE–FTS(ME) method where the computation of z_α is based on umbrella sampling and FEP (Eq. (5.3.6)). The modified algorithm consists of running the FTS method before the neural network training to obtain the transition path $\{\boldsymbol{\varphi}^\alpha\}_{\alpha=1}^M$, which is then used as a basis for umbrella sampling across the transition tube to subsequently train the neural network.

The path-based umbrella sampling requires new bias potentials that can lead to better overlaps between adjacent replicas, as well as sufficient exploration of regions transverse to the path. The latter is necessary to ensure the neural network representing the committor function is also accurate in regions away from the transition path. To this end, we construct new bias potentials such that different bias strengths can be specified in directions parallel and transverse to the path. Let $\mathbf{t}^\alpha$ be the unit tangent vector at node $\boldsymbol{\varphi}^\alpha$, evaluated using finite differences. We then form the projection matrices $\mathbf{P}_\alpha^\parallel = \mathbf{t}^\alpha \otimes \mathbf{t}^\alpha$ and $\mathbf{P}_\alpha^\perp = \mathbf{I} - \mathbf{t}^\alpha \otimes \mathbf{t}^\alpha$ to decompose a vector into a component that is parallel and transverse to $\mathbf{t}^\alpha$, respectively. The bias potential for the α-th replica can be written as

$$W_\alpha(\mathbf{x}; \{\boldsymbol{\varphi}^\alpha\}) = \frac{1}{2}\kappa_\alpha^\parallel(\mathbf{x} - \boldsymbol{\varphi}^\alpha)\mathbf{P}_\alpha^\parallel(\mathbf{x} - \boldsymbol{\varphi}^\alpha) + \frac{1}{2}\kappa_\alpha^\perp(\mathbf{x} - \boldsymbol{\varphi}^\alpha)\mathbf{P}_\alpha^\perp(\mathbf{x} - \boldsymbol{\varphi}^\alpha)\,, \tag{5.5.5}$$

where $\kappa_\alpha^\parallel$ and $\kappa_\alpha^\perp$ are the bias strengths for the parallel and transverse direction, respectively.

Algorithm 8: The BKE–FTS(ME)+SL Method

```
Data: Initial conditions $\boldsymbol{\theta}_0$, $\{\boldsymbol{\varphi}_0^\alpha\}$. Reactant and product batches $\mathcal{A}$ and $\mathcal{B}$.
      Hyperparameters for optimizers $\boldsymbol{\eta}$. The FTS Method step size $\Delta\tau$ and penalty
      strength $\lambda_{\mathrm{S}}$. Penalty strengths $\lambda_{\mathrm{A}}$, $\lambda_{\mathrm{B}}$, and $\lambda_{\mathrm{SL}}$. Starting and ending iteration
      index, $k_{\mathrm{emp,s}}$ and $k_{\mathrm{emp,e}}$, and sampling period $\tau_{\mathrm{emp}}$ for supervised learning.
1  for $k = 0, \dots, K$ do
2      for $\alpha = 1, \dots, M$ in parallel do
3          for $m = 1, \dots, |\mathcal{R}_k^\alpha|$ do
4              Sample $\mathbf{x}_m^\alpha$ with MD/MC simulation constrained in the Voronoi cell
                $R_\alpha(\{\boldsymbol{\varphi}_k^\alpha\})$, e.g., Eqs. (4.3.13)–(4.3.14).
5              Store $\mathbf{x}_m^\alpha$ in batch $\mathcal{R}_k^\alpha$.
6          if $k \geq k_{\mathrm{emp,s}}$ and $k < k_{\mathrm{emp,e}}$ and $k \pmod{\tau_{\mathrm{emp}}} = 0$ then
7              Evaluate $q_{\mathrm{emp}}$ at $\mathbf{x}^\alpha \in \mathcal{R}_k^\alpha$ with Eq. (5.4.1).
8              Store $(q_{\mathrm{emp}}, \mathbf{x}^\alpha)$ in batch $\mathcal{C}^\alpha$.
9      $\boldsymbol{\varphi}_\alpha^{k+1} \leftarrow$ Eqs. (4.3.18)–(4.3.19).
10     Sample mini-batch $\mathcal{A}_k \subset \mathcal{A}$, $\mathcal{B}_k \subset \mathcal{B}$, and $\mathcal{C}_k^\alpha \subset \mathcal{C}^\alpha$.
11     Compute $z_\alpha$ by solving the master equation Eq. (5.5.4).
12     Compute $\nabla_{\boldsymbol{\theta}} \hat{L}(\boldsymbol{\theta}_k; \{(\mathcal{R}_k^\alpha, z_\alpha)\}, \mathcal{A}_k, \mathcal{B}_k) + \nabla_{\boldsymbol{\theta}} \hat{L}_{\mathrm{SL}}(\boldsymbol{\theta}_k; \{\mathcal{C}_k^\alpha\})$ with Eqs. (5.4.5)
        and (5.5.2).
13     Update $\boldsymbol{\theta}_k \rightarrow \boldsymbol{\theta}_{k+1}$ with optimizer.
```

Figure 5.5: Pseudo-code for the BKE–FTS(ME)+SL method.

To promote exploration of regions transverse to the path, the bias strengths are set such that $\kappa_\alpha^\perp < \kappa_\alpha^\parallel$. For sufficiently strong bias, this results in every replica exploring an oblate ellipsoidal region, where the center of the ellipsoid is located at node $\boldsymbol{\varphi}^\alpha$, and its axis of rotation is parallel to the tangent vector $\mathbf{t}^\alpha$. Note that a similar bias potential has also been used in Ref. [235] but defined with respect to a low-dimensional collective-variable space.

The loss gradient $\nabla_{\boldsymbol{\theta}} \hat{L}$ needed for this algorithm can be computed with Eq. (5.3.2) and Eq. (5.3.6) from the BKE–US method, using samples obtained from biased MD/MC simulations. As in the BKE–FTS(ME) method, there exists no feedback loop between the neural network and umbrella sampling because the bias potentials are based on the transition path, which remains static during training. This modification to the BKE–FTS(ME) method shall be referred to as the BKE–FTS(US) method, whose pseudocode is described in Algorithm 7 (Fig. 5.4). The algorithm shares similar advantages as the BKE–FTS(ME) method, since homogeneous sampling across the transition tube and overlap in configuration space is readily achieved for large enough bias strengths. Unlike the master-equation approach, the bias and variance in the reweighting factors z_α estimated from FEP are amenable to error analysis [234]. As shown later in Section 5.6.3, we provide an error analysis of the

```
Algorithm 9: The BKE–FTS(US)+SL Method
Data: Initial conditions $\boldsymbol{\theta}_0$. Nodal points of the transition path $\{\boldsymbol{\varphi}^\alpha\}$ obtained from
      the FTS method. Reactant and product batches $\mathcal{A}$ and $\mathcal{B}$. Hyperparameters
      for optimizers $\boldsymbol{\eta}$. Penalty strengths $\lambda_\mathrm{A}$, $\lambda_\mathrm{B}$, and $\lambda_\mathrm{SL}$. Starting and ending
      iteration index, $k_\mathrm{emp,s}$ and $k_\mathrm{emp,e}$, and sampling period $\tau_\mathrm{emp}$ for supervised
      learning.
1  for $k = 0, \dots, K$ do
2      for $\alpha = 1, \dots, M$ in parallel do
3          for $m = 1, \dots, |\mathcal{M}_k^\alpha|$ do
4              Sample $\mathbf{x}_m^\alpha \sim \rho_\alpha(\mathbf{x}; \{\boldsymbol{\varphi}^\alpha\})$ with MD/MC simulation, e.g., Eqs. (5.3.1)
                and (5.5.5).
5              Store $\mathbf{x}_m^\alpha$ in batch $\mathcal{M}_k^\alpha$.
6          if $k \geq k_\mathrm{emp,s}$ and $k < k_\mathrm{emp,e}$ and $k \pmod{\tau_\mathrm{emp}} = 0$ then
7              Evaluate $q_\mathrm{emp}$ at $\mathbf{x}^\alpha \in \mathcal{M}_k^\alpha$ with Eq. (5.4.1).
8              Store $(q_\mathrm{emp}, \mathbf{x}^\alpha)$ in batch $\mathcal{C}^\alpha$.
9      Sample mini-batch $\mathcal{A}_k \subset \mathcal{A}$, $\mathcal{B}_k \subset \mathcal{B}$, and $\mathcal{C}_k^\alpha \subset \mathcal{C}^\alpha$.
10     Compute $z_\alpha$ with a free-energy method, e.g., FEP Eq. (5.3.6).
11     Compute $\nabla_{\boldsymbol{\theta}} \hat{L}(\boldsymbol{\theta}_k; \{(\mathcal{M}_k^\alpha, z_\alpha)\}, \mathcal{A}_k, \mathcal{B}_k) + \nabla_{\boldsymbol{\theta}} \hat{L}_\mathrm{SL}(\boldsymbol{\theta}_k; \{\mathcal{C}_k^\alpha\})$ with Eqs. (5.3.2)
        and (5.4.5).
12     Update $\boldsymbol{\theta}_k \to \boldsymbol{\theta}_{k+1}$ with optimizer.
```

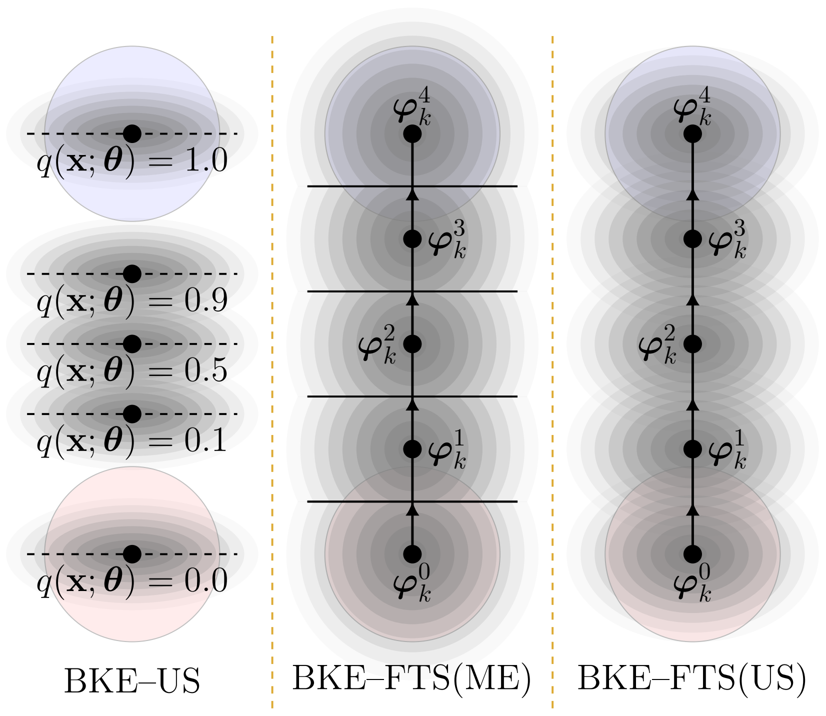
Figure 5.6: Pseudo-code for the BKE–FTS(US)+SL method.

estimated average loss function, and a procedure where the bias in the estimator can be removed, thereby enabling accurate estimation of reaction rates with smaller batch sizes.

5.5.3 The FTS Method with Supervised Learning

Both the BKE–FTS(ME) and BKE–FTS(US) methods can be combined with the supervised learning methodology developed in Section 5.4 to further improve the accuracy of the committor function. Since the samples obtained by either method homogeneously cover the transition tube, they provide access to configurations that can be used for computing empirical committor function $q_\mathrm{emp}(\mathbf{x})$ necessary for supervised learning. The empirical committor function $q_\mathrm{emp}(\mathbf{x})$ may be evaluated by the replicas between the sampling and optimization step of the algorithms. Similar to the procedure described in Section 5.4, it can be evaluated at a rate τ_emp between a starting iteration $k_\mathrm{emp,s}$ and an ending iteration $k_\mathrm{emp,e}$. Given these estimates, the supervised-learning loss in Eq. (5.4.5) can be used to compute the compound loss gradient to update the neural network. We shall call these composite algorithms as the BKE–FTS(ME)+SL and BKE–FTS(US)+SL method, whose pseudo-codes are described in Algorithm 8 (Fig. 5.5) and Algorithm 9 (Fig. 5.6), respectively.

5.6 Computational Studies in Low-Dimensional Systems



$q(\mathbf{x};\boldsymbol{\theta}) = 1.0$
$q(\mathbf{x};\boldsymbol{\theta}) = 0.9$
$q(\mathbf{x};\boldsymbol{\theta}) = 0.5$
$q(\mathbf{x};\boldsymbol{\theta}) = 0.1$
$q(\mathbf{x};\boldsymbol{\theta}) = 0.0$
BKE–US
$\boldsymbol{\varphi}_k^4$
$\boldsymbol{\varphi}_k^3$
$\boldsymbol{\varphi}_k^2$
$\boldsymbol{\varphi}_k^1$
$\boldsymbol{\varphi}_k^0$
BKE–FTS(ME)
$\boldsymbol{\varphi}_k^4$
$\boldsymbol{\varphi}_k^3$
$\boldsymbol{\varphi}_k^2$
$\boldsymbol{\varphi}_k^1$
$\boldsymbol{\varphi}_k^0$
BKE–FTS(US)

Figure 5.7: Summary of the methods presented for solving the BKE using a neural network. (Left) BKE–US, in which umbrella sampling with respect to $\hat{q}(\mathbf{x};\boldsymbol{\theta})$ is performed. (Middle) BKE–FTS(ME), in which the typical FTS method is used to sample the reaction pathway where hard-walls are implemented to constrain replicas to their Voronoi cell. (Right) BKE–FTS(US), in which umbrella sampling is performed with respect to the Voronoi cell centers.

In this section, we test Algorithms 4–9 (see Fig. 5.7 for a visual summary) to two model systems consisting of a single particle diffusing in non-convex potential energies in one dimension (1D) and two dimensions (2D), respectively. Reference solutions can be obtained in 1D and 2D via analytical method and the finite element method (FEM), respectively, which will be used to ascertain the relative accuracy of the algorithms. Before we introduce these two systems, we elaborate on the choice of the neural network, optimizer, and initial conditions. For both systems, we use a single-hidden layer neural network with ReLU activation functions and a sigmoidal output layer [232]:

$$\hat{q}(\mathbf{x};\boldsymbol{\theta}=\{\mathbf{W}_1,\mathbf{w}_2,\mathbf{b}\}) = \sigma\left(\mathbf{w}_2\cdot \mathrm{ReLU}(\mathbf{W}_1\mathbf{x}+\mathbf{b}_1)\right)\,, \tag{5.6.1}$$

where $\mathrm{ReLU}(s)=\max(0,s)$, $\sigma(s)=\frac{1}{1+e^{-s}}$, $\mathbf{W}_1$ is an m-by-d matrix of weights of the hidden layer, $\mathbf{w}_2$ and $\mathbf{b}_1$ are m-dimensional vectors of weights of the output layer and biases of the hidden layer, respectively, and the number of neurons is $m=200$. The chosen optimizer is the Heavy-Ball method [213] and Adam [226] for the 1D and 2D system, respectively; see Appendix B.2 for a brief review of each optimizer and associated hyperparameters for each study.

The neural network parameters are initialized randomly and subsequently updated by minimizing the following mean-squared error function

$$I(\boldsymbol{\theta};\{\mathbf{x}_0^\alpha\}) = \frac{1}{M}\sum_{\alpha=1}^{M}\left(\hat{q}(\mathbf{x}_0^\alpha;\boldsymbol{\theta})-\frac{\alpha-1}{M-1}\right)^2\,, \tag{5.6.2}$$

where a gradient descent algorithm is used with a stepsize of 0.001 until $I(\boldsymbol{\theta})\le 10^{-3}$. Here, $\mathbf{x}_0^\alpha$ is the initial configuration of the α-th replica, and is chosen to be the linear interpolation between a known energy-minimizing configuration at the reactant state $\mathbf{x}_0^{\mathrm{A}}$ and product state $\mathbf{x}_0^{\mathrm{B}}$:

$$\mathbf{x}_0^\alpha = \left(1-\frac{\alpha-1}{M-1}\right)\mathbf{x}_0^{\mathrm{A}} + \left(\frac{\alpha-1}{M-1}\right)\mathbf{x}_0^{\mathrm{B}}\,. \tag{5.6.3}$$

For the BKE–FTS(ME) and BKE–FTS(ME)+SL methods, the initial nodal points of the path are chosen as $\boldsymbol{\varphi}_0^\alpha=\mathbf{x}_0^\alpha$. For the BKE–FTS(US) and the BKE–FTS(US)+SL method, since the FTS method is run before the neural network training, $\mathbf{x}_0^\alpha$ is set to the nodal point $\boldsymbol{\varphi}^\alpha$ of the converged path. The choice in Eq. (5.6.2) ensures an initial guess of $\boldsymbol{\theta}$ that results in a monotonic increase of the committor function from the reactant to the product states. It also provides an initial value of the committor function that is compatible with the target value of the committor-based umbrella sampling, avoiding large force evaluations for MD simulations. Additional details pertaining to individual studies such as sampling schemes generating mini-batches for optimization, choices of penalty strengths, and parameters controlling the FTS method can be found in the Appendix B.2.

The accuracy of the algorithms is measured using both an L_1 norm measuring error in $\hat{q}(\mathbf{x};\boldsymbol{\theta})$, and the ensemble average of the BKE loss function given by Eq. (5.2.4). The latter is proportional to the reaction rate in Eq. (4.2.35). The L_1-norm error is defined over the region spanned by the transition tube, $T_\Lambda=\{\mathbf{x}\in\Omega : |\mathbf{J}(\mathbf{x})|\ge\Lambda\}$ where Λ is a cut-off value, and normalized by the volume of the region. This yields

$$\|\hat{q}-q\|_1 = \frac{1}{\int_{T_\Lambda}\mathrm{d}\mathbf{x}}\int_{T_\Lambda}\mathrm{d}\mathbf{x}\,|\hat{q}(\mathbf{x};\boldsymbol{\theta})-q(\mathbf{x})|\,. \tag{5.6.4}$$

In all algorithms, an on-the-fly estimate of the ensemble average of Eq. (5.2.4) is computed at the k-th iteration with the following formula:

$$\left\langle \frac{1}{2}|\nabla_{\mathbf{x}}\hat{q}(\mathbf{x};\boldsymbol{\theta}_{\mathrm{k}})|^2 \right\rangle_{\text{fly}} = \begin{cases} \dfrac{\sum_{\alpha=1}^{M} \dfrac{z_\alpha}{|\mathcal{M}_k^\alpha|} \sum_{\mathbf{x}\in\mathcal{M}_k^\alpha} \left[\dfrac{\ell(\mathbf{x};\boldsymbol{\theta}_k)}{c(\mathbf{x};\boldsymbol{\theta}_k)}\right]}{\sum_{\alpha=1}^{M} \dfrac{z_\alpha}{|\mathcal{M}_k^\alpha|} \sum_{\mathbf{x}\in\mathcal{M}_k^\alpha} \left[\dfrac{1}{c(\mathbf{x};\boldsymbol{\theta}_k)}\right]} & \text{for umbrella sampling} \\ \sum_{\alpha=1}^{M} \dfrac{z_\alpha}{|\mathcal{R}_k^\alpha|} \sum_{\mathbf{x}\in\mathcal{R}_k^\alpha} \ell(\mathbf{x};\boldsymbol{\theta}_k) & \text{for the master equation} \end{cases}, \tag{5.6.5}$$

where the reweighting factors z_α are evaluated using Eq. (5.3.6) for umbrella sampling and Eq. (5.5.4) for the master equation, respectively. The estimate in Eq. (5.6.5) is then compared to the average BKE loss function that is evaluated using reference solutions.

5.6.1 First Study: 1D Quartic Potential

In this section we study a 1D particle diffusing in a quartic potential $V(\mathbf{x}) = (1-\mathbf{x}^2)^2$ with $k_{\mathrm{B}}T = 1/15$. This potential has two minima at $\mathbf{x} = -1, 1$ with a saddle point at $\mathbf{x} = 0$, which is the transition state of the model. Setting the reactant state $A = (-\infty, -1]$ and product state $B = [1, \infty+)$, the exact solution for the committor function $q_{\text{exact}}(\mathbf{x})$ can be obtained as

$$q_{\text{exact}}(\mathbf{x}) = \frac{\int_{-1}^{\mathbf{x}} \mathrm{d}\mathbf{x}' e^{15V(\mathbf{x}')}}{\int_{-1}^{1} \mathrm{d}\mathbf{x}' e^{15V(\mathbf{x}')}}\,. \tag{5.6.6}$$

Using Eq. (5.6.6), the average of the BKE loss function $\left\langle \frac{1}{2}|\nabla_{\mathbf{x}}q_{\text{exact}}(\mathbf{x})|^2 \right\rangle$ can be computed as

$$\left\langle \frac{1}{2}|\nabla_{\mathbf{x}}q_{\text{exact}}(\mathbf{x})|^2 \right\rangle = \frac{1}{2}\left(Q\left(\int_{-1}^{1} \mathrm{d}\mathbf{x}\; e^{\beta V(\mathbf{x})}\right)\right)^{-1} \approx 10^{-6}\,. \tag{5.6.7}$$

To compute the L_1-norm error, we set the transition tube region $T_\Lambda = \Omega \setminus A \cup B = (-1, 1)$.

Figure 5.8(a) shows that the neural network approximations $\hat{q}(\mathbf{x};\boldsymbol{\theta})$ obtained from all methods converge to the exact solution. However, the histograms of sampled configurations obtained from committor-based umbrella sampling lack overlap between the reactant/product states and the transition state (Fig. 5.8(b), top). As discussed in Section 5.3, this lack of overlap indicates that on-the-fly estimates of the average BKE loss, and thus the chemical reaction rates, may not be accurate and are subject to large variance/noise. On the other hand, the histograms from algorithms that use the FTS method (Fig. 5.8(b), middle and bottom) show homogeneous sampling across the transition tube with sufficient overlaps, which should translate to accurate low-variance estimates of reaction rates. Indeed, Fig. 5.8(c) shows that the on-the-fly estimates from the BKE–US and BKE–US+SL methods exhibit large

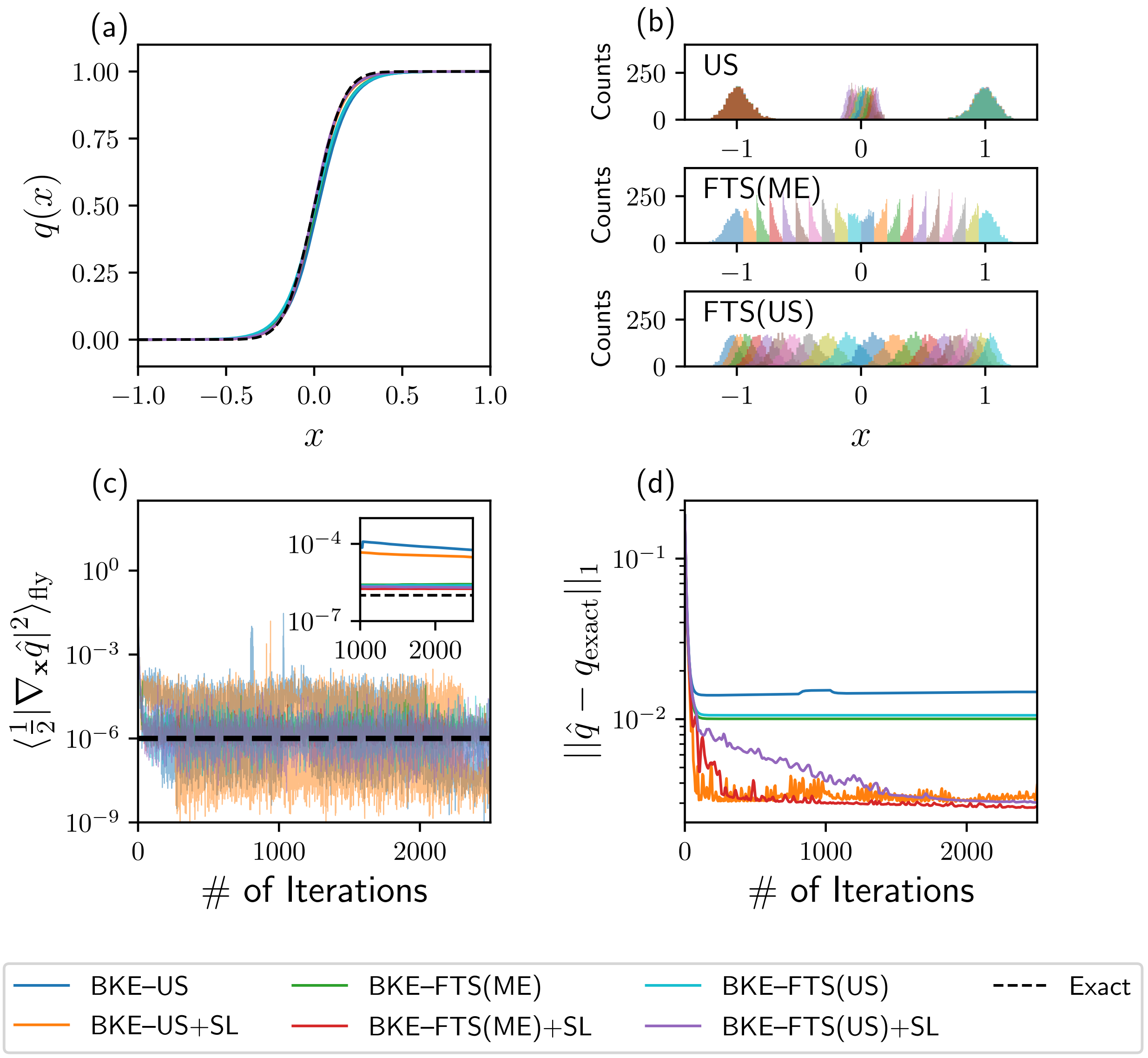


Figure 5.8: (a) Committor function obtained from all methods compared with the exact solution. (b) Histograms of samples obtained from the BKE–US method (top), the BKE–FTS(ME) method (middle), and the BKE–FTS(US) method (bottom). (c) On-the-fly estimates of the average BKE loss obtained at every iteration and computed using a batch size of 16, with an inset plot showing their cumulative averages over the last 1500 iterations. (d) The L_1-norm error as a function of iterations.

fluctuations, spanning six orders in magnitude for a batch size of 16, while the algorithms that use the FTS method can reduce this variance by approximately one order of magnitude for the same batch size. When these on-the-fly estimates are cumulatively averaged, as shown in the inset of Fig. 5.8(c), we also see that the BKE–US and BKE–US+SL methods yield inaccurate estimates of the average BKE loss when compared to the algorithms employing the FTS method, as these estimates are off from the exact value by two orders of magnitude. Irrespective of the sampling method, the addition of supervised learning elements can yield an order-of-magnitude increase in the accuracy of the committor function, as seen from the L_1-norm error in Fig. 5.8(d). Based on these results, we may conclude that the addition of the FTS method and SL elements yields accurate committor functions and low-variance estimates of the reaction rates.

5.6.2 Second Study: 2D Müller-Brown Potential

Although the 1D system already showcases the salient advantages of incorporating SL elements and the FTS method, it only serves as a check to ensure that all algorithms can converge in a setting where an exact solution is available. The advantages and disadvantages of all algorithms can be observed with a more complex problem involving a 2D potential energy landscape, where the transition path is curved. To this end, we now study a particle subject to the 2D Müller-Brown (MB) potential given by Eq. (4.4.1). Using the committor function obtained using the finite element method, $q_{\text{FEM}}(\mathbf{x})$, we will derive metrics that will test both the accuracy of both the sampling procedures and the obtained neural network. The ensemble-averaged BKE loss with $q_{\text{FEM}}(\mathbf{x})$ over Ω is obtained by evaluating the variational objective function in Eq. (5.2.2):

$$\left\langle \frac{1}{2} |\nabla_{\mathbf{x}} q_{\text{FEM}}|^2 \right\rangle \approx 2.46 \cdot 10^{-4} \,. \tag{5.6.8}$$

To compute the L_1-norm error, we select the transition tube domain to be $T_\Lambda = \{\mathbf{x} \in \Omega : |\mathbf{J}(\mathbf{x})| > \Lambda = 1.61 \cdot 10^{-4}\}$, which corresponds to the outermost white line in Fig. 4.5(b). In addition to on-the-fly estimates, the ensemble average of the BKE loss from the neural network representation $\hat{q}(\mathbf{x};\boldsymbol{\theta})$ can be evaluated by numerically integrating over the entire domain, and is given by

$$\left\langle \frac{1}{2} |\nabla_{\mathbf{x}} \hat{q}(\mathbf{x};\boldsymbol{\theta}_{\text{k}})|^2 \right\rangle_{\text{full}} = \int_{\Omega} \mathrm{d}\mathbf{x} \rho(\mathbf{x}) \ell(\mathbf{x};\boldsymbol{\theta}_{\text{k}}) = \langle \ell(\mathbf{x};\boldsymbol{\theta}_{\text{k}}) \rangle \,. \tag{5.6.9}$$

Equation (5.6.9) provides an additional metric for evaluating accuracy; in particular, comparing Eq. (5.6.9) with the on-the-fly estimates allows us to evaluate the sampling error that arises from the choice of estimator, while comparing Eq. (5.6.9) with the FEM value (Eq. (5.6.8)) allows us to evaluate the error inherent to the neural network.

Figures 5.9(a-c) show the isocommittor lines and sampled configurations obtained from all algorithms. We see from the isocommittor lines that methods employing supervised learning

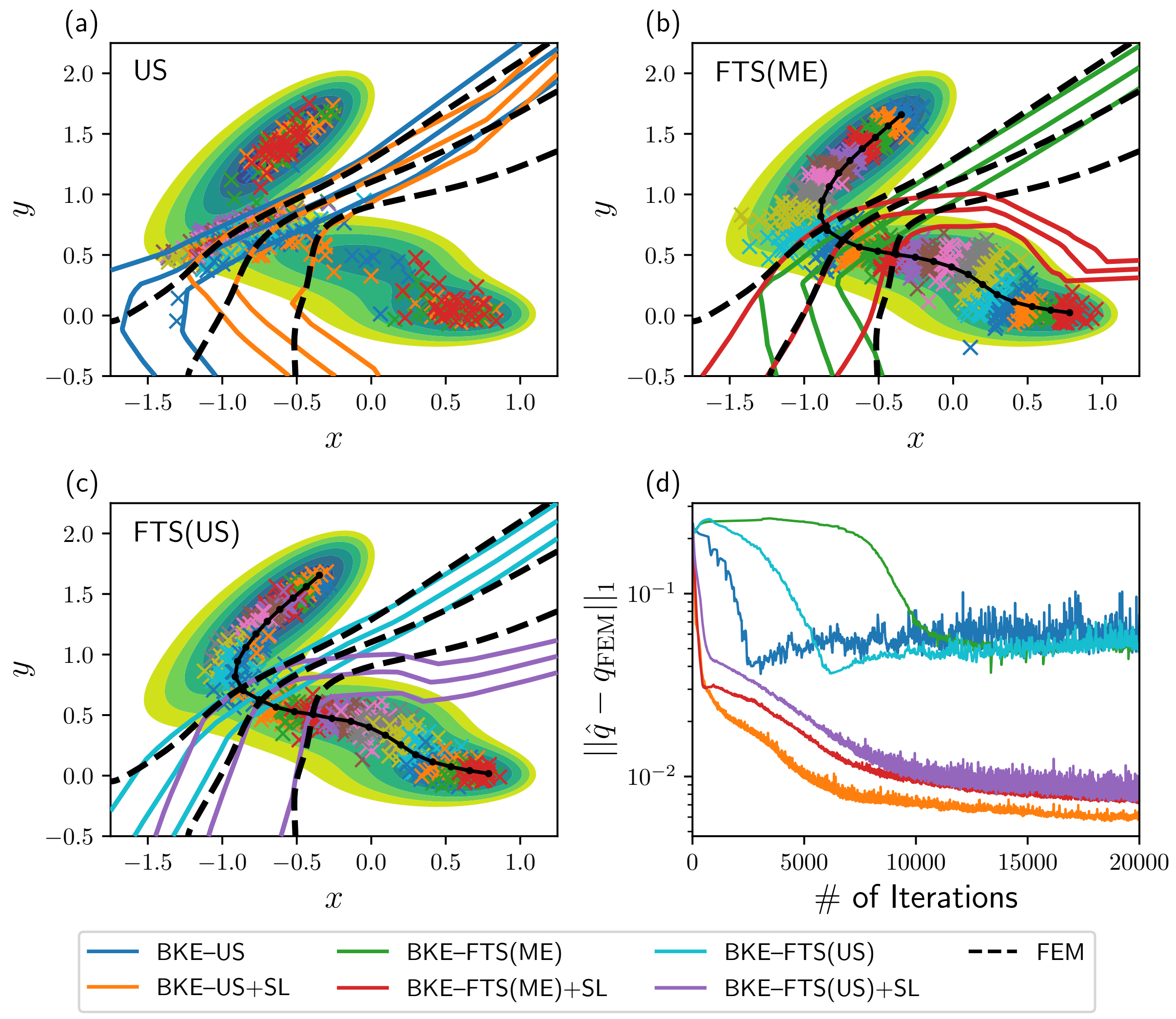


Figure 5.9: Isocommittor lines for $q = 0.1$, 0.5, and 0.9 (left to right) from (a) the BKE–US and BKE–US+SL method, (b) the BKE–FTS(ME) and BKE–FTS(ME)+SL method, and (c) the BKE–FTS(US) and BKE–FTS(US)+SL method. $\times$ markers denote representative samples obtained from algorithms without supervised learning. Dotted lines are the transition paths obtained from the FTS method. (d) The L_1-norm error of the committor function as a function of iterations.

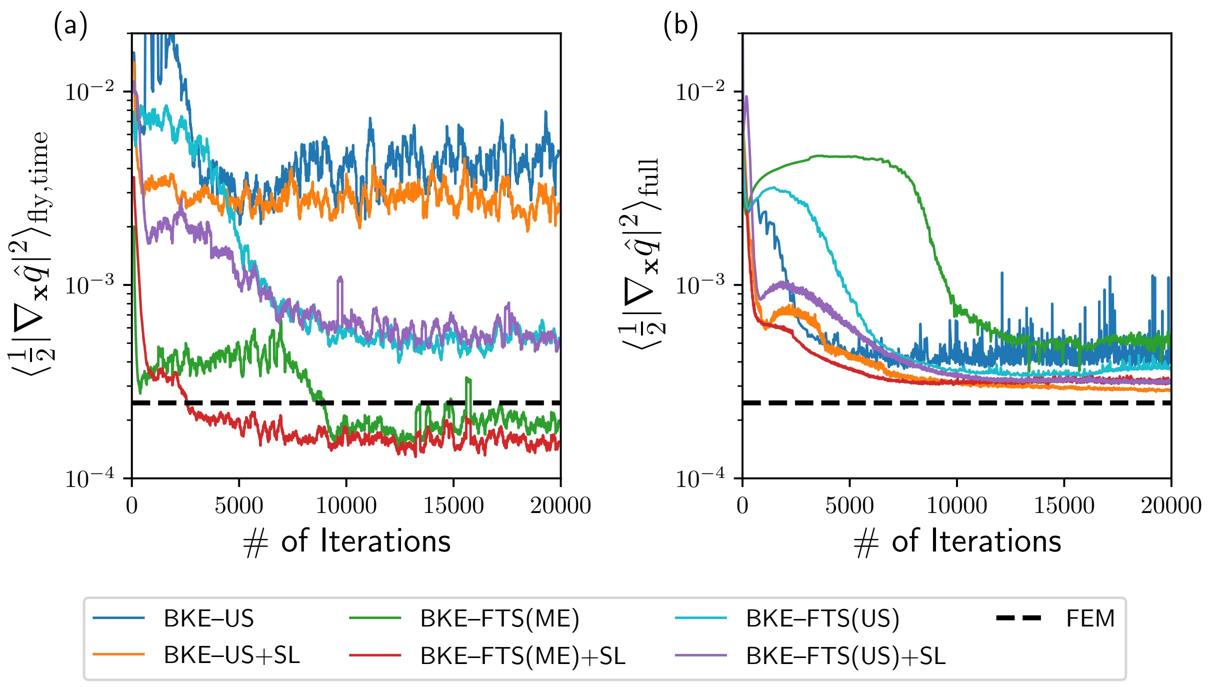


Figure 5.10: (a) The filtered on-the-fly estimate of the BKE loss obtained at every iteration, with the filtering window set to 200 iterations. (b) The ensemble-averaged loss per Eq. (5.6.9) obtained at every iteration.

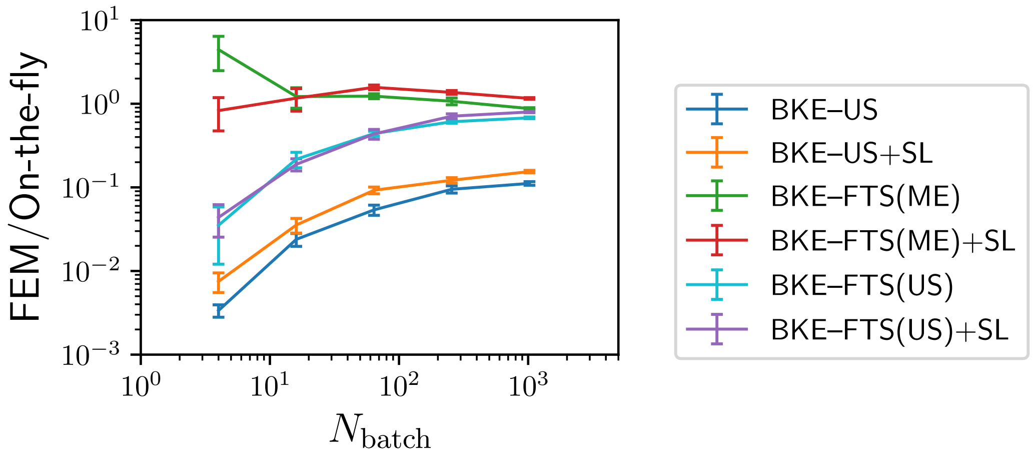


Figure 5.11: The ratio between the average BKE loss from FEM solution (Eq. (5.6.8)) and on-the-fly estimates, where the latter is cumulatively averaged over the last 3000 iterations of the neural network training.

elements improve the accuracy of the committor functions both in and outside the transition tube, as these surfaces follow the FEM solution far more closely than the ones without such elements. This increase in accuracy is also reflected in the L_1-norm error shown in Fig. 5.9(d), where the error from methods with supervised learning is reduced by an order of magnitude regardless of the chosen sampling method. Furthermore, similar to the 1D system, committor-based umbrella sampling yields samples that are focused near the transition state with little overlap between the reactant/product basins and the transition state region; see Fig. 5.9(a). As mentioned in Section 5.3, this lack of overlap can negatively impact the accuracy of the estimated reaction rates due to inaccurate estimates of free energy differences between neighboring replicas and thereby the reweighting factors (Fig. B.3). Conversely, all algorithms using the FTS method yield overlapping samples that homogeneously cover the transition tube and hence accurate estimates of reweighting factors (Figs. B.4 and B.5), indicating that reaction rate estimates may be computed with higher accuracy and lower variance.

Figure 5.10(a) shows the on-the-fly estimates of the reaction rates or the average BKE loss from all methods, computed using a smaller batch size of 64 samples and filtered over the nearest 200 iterations. With the exception of the BKE–FTS(ME) and BKE–FTS(ME)+SL methods, these on-the-fly estimates converge towards values far from the FEM solution even though the ensemble-averaged BKE loss computed by numerical integration (Eq. (5.6.9)) shows convergence towards the FEM value (Fig. 5.10(c)). This shows the sampling error is still large, and larger batch sizes (N_{batch}) are needed to obtain accurate on-the-fly estimates. Figure 5.11(a) shows the ratio of the FEM and the on-the-fly estimates as a function of batch size, where all the methods employing the FTS methods converge towards the FEM value with the exception of the BKE–US and BKE–US+SL methods, which plateau to a ratio of 0.1. As mentioned in Section 5.3, this discrepancy is related to the lack of overlaps in the samples between the transition state and the reactant/product basins, resulting in the inaccurate estimates of z_α (Fig. B.3). These results show that replacing the committor-based umbrella sampling with the FTS method results in more accurate estimates of the reaction rates.

Furthermore, the FTS method with path-based umbrella sampling is amenable to error analysis, allowing us to estimate the errors in the reaction rates. In what follows, we provide such an analysis for the BKE–FTS(US) and BKE–FTS(US)+SL methods, using which the sampling errors in the on-the-fly estimates can be eliminated. As will be shown later in Fig. 5.19, this allows accurate computation of the average BKE loss functions for the BKE–FTS(US) and BKE–FTS(US)+SL methods at any batch size. Lastly, although the average BKE loss computed by numerical integration may be closer to the FEM solution than the on-the-fly estimates, such computation is impractical for high-dimensional problems due to the increased cost of quadrature, necessitating the procedure constructed from error analysis to improve the accuracy in the on-the-fly estimates.

5.6.3 Error Analysis of the Average BKE Loss Estimator

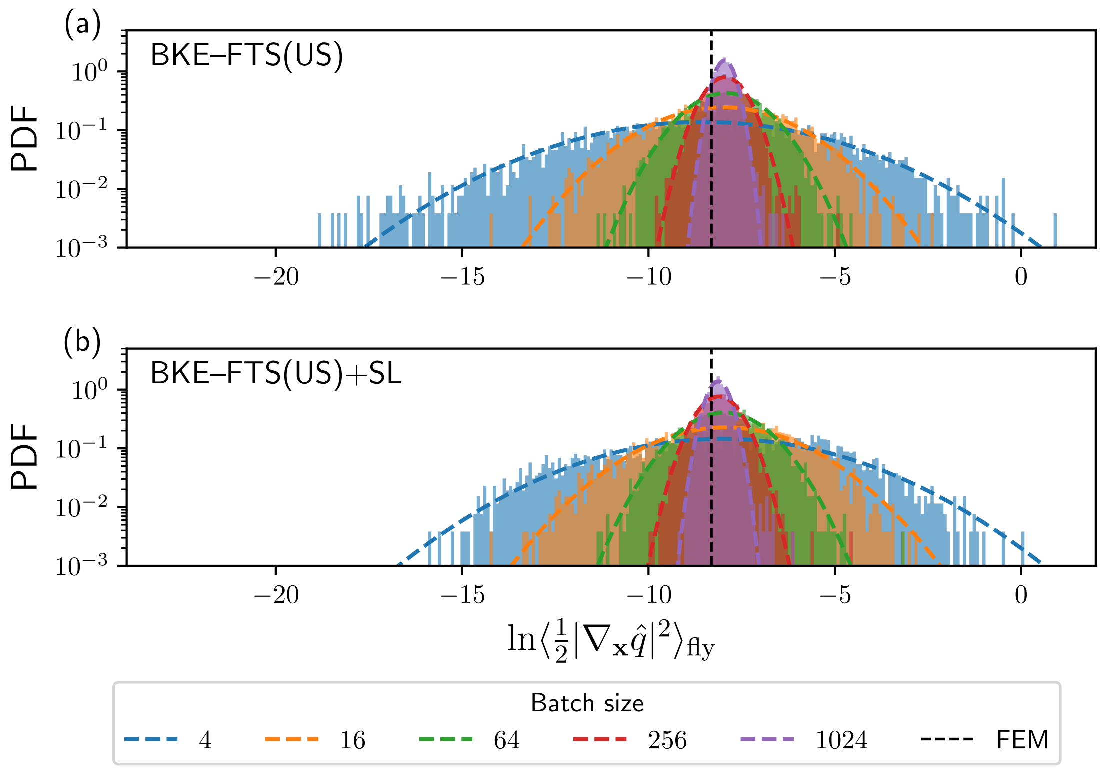


Figure 5.12: Histograms yielding the probability density functions for the on-the-fly estimate of the BKE loss at various batch sizes from the last 3000 iterations of training for the (a) BKE–FTS(US), and (b) BKE–FTS(US)+SL methods. Corresponding dashed lines are log-normal distributions fitted using the method of moments [236], while the dashed vertical black line corresponds to the average BKE loss from the FEM solution.

Before we begin the error analysis, we first plot the normalized histograms, i.e., the empirical probability density functions (PDFs), of the logarithm of on-the-fly BKE loss for both the BKE–FTS(US) and BKE–FTS(US)+SL methods (Fig. 5.12), which show that fluctuations of these estimates are centered around the FEM value. Furthermore, the resulting PDFs can be fitted to a log-normal distribution via the method of moments [236] with increasing agreement as the batch size is increased. The emergence of the log-normal distribution can be attributed to either the change in model parameters $\boldsymbol{\theta}_k$ during optimization or the nature of umbrella sampling when used in conjunction with the estimator given by Eq. (5.6.5). Since the log-normal statistics emerge when the neural network is already converged, it is more likely for sampling to be the chief cause of these statistics, rather than the optimization. This hypothesis can be tested by computing the on-the-fly BKE loss when the neural network parameters are fixed at every iteration, which has the effect of

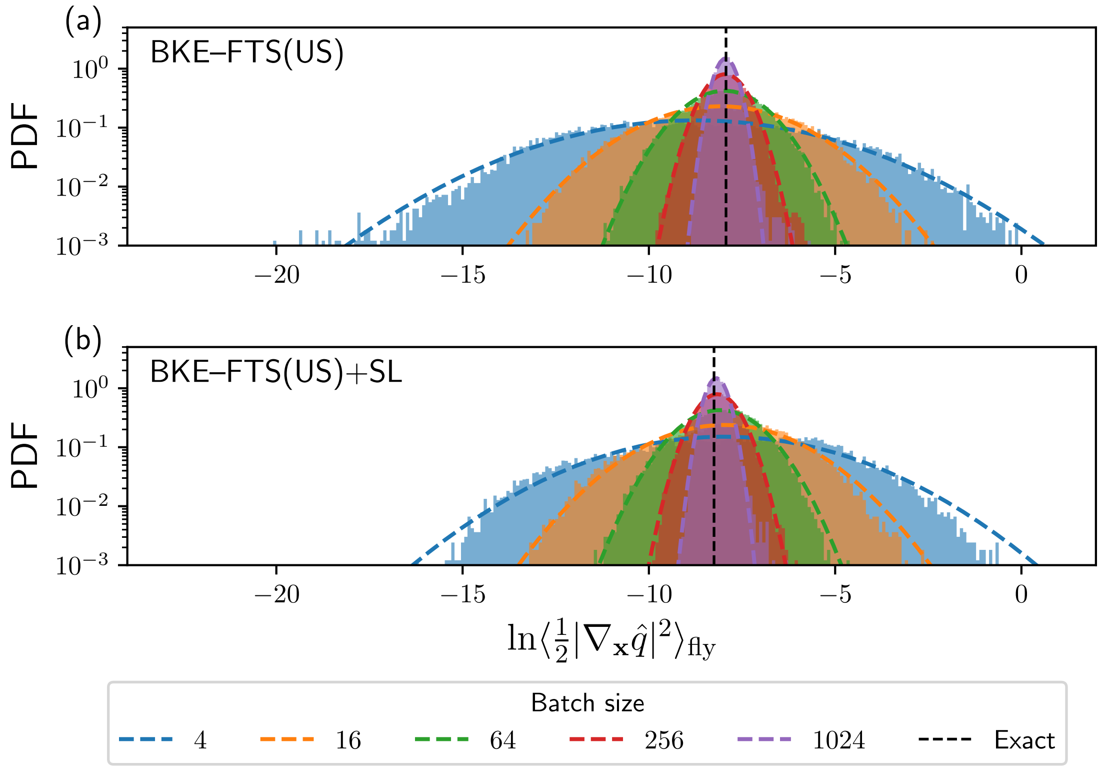


Figure 5.13: Histograms of the on-the-fly estimate of the average BKE loss at various batch sizes, with the neural network parameters fixed at every iteration for the (a) BKE–FTS(US), and (b) BKE–FTS(US)+SL methods. The neural network configuration corresponds to the one obtained from training at batch size 4. Corresponding dashed lines are log-normal distributions fitted using the method of moments [236], while the dashed vertical black line corresponds to the average BKE loss computed by numerical integration (Eq. (5.6.9)).

decoupling the influence of optimization from sampling. The histograms from this numerical experiment are shown in Fig. 5.13, where log-normal distributions are produced as before, and their peaks are located precisely at the ensemble-averaged BKE loss computed by numerical integration (Eq. (5.6.9)). The logarithm of the average BKE loss can be shifted by the mean and normalized by the standard deviation of the corresponding distributions to produce approximate standard normal distributions as seen in Fig. 5.14, with increasing batch sizes having an increasing agreement with a standard normal distribution.

With the observation of log-normal statistics established, we now determine its origin by investigating each component that contributes to the computation of the on-the-fly BKE loss in Eq. (5.6.5). To this end, we provide a more concise notation for the estimator (Eq. (5.6.5))

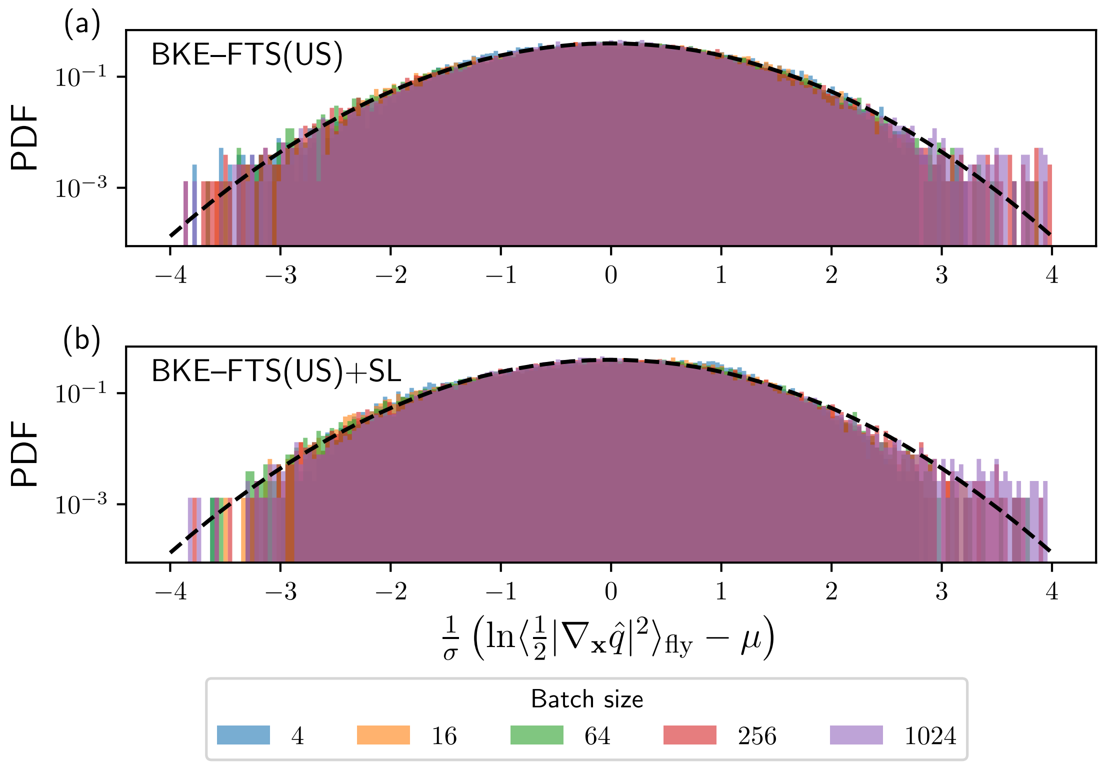


Figure 5.14: Histograms of the on-the-fly estimate of the average BKE loss shifted by the mean μ and normalized by the standard deviation σ at various batch sizes, with the neural network parameters fixed at every iteration for the (a) BKE–FTS(US), and (b) BKE–FTS(US)+SL methods. The neural network configuration corresponds to the one obtained from training with a batch size of 4. The black dashed line is a log-normal distributions with $\mu = 0$ and $\sigma = 1$.

by re-writing it as

$$\left\langle \frac{1}{2}|\nabla_{\mathbf{x}}\hat{q}(\mathbf{x};\boldsymbol{\theta}_{\mathrm{k}})|^2 \right\rangle_{\text{fly}} = \frac{\sum_{\alpha=1}^{M} z_\alpha \left(\frac{1}{|\mathcal{M}_k^\alpha|} \sum_{\mathbf{x}\in\mathcal{M}_k^\alpha} \left[\frac{\ell(\mathbf{x};\boldsymbol{\theta}_k)}{c(\mathbf{x};\boldsymbol{\theta}_k)} \right] \right)}{\sum_{\alpha=1}^{M} z_\alpha \left(\frac{1}{|\mathcal{M}_k^\alpha|} \sum_{\mathbf{x}\in\mathcal{M}_k^\alpha} \left[\frac{1}{c(\mathbf{x};\boldsymbol{\theta}_k)} \right] \right)} = \frac{\sum_{\alpha=1}^{M} z_\alpha \bar{\ell}_\alpha^*}{\sum_{\alpha=1}^{M} z_\alpha \bar{1}_\alpha^*}, \tag{5.6.10}$$

where we define the division by $c(\mathbf{x};\boldsymbol{\theta}_k)$ per sample with the $*$ operator, and denote the standard sample mean using the bar operator. Equation (5.6.10) requires computing free energies through z_α, and sample means from each replica through $\bar{\ell}_\alpha^*$ and $\bar{1}_\alpha^*$, which indicates that the origin of the log-normal statistics of the average BKE loss can be found once the statistics for z_α, $\bar{\ell}^*_\alpha$, and $\bar{1}^*_\alpha$ are determined individually. In what follows, we first investigate the statistics of z_α as computed via FEP.

To begin, we write the free-energy difference $\Delta F_{\alpha,\alpha'} = F_\alpha - F_{\alpha'}$ per Eq. (5.3.5) as

$$\beta \Delta F_{\alpha,\alpha'} = -\log \left[\frac{1}{|\mathcal{M}_k^{\alpha'}|} \sum_{\mathbf{x} \in \mathcal{M}_k^{\alpha'}} \exp(-\beta \Delta W_{\alpha,\alpha'}(\mathbf{x}; \boldsymbol{\theta}_k)) \right] , \tag{5.6.11}$$

where $\Delta W_{\alpha,\alpha'}(\mathbf{x}; \boldsymbol{\theta}_k) = W_\alpha(\mathbf{x}; \boldsymbol{\theta}_k) - W_{\alpha'}(\mathbf{x}; \boldsymbol{\theta}_k)$. Note that free-energy differences are typically computed for adjacent replicas, so that $\alpha = \alpha' \pm 1$. For sufficiently small $\Delta W_{\alpha,\alpha'}(\mathbf{x}; \boldsymbol{\theta}_k)$, use of Taylor series expansions yields

$$\beta \Delta F_{\alpha,\alpha'} \approx -\log \left[\frac{1}{|\mathcal{M}_k^{\alpha'}|} \sum_{\mathbf{x} \in \mathcal{M}_k^{\alpha'}} (1 - \beta \Delta W_{\alpha,\alpha'}(\mathbf{x}; \boldsymbol{\theta}_k)) \right] \tag{5.6.12}$$

$$\approx -\log \left[1 - \frac{1}{|\mathcal{M}_k^{\alpha'}|} \sum_{\mathbf{x} \in \mathcal{M}_k^{\alpha'}} \beta \Delta W_{\alpha,\alpha'}(\mathbf{x}; \boldsymbol{\theta}_k) \right] \tag{5.6.13}$$

$$\approx \frac{1}{|\mathcal{M}_k^{\alpha'}|} \sum_{\mathbf{x} \in \mathcal{M}_k^{\alpha'}} \beta \Delta W_{\alpha,\alpha'}(\mathbf{x}; \boldsymbol{\theta}_k) \, . \tag{5.6.14}$$

According to the central limit theorem and assuming that the samples $\mathbf{x} \in \mathcal{M}_k^\alpha$ are independent and identically distributed, the sample mean of $\Delta W_{\alpha,\alpha'}$ is normally distributed, and thus the free-energy differences $\Delta F_{\alpha,\alpha'}$ are also normally distributed. This argument only holds for small $\Delta W_{\alpha,\alpha'}(\mathbf{x}; \boldsymbol{\theta}_k)$, which can be achieved when there is overlap in configuration space—a condition that is ensured with a good choice of the bias strength parameters. Since $\Delta F_{\alpha,\alpha'}$ is normally distributed, its exponentiation $e^{-\beta \Delta F_{\alpha,\alpha'}}$ is log-normally distributed. Using Eq. (5.3.6), for α not equal to the reference index γ, the un-normalized reweighting factor $z^\star_\alpha$ obtained from FEP is also log-normally distributed, since it is computed from products of $e^{-\beta \Delta F_{\alpha,\alpha'}}$ factors that are log-normally distributed [237]. Upon normalizing $z^\star_\alpha$ to obtain z_α, we should observe approximately log-normal statistics for z_α, since the normalization requires dividing $z^\star_\alpha$ with its sum, which is approximately log-normal [238–242].

The arguments we put forth for the statistics of $\beta \Delta F_{\alpha,\alpha'}$ and z_α can be verified in simulations by evaluating the probability density functions for the quantities of interest. For the forward free-energy differences $\beta \Delta F_{(\alpha+1),\alpha}$ and the backward free-energy differences $\beta \Delta F_{(\alpha-1),\alpha}$, the observed distributions can be described by normal distributions (Figs. B.6 and B.7), which immediately imply that their exponentiation is log-normally distributed. The resulting reweighting factors z_α are found to be log-normally distributed, in agreement with our heuristic arguments, as seen from the PDFs of $\ln z_\alpha$ in the first row of Fig. 5.15 for representative replicas, and Fig. B.8 for all replicas. Note that there exist free-energy differences, such as $\beta \Delta F_{9,8}$ and $\beta \Delta F_{10,9}$, that have a slight deviation in the tails due to the presence of higher-moment terms. These effects are mostly removed when evaluating the

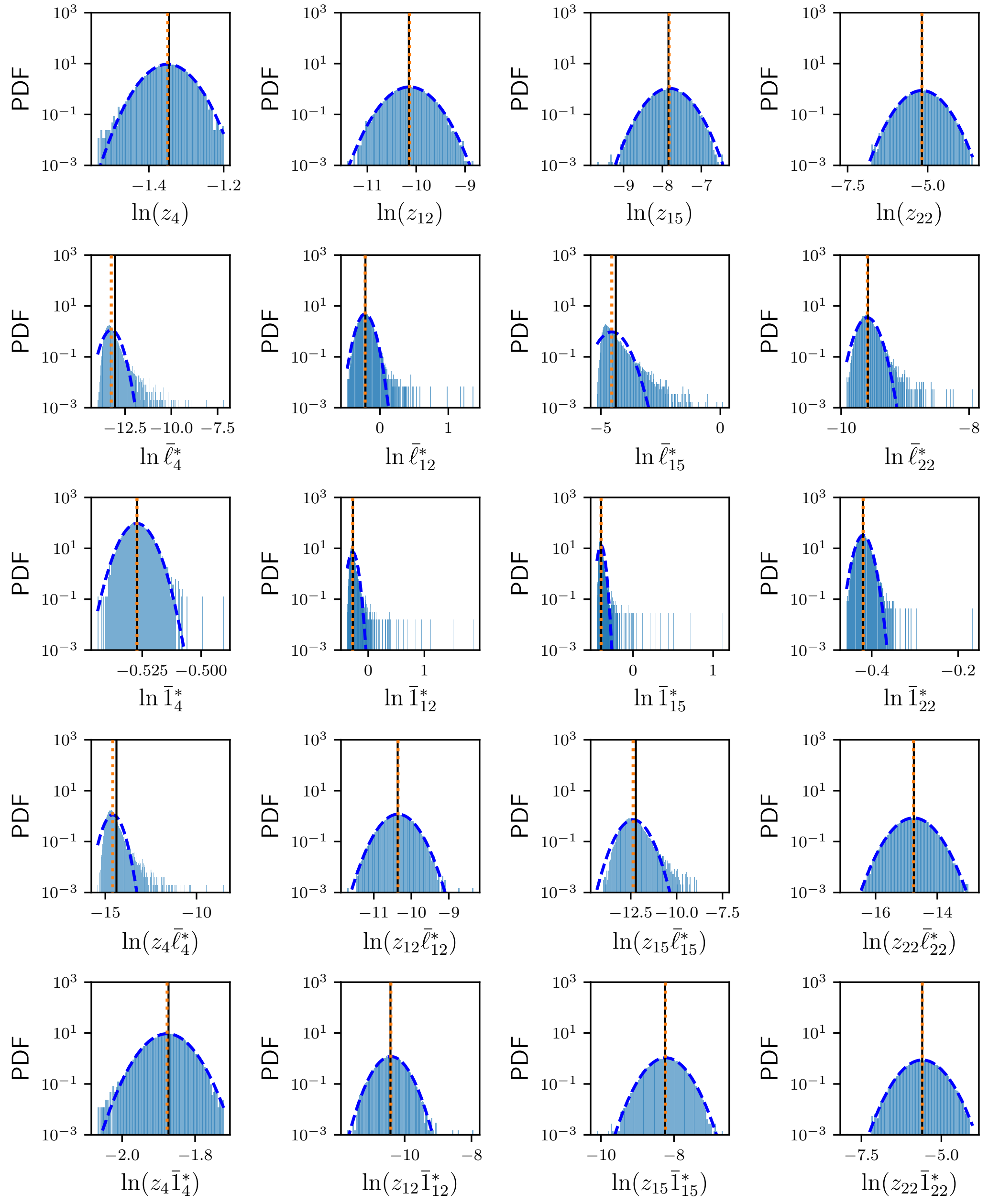


Figure 5.15: Probability density functions of the various quantities representative of Table 5.1. Data is obtained from sampling with batch size 1024, with a fixed neural network obtained from the BKE–FTS(US)+SL method at the same batch size. Dashed blue lines are log-normal distributions fitted using the method of moments [236], while the vertical dotted orange and solid black lines correspond to the mean of the histograms and the corresponding ensemble average computed via numerical integration, respectively.

PDFs for $\ln z_\alpha$, and it is expected that these tails disappear as the batch size is increased since this leads to free-energy differences that further obey a normal distribution. To summarize the statistics observed in all replicas, we group replicas with similar behaviors into four groups, corresponding to the reactant (1-10), transition (11-13), metastable (14-18), and product (19-24) states. The results for z_α for these groups are shown in the second column of Table 5.1.

Replicas	z_α	$\bar{\ell}^*_\alpha$	$\bar{1}^*_\alpha$	$z_\alpha\bar{\ell}^*_\alpha$	$z_\alpha\bar{1}^*_\alpha$	$\mathrm{Var}(\ln z_\alpha) > \mathrm{Var}(\ln \bar{\ell}^*_\alpha)$	$\mathrm{Var}(\ln z_\alpha) > \mathrm{Var}(\ln \bar{1}^*_\alpha)$
Reactant State (1-10)	✓	✗	✗	✗	✓	✗	✓
Transition State (11-13)	✓	✓	✗	✓	✓	✓	✓
Metastable State (14-18)	✓	✗	✗	✗	✓	✗	✓
Product State (19-24)	✓	✓	✗	✓	✓	✓	✓

Table 5.1: Summary of error analysis results. For columns two through six, ✓ indicates the distributions in all or most replicas are log-normal, while ✗ indicates the distributions in all or most replicas are approximately log-normal with a skew or slight deviations in the tails. For columns seven and eight, they indicate if the inequality holds. The corresponding histograms for columns two through six can be found in Figs. B.8–B.12.

With the sampling distributions of z_α understood, we now study the sampling distributions for $\bar{\ell}^*_\alpha$ and $\bar{1}^*_\alpha$. Assuming the values of ℓ^*_α and 1^*_α are independent and identically distributed, one may expect the corresponding sample means $\bar{\ell}^*_\alpha$ and $\bar{1}^*_\alpha$ to be normally distributed according to the central limit theorem. However, we observe from simulations that these sample means are better described by log-normal distributions; see the second and third rows of Fig. 5.15 for representative histograms, and Figs. B.9 and B.10 for all histograms. Since log-normality arises when normally-distributed random variables are exponentiated, its origin is likely due to the sums of exponentials in $c(\mathbf{x};\boldsymbol{\theta}_k)$ for $\bar{1}^*_\alpha$, and the neural network model $\hat{q}(\mathbf{x};\boldsymbol{\theta}_k)$ for $\bar{\ell}^*_\alpha$, where the output layer of $\hat{q}(\mathbf{x};\boldsymbol{\theta}_k)$ contains the sigmoidal function $\sigma(s) = 1/(1+e^{-s})$. Nevertheless, the distributions possess tails that render the log-normality only approximate in nature. We summarize these observations in the third and fourth columns of Table 5.1.

Despite the approximate log-normality in $\bar{\ell}^*_\alpha$ and $\bar{1}^*_\alpha$, one need not understand accurately the distributions of $\bar{\ell}^*_\alpha$ and $\bar{1}^*_\alpha$, as the distributions obtained for the products $z_\alpha\bar{\ell}^*_\alpha$ and $z_\alpha\bar{1}^*_\alpha$, which are needed by the estimator in Eq. (5.6.11), are log-normal; see the fourth and fifth rows of Fig. 5.15 for representative histograms, and Figs. B.11 and B.12 for all histograms, as well as the fifth and sixth columns of Table 5.1 for a concise summary. The only exceptions are the histograms for $z_\alpha\bar{\ell}^*_\alpha$ at the reactant (1-10) and metastable state (14-18), which have slightly skewed log-normal behavior. However, these do not contribute significantly to the overall BKE loss when compared to the transition state. To understand why log-normality emerges again for $z_\alpha\bar{\ell}^*_\alpha$ and $z_\alpha\bar{1}^*_\alpha$, let us convert the products into sums by taking the logarithm, so that $\ln z_\alpha \bar{\ell}^*_\alpha = \ln z_\alpha + \ln \bar{\ell}^*_\alpha$ and $\ln z_\alpha \bar{1}^*_\alpha = \ln z_\alpha + \ln \bar{1}^*_\alpha$. The distribution of the sum of two independent random variables, denoted more generally as $Y = X_1 + X_2$, can be obtained from the distributions for X_1 and X_2 in terms of a convolution

$$\rho_Y(y) = \int_{-\infty}^{\infty} \mathrm{d}x \; \rho_{X_1}(x)\rho_{X_2}(y-x)\,. \tag{5.6.15}$$

When one random variable, e.g., X_2, possesses a much lower variance than the other random variable, we expect that the value of X_2 will be constant relative to X_1. In this limit, we may approximate $\rho_{X_2}(x)$ with a Dirac delta function to yield

$$\rho_Y(y) \approx \int_{-\infty}^{\infty} \mathrm{d}x \; \rho_{X_1}(x)\delta(y-x) = \rho_{X_1}(y)\,. \tag{5.6.16}$$

Thus, the distribution for the sum is solely determined by the distribution of the random variable with the highest variance. Although this argument is only a weak approximation, as the random variables involved in $z_\alpha \bar{\ell}^*_\alpha$ and $z_\alpha \bar{1}^*_\alpha$ are correlated due to being processed from the same $\mathbf{x}$ values, it gives an insight as to why $z_\alpha \bar{\ell}^*_\alpha$ and $z_\alpha \bar{1}^*_\alpha$ are log-normally distributed. Note that the true distributions of $\ln \bar{\ell}^*_\alpha$ and $\ln \bar{1}^*_\alpha$ are not exactly known, but the distributions of $\ln z_\alpha$ consist of normal distributions. If $\ln z_\alpha$ possesses a larger variance than $\ln \bar{\ell}^*_\alpha$ or $\ln \bar{1}^*_\alpha$ we expect from Eq. (5.6.16) that the distribution of the sum in $\ln z_\alpha \bar{\ell}^*_\alpha$ and $\ln z_\alpha \bar{1}^*_\alpha$ matches the normal distribution of $\ln z_\alpha$. This argument is verified in the seventh and eighth columns of Table 5.1, where we see that $\ln z_\alpha \bar{\ell}^*_\alpha$ and $\ln z_\alpha \bar{1}^*_\alpha$ are normally distributed whenever $\ln z_\alpha$ possess higher variance.

With the log-normality of $z_\alpha \bar{\ell}^*_\alpha$ and $z_\alpha \bar{1}^*_\alpha$ verified, we can examine the numerator $\sum_{\alpha=1}^{M} z_\alpha \bar{\ell}^*_\alpha$ and denominator $\sum_{\alpha=1}^{M} z_\alpha \bar{1}^*_\alpha$ of Eq. (5.6.10), which make up the on-the-fly average BKE loss. Since the sum of log-normal random variables can be approximately described by a log-normal distribution [238–242], both the numerator and denominator should be approximately log-normal. From simulations, we find that the numerator is log-normally distributed (Fig. 5.16(a)) while the denominator is log-normally distributed with slight deviations in the tails (Fig. 5.16(b)). Since the ratio of two log-normal random variables is also log-normal, the resulting on-the-fly BKE loss should be log-normal, as shown in Fig. 5.16(c). This is also in agreement with what is observed during training (Fig. 5.12), and when the neural network is fixed (Fig. 5.13). Although the log-normality of the denominator is only approximate, one can use the previous argument on sums of random variables, i.e., Eq. (5.6.16), to show that the sampling distribution of the on-the-fly BKE loss is still log-normal, since the numerator has higher variance than the denominator, thereby allowing the log-normality of the numerator to dominate in the on-the-fly BKE loss. Given these results, we conclude that the on-the-fly estimates of the average BKE loss obtained from the BKE–FTS(US) and BKE–FTS(US)+SL methods are approximately log-normal.

Using the log-normal distribution of the average BKE loss, one can determine the asymptotic behavior of the sampling error as a function of batch size N_{batch}. Denoting the mean and variance of the log-normal distribution as μ and σ^2, respectively, we expect that

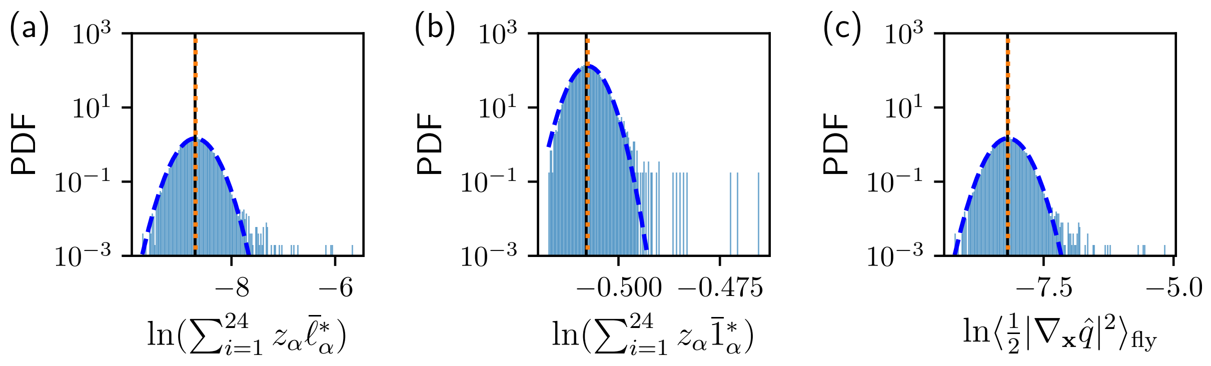


Figure 5.16: Probability density functions of sums of $z_\alpha\bar{\ell}_\alpha^*$, $z_\alpha\bar{1}_\alpha^*$, and the on-the-fly estimate of the BKE loss function in logarithmic space. Data is obtained from sampling with batch size 1024, with a fixed neural network obtained from the BKE–FTS(US)+SL method at the same batch size. Dashed blue lines are log-normal distributions fitted using the method of moments [236], while the vertical dotted orange and solid black lines correspond to the mean of the histograms and the corresponding ensemble average computed via numerical integration, respectively.

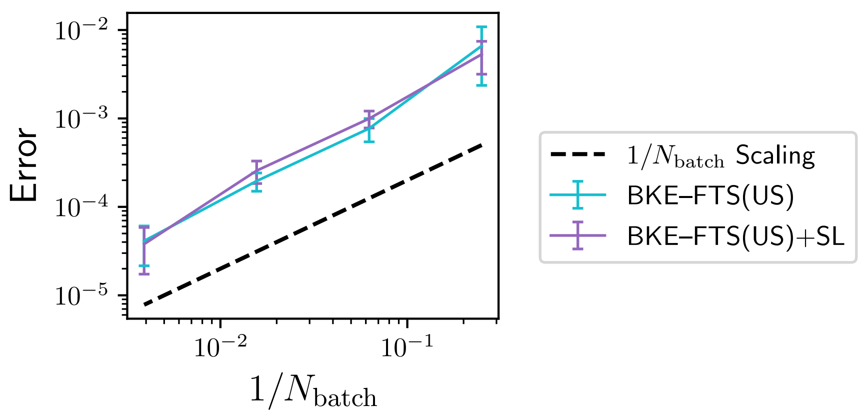


Figure 5.17: The absolute error in the on-the-fly BKE loss at different batch sizes, with respect to the largest batch size. All error bars are 95% confidence intervals.

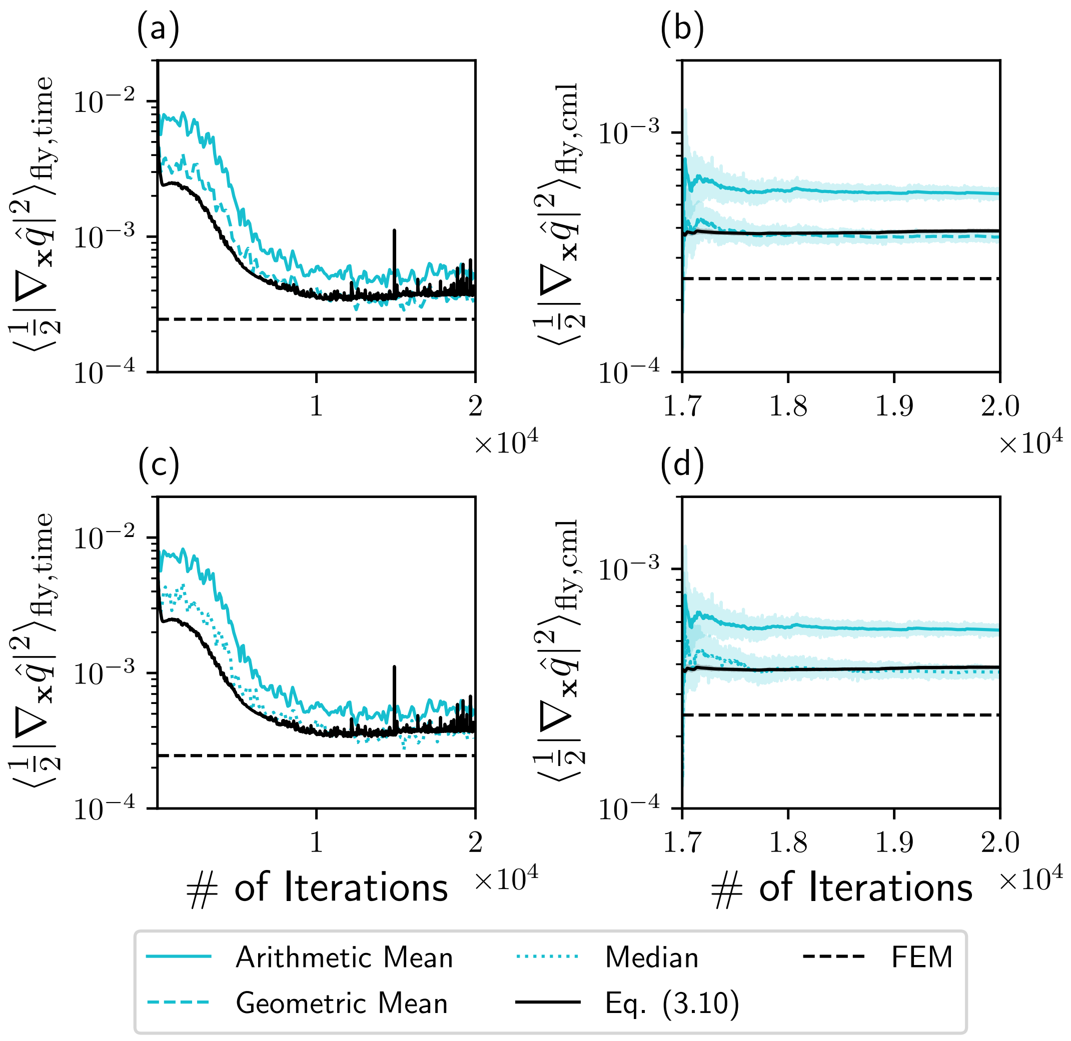


Figure 5.18: Using the geometric mean (a,b) and median (c,d) to remove the sampling error in the filtered on-the-fly estimates (left column) and cumulative average (right column) of the on-the-fly estimates in the BKE–FTS(US) method for batch size 64. Note that the remaining error between the FEM value and the average BKE loss computed per Eq. (5.6.9) is due to the inherent error of the chosen neural network. Cumulative mean and median are performed over the last 3000 iterations of the algorithm. Shaded colors in (b) and (d) are 95% confidence intervals.

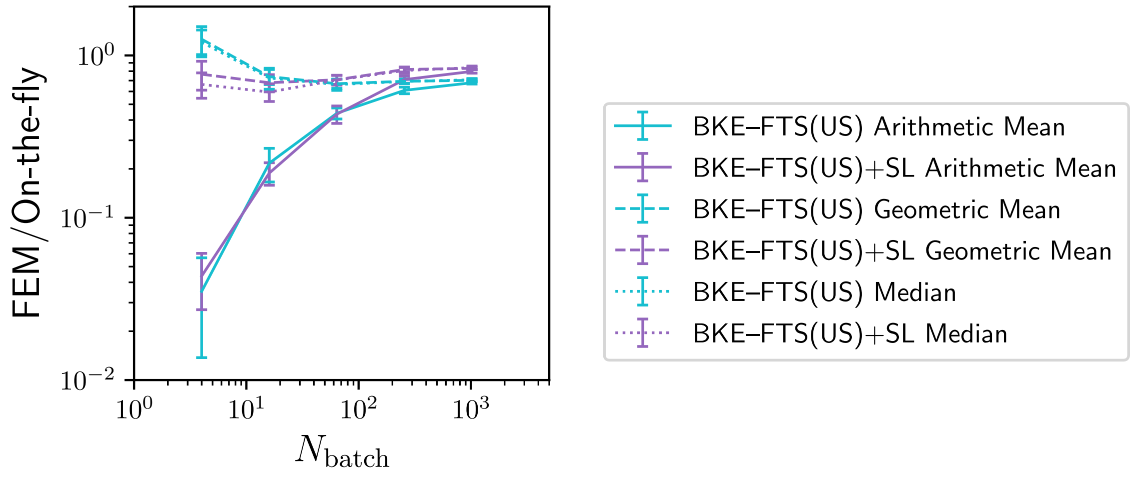


Figure 5.19: The ratio between FEM and on-the-fly estimates, after taking the geometric mean and median. Error bars are 95% confidence interval.

the cumulative mean of the on-the-fly BKE loss over iterations is given by [237]

$$\frac{1}{K-k^\star+1}\sum_{k=k^\star}^{K}\left\langle\frac{1}{2}|\nabla_{\mathbf{x}}\hat{q}(\mathbf{x};\boldsymbol{\theta}_{\mathrm{k}})|^2\right\rangle_{\text{fly}} \approx \exp\left(\mu+\frac{1}{2}\sigma^2\right), \tag{5.6.17}$$

where K is the final iteration index, and $k^\star$ is the iteration index when the on-the-fly estimates begin to fluctuate around a plateau. Equation (5.6.17) implies that the cumulative mean of on-the-fly estimates is always multiplied by a factor $\exp\left(\frac{1}{2}\sigma^2\right) > 1$, since $\sigma^2 > 0$. This explains why the on-the-fly estimates in Fig. 5.9(a) from both the BKE–FTS(US) and BKE–FTS(US)+SL methods are larger than the FEM value, and why the ratio between the FEM value and the on-the-fly estimates in Fig. 5.10 is always less than one. Furthermore, $\sigma^2 \sim O(1/N_{\text{batch}})$, implying for large N_{batch} that

$$\frac{1}{K-k^\star+1}\sum_{k=k^\star}^{K}\left\langle\frac{1}{2}|\nabla_{\mathbf{x}}\hat{q}(\mathbf{x};\boldsymbol{\theta}_{\mathrm{k}})|^2\right\rangle_{\text{fly}} \sim \exp(\mu)\left(1+O(1/N_{\text{batch}})\right), \tag{5.6.18}$$

thus showing the sampling error in the on-the-fly estimates scales as $O(1/N_{\text{batch}})$. Defining the absolute error as the difference between the cumulative mean of the on-the-fly estimates obtained at smaller batch sizes and the one obtained at the largest batch size, we plot the absolute error as a function of N_{batch} in Figure 5.11 for both the BKE–FTS(US) and BKE–FTS(US)+SL methods, where the $O(1/N_{\text{batch}})$ scaling can be observed.

The knowledge of the log-normal distribution can also be used to remove the sampling error between the on-the-fly estimates and the ensemble-averaged loss computed by numerical integration (Eq. (5.6.9)). This can be achieved by taking the median and geometric mean of the on-the-fly estimates since they are equal to the true mean $\exp(\mu)$ for log-normally distributed random variables [237]. We demonstrate this by applying the geometric mean (Figs. 5.18(a,b)) and median (Figs. 5.18(c,d)) to remove the sampling error in the filtered on-the-fly estimates and the cumulative mean from the BKE–FTS(US) method.

Furthermore, the geometric mean or median can be used to obtain similar accuracy in the average BKE loss across all batch sizes, as seen in Fig. 5.19 where we plot the ratio between the FEM value and the geometric mean and median of the on-the-fly estimates from the BKE–FTS(US) and BKE–FTS(US)+SL methods. Note that the ratio obtained from the BKE–FTS(US) method at the smallest batch size is larger than one, in contrast to the expected log-normal prediction that is less than one, but this result is consistent with the presence of the tails in the histograms for the smallest batch size; see Figs. 5.12(a) and 5.13(a). Nevertheless, the accuracy obtained from the smallest batch size after applying the geometric mean and median is comparable to the accuracy obtained from the largest batch size. Thus, one can use the BKE–FTS(US) and BKE-FTS(US)+SL methods to train neural networks with smaller batch sizes, which results in cheaper simulation costs, without loss in the accuracy in the reaction rates estimated on-the-fly.

5.7 Computational Study of a Solvated Dimer System

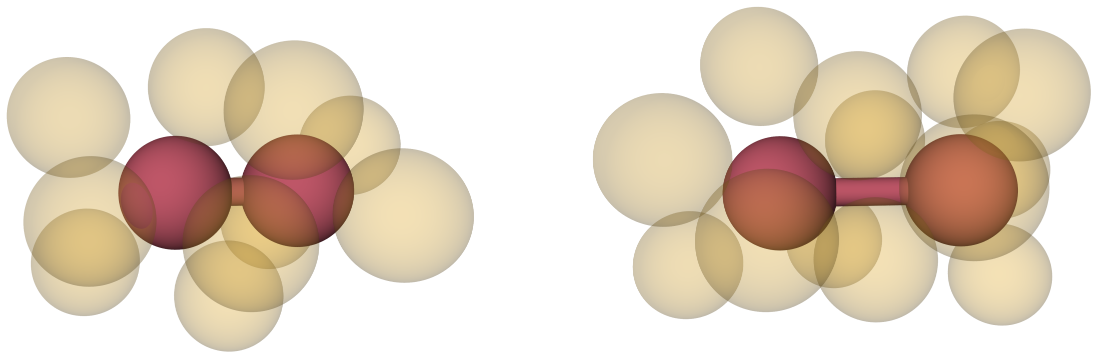


Figure 5.20: Dimer particles in (left) compact and (right) extended states for $r_0 = 2^{1/6}$ and $s = 0.25$ for a system with $\rho = 0.9$. Only the seven nearest neighbors of each solvent particle are visualized and made transparent. Image created using Ovito [31].

Until now, all previous studies correspond to a single particle diffusing in low-dimensional energy landscapes where a reference solution for $q(\mathbf{x})$ is known through analytical or numerical methods, allowing us to understand the accuracy of the proposed methods. However, the neural network representation of the committor function can also be employed in molecular systems with a high-dimensional configuration space with no reference solution, demonstrating the applicability of the proposed methods. To this end, we now test Algorithms 4–9 on a solvated dimer system [243], where the dimer transitions between a compact and an extended state; see Fig. 5.20. In what follows, we compute the committor function and reaction rate corresponding to the transition between the compact and the extended states of the dimer.

In this system, the dimer particles interact via a bond potential given by

$$V_{\mathrm{dimer}}\left(r\right) = h\left[1 - \frac{\left(r - r_0 - s\right)^2}{s^2}\right]^2 , \tag{5.7.1}$$

where r is the distance between the particles, $h = 5.0\ k_{\mathrm{B}}T$ is the height of the barrier, $r_0 = 2^{1/6}$ sets the distance in the compact state, and $s = 0.25$ sets the distance in the extended state. The distance in the compact state is $r = r_0$, and the distance in the extended state is $r = r_0 + 2s$ (Fig. 5.20). The solvent particles interact between themselves and the dimer particles by the Weeks-Chandler-Andersen potential [244]

$$V_{\mathrm{WCA}}\left(r\right) = \left(4\epsilon\left[\left(\frac{1}{r}\right)^{12} - \left(\frac{1}{r}\right)^{6}\right] + \epsilon\right)\Theta\left(r_{\mathrm{WCA}} - r\right) , \tag{5.7.2}$$

where $\epsilon = 1.0$, $r_{\mathrm{WCA}} = 2^{1/6}$, and $\Theta\left(x\right)$ is the Heaviside function. We test all the methods on systems of densities 0.05, 0.4, and 0.7 with a dimer and 30 solvent particles, and a system of density 0.9 with a dimer and 46 solvent particles. For all systems, the temperature is maintained at $k_{\mathrm{B}}T = 1$.

In comparison to the low-dimensional systems, molecular systems may have many particles with different species identities. To increase efficiency in training, the neural network should satisfy invariances with respect to translations, rotations, and permutations of the particle positions $\mathbf{x}$ and species identities $\mathbf{z}$. To this end, we use a neural network of the form

$$\hat{q}(\mathbf{x}, \mathbf{z}; \boldsymbol{\theta}) = \sigma\left(f(\mathbf{x}, \mathbf{z}; \boldsymbol{\theta})\right) , \tag{5.7.3}$$

where the species identities $\mathbf{z}$ correspond to $z = 1$ for a dimer particle and $z = 0$ for a solvent particle, and $f(\mathbf{x}, \mathbf{z}; \boldsymbol{\theta})$ being the implementation of SchNet [245] available with PyTorch Geometric [246]. SchNet is a message-passing neural network that determines the contribution to the committor function for each particle, satisfying permutation invariance of the particle identities, using a scheme dependent only on the distances between particles, satisfying the aforementioned translational and rotational invariances. SchNet first maps for each particle a high dimensional feature vector that is obtained from an embedding of the particle identities. The feature vectors are then updated using continuous-filter convolutions over the relative distances of a particle to its neighboring particles, which incorporate information about the particle environment; these operations are termed interaction blocks. The use of the feature vectors and interaction blocks allows for SchNet to learn the effect of particle environments on the per particle contribution to the committor function without the use of handcrafted descriptors. The feature vectors are then reduced into a scalar per particle contribution to the committor function through a dense neural network, which are summed together and passed through a sigmoid to obtain the neural network representation of the committor function. In this work, we use a feature vector size of 64 and 3 interaction blocks and perform the continuous-filter convolution for each particle over all other particles. For details on the associated hyper-parameters for each study and parameters used for BKE–US, BKE–FTS(ME), and BKE–FTS(US), see Appendix B.2.3. See also Ref. [245] for more details on the general architecture of SchNet and our code repository[2] for its implementation in this work.

We apply the same training procedure as done for the 1D and 2D systems with the BKE–US, BKE–FTS(ME), and BKE–FTS(US) methods plus their SL variants, where all methods use 24 replicas of a batch size of 8 samples collected every 25 steps. Initial configurations for sampling are obtained using umbrella sampling simulations with respect to the dimer bond distance r with a potential of the form

$$W_\alpha = \frac{1}{2}\kappa_\alpha \left(r - r_\alpha\right)^2 \,, \tag{5.7.4}$$

where $\kappa_\alpha = 1200\ k_\mathrm{B}T$ and $r_\alpha = 0.75 + \frac{1.9-0.75}{31}\left(\alpha - 1\right)$ for $\alpha \in [1, 32]$. These simulations generate a set of equilibrium configurations corresponding to the reactant, product, and in-between states. Furthermore, they are used to initialize the neural network and evaluate the quality of the trained neural network with a fixed data set. This data set consists of 10^4 samples per umbrella sampling replica generated from simulations of length 10^7 time steps with a sampling period of 10^3 time steps.

The neural network initialization is done through a similar procedure as described in Section 5.6. The neural network parameters are initialized randomly, and updated by minimizing Eq. (5.6.2) using Adam with a stepsize of $1 \cdot 10^{-5}$ until $I(\boldsymbol{\theta}) \leq 10^{-4}$. The initial configurations $\mathbf{x}_0^\alpha$ are chosen to be the configurations obtained using the above umbrella sampling procedure with bond distances closest to $r_\alpha = 0.98 + \frac{1.75-0.98}{23}(\alpha - 1)$ for $\alpha \in [1, 24]$. As in the previous 1D and 2D cases, the BKE–FTS(ME) and BKE–FTS(ME)+SL methods use $\boldsymbol{\varphi}_0^\alpha = \mathbf{x}_0^\alpha$, and the BKE–FTS(US) and the BKE–FTS(US)+SL methods sets $\mathbf{x}_0^\alpha$ to be the nodal point $\boldsymbol{\varphi}^\alpha$ of the converged path. All additional details related to sampling schemes generating mini-batches for optimization, penalty strengths, and parameters controlling the FTS method can be found in the Appendix B.2.3.

Figure 5.21 shows the on-the-fly estimates of the reaction rates or the average BKE loss from all methods tested on various densities for a batch size of 8 samples. For densities of 0.05, 0.4, and 0.7 (Fig. 5.21(a-c)), the BKE–FTS(ME) and BKE–FTS(US) estimates plateau around the same value near the estimate obtained from direct simulation, while BKE–US has high variance around a different plateau. For a density of 0.9 all methods plateau around the same value. As with the low-dimensional systems, the BKE–FTS(ME) and BKE–FTS(US)

[2] https://github.com/muhammadhasyim/tps-torch

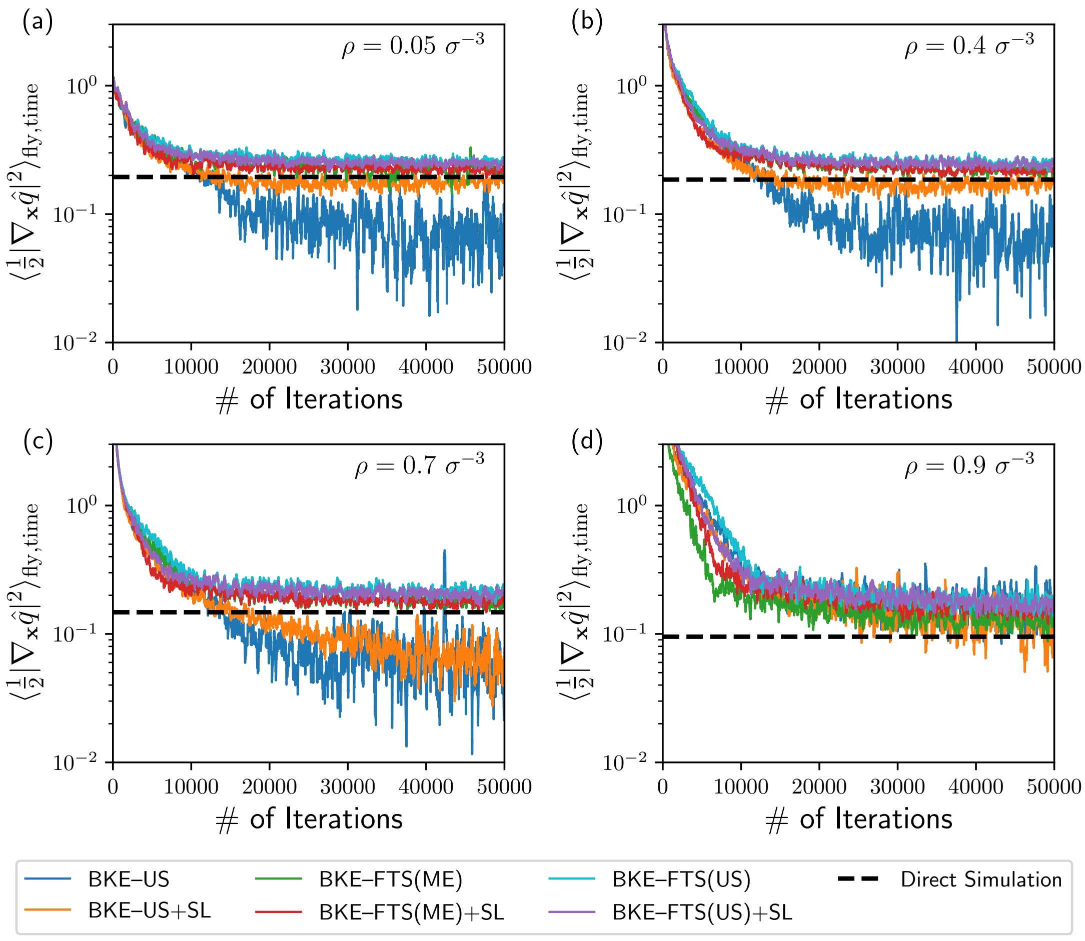


Figure 5.21: The filtered on-the-fly estimate of the BKE loss obtained at every iteration for the solvated dimer system, with the filtering window set to 200 iterations. A total of 10^4 unbiased trajectories are used to compute a direct estimate of the reaction rate (dashed line) for comparison with the proposed methods.

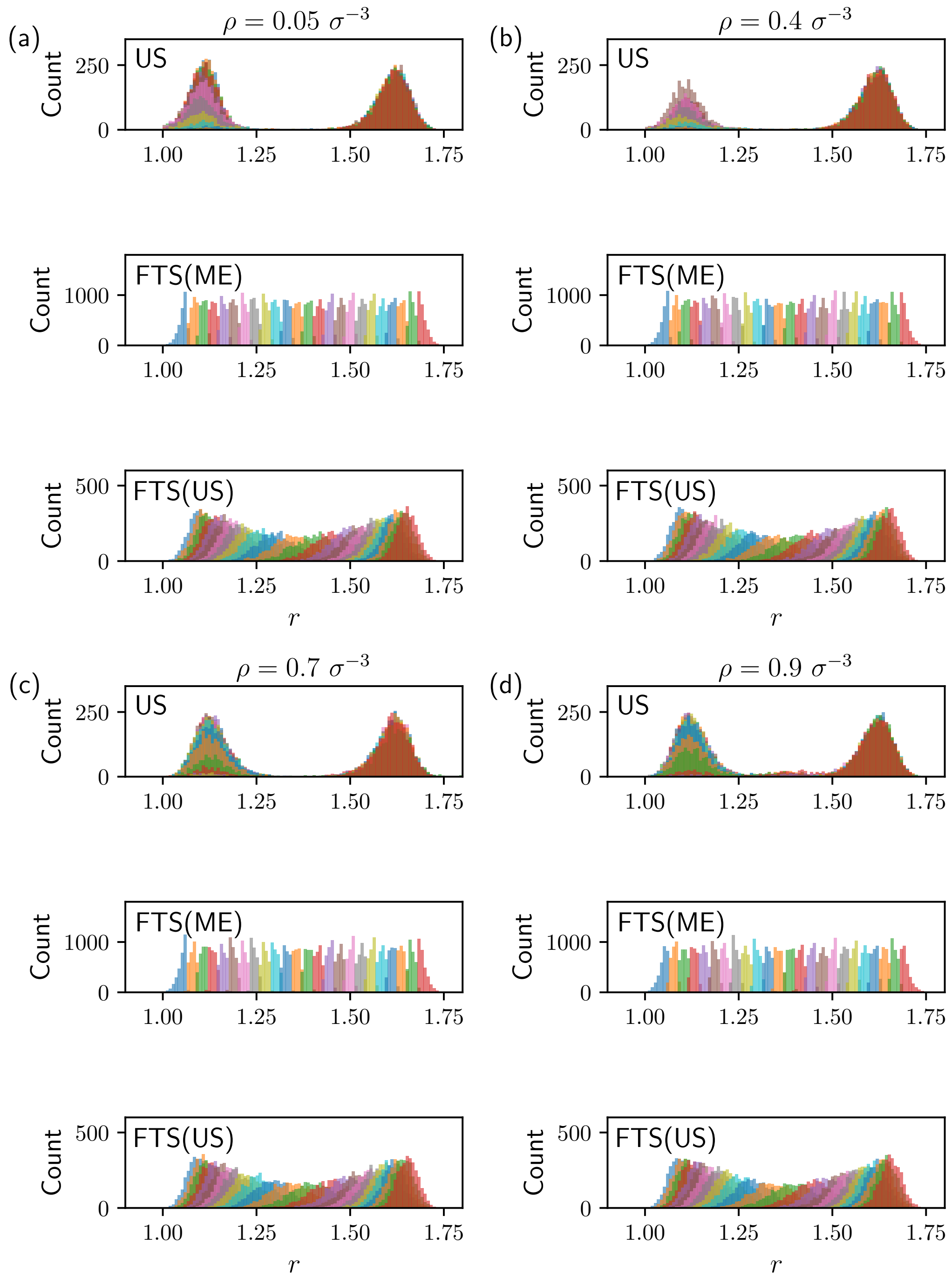


Figure 5.22: Histograms of dimer distances obtained from the BKE–US method (top), the BKE–FTS(ME) method (middle), and the BKE–FTS(US) method (bottom) for (a) $\rho = 0.05$, (b) $\rho = 0.4$, (c) $\rho = 0.7$, and (d) $\rho = 0.9$.

methods sample the reaction pathway, corresponding to dimer distances between the compact and extended states, homogeneously across all densities. In contrast, the BKE–US method does not homogeneously sample the reaction pathway although the transition state is better sampled at $\rho = 0.9$ compared to lower densities (Fig. 5.22). This behavior results in slightly improved overlaps between samples from the reactant/product state and the transition state, which may explain why the reasonable agreement is obtained between the BKE–US method and the direct estimate at $\rho = 0.9$.

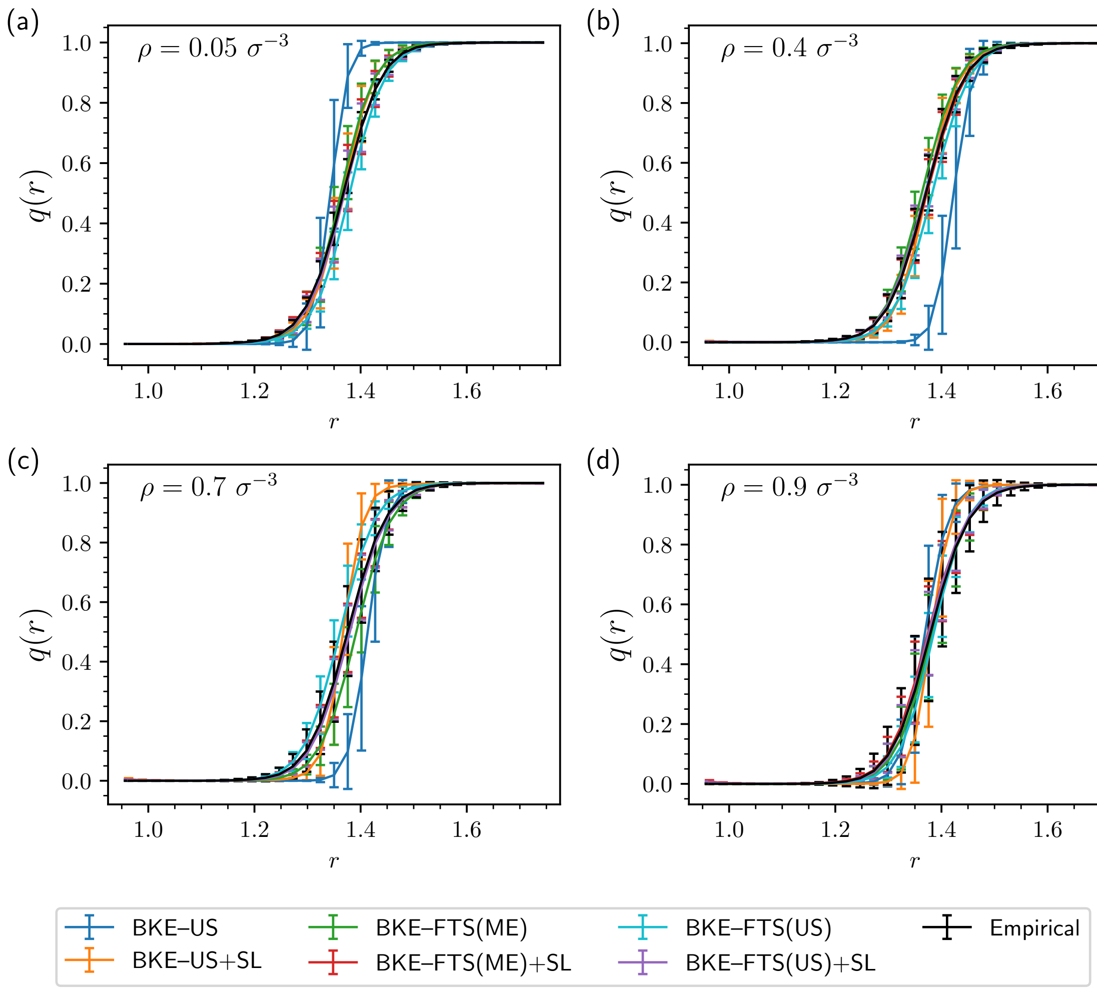


Figure 5.23: Average committor profiles for the methods compared to the empirical results for dimer in solvent systems. The values are binned for 31 windows between $r_{\text{min}} = 0.95$ and $r_{\text{max}} = 1.75$.

The accuracy of all methods can be assessed by comparing an empirical committor function $q_{\text{emp}}(r)$ computed at a fixed value of bond length r with the corresponding value $\hat{q}(r, \mathbf{z}; \boldsymbol{\theta})$

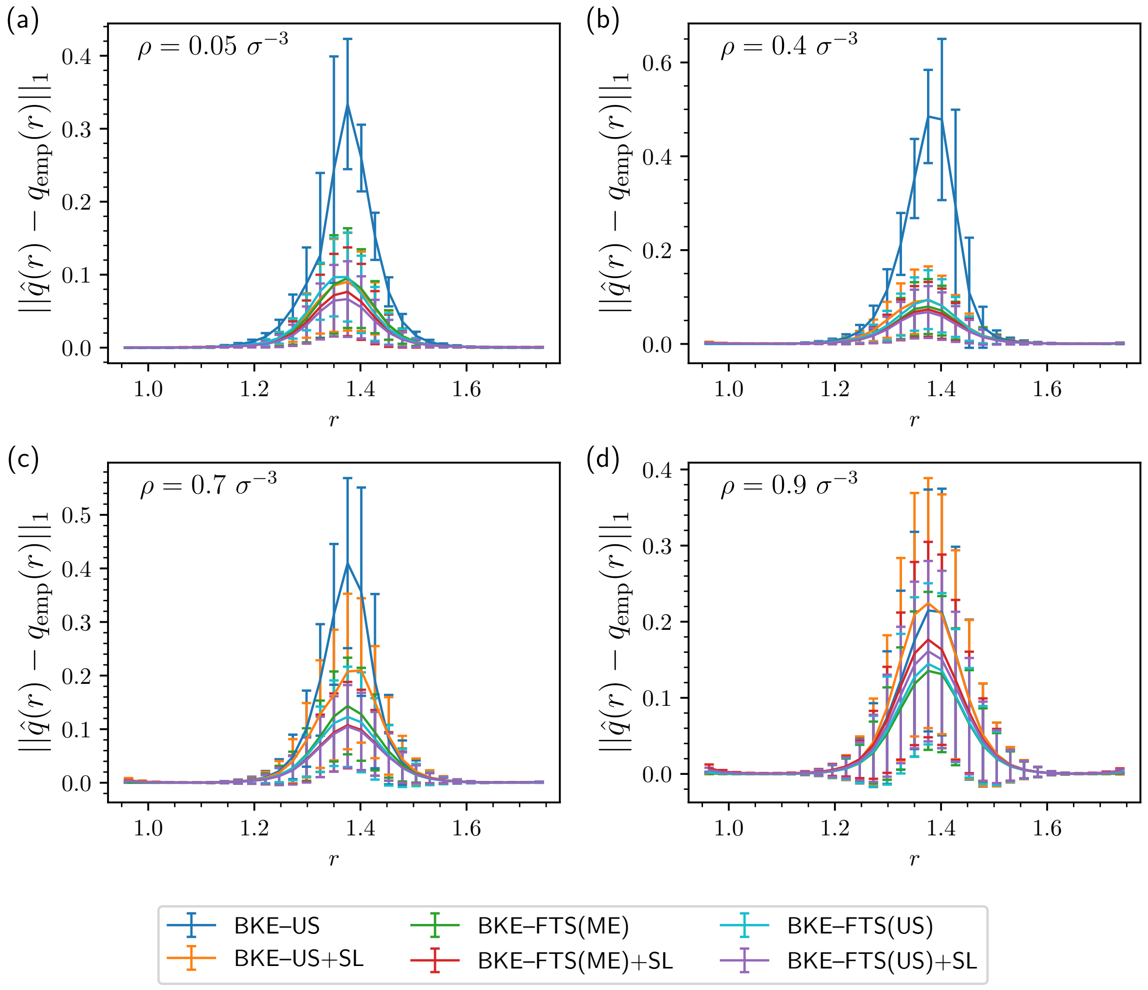


Figure 5.24: Mean absolute error profiles for the methods compared to the empirical results for dimer in solvent systems. The values are binned for 31 windows between $r_{\min} = 0.95$ and $r_{\max} = 1.75$.

obtained from the neural network. At fixed r, the committor values are spread across a distribution since the committor depends not only on r but also on solvent configurations. Thus, both $q_{\text{emp}}(r)$ and $\hat{q}(r, \mathbf{z}; \boldsymbol{\theta})$ represent estimates of the mean committor at fixed r. Given the full empirical committor function $q_{\text{emp}}(\mathbf{x})$ (Eq. (5.4.1)) and neural network $\hat{q}(\mathbf{x}, \mathbf{z}; \boldsymbol{\theta})$, we can compute these means via a binning procedure. Letting $\mathcal{Q}_i$ be a set of configurations such that every $\mathbf{x} \in \mathcal{Q}_i$ satisfies $r \in (r_{i-1}, r_i]$, the binning procedure yields the following formulas:

$$\hat{q}(r_i, \mathbf{z}; \boldsymbol{\theta}) = \frac{1}{|\mathcal{Q}_i|} \sum_{\mathbf{x} \in \mathcal{Q}_i} \hat{q}(\mathbf{x}, \mathbf{z}; \boldsymbol{\theta}) \,, \tag{5.7.5}$$

$$q_{\text{emp}}(r_i) = \frac{1}{|\mathcal{Q}_i|} \sum_{\mathbf{x} \in \mathcal{Q}_i} q_{\text{emp}}(\mathbf{x}) \,, \tag{5.7.6}$$

where every $\mathbf{x} \in \mathcal{Q}_i$ is obtained from the configurations sampled via the umbrella potential in Eq. (5.7.4) and $q_{\text{emp}}(\mathbf{x})$ is computed using 1250 trajectories per configuration $\mathbf{x}$. Figure 5.23 plots $q_{\text{emp}}(r_i)$ and $\hat{q}(r_i, \mathbf{z}; \boldsymbol{\theta})$ with their respective variances, which represent the intrinsic spread of committor values around their mean at $r = r_i$. We see that the BKE–US and BKE–US+SL methods have a systematic difference between the average binned neural network and empirical values. Meanwhile, the BKE–FTS(ME) and BKE–FTS(US) have a slightly lower systematic difference, which decreases further upon the use of supervised learning.

We further assess the accuracy of all methods by computing the mean of absolute error between the binned values of the neural network committor and the empirical committor, i.e.,

$$\|\hat{q}(r_i) - q_{\text{emp}}(r_i)\|_1 = \frac{1}{|\mathcal{Q}_i|} \sum_{\mathbf{x} \in \mathcal{Q}_i} |\hat{q}(\mathbf{x}, \mathbf{z}; \boldsymbol{\theta}) - q_{\text{emp}}(\mathbf{x})| \,. \tag{5.7.7}$$

Figure 5.24 shows the mean of absolute errors for all densities, where we find that the error is the largest near $q(r) = 1/2$. Furthermore, we observe a hierarchy in the reduction of errors. For densities ρ of 0.05–0.7, the order of methods with increasing accuracy goes as BKE–US $<$ BKE–FTS(ME) $<$ BKE–FTS(US), and the addition of supervised learning improves the accuracy of each respective method.

We now assess the accuracy of the methods through the average BKE loss, and thereby the reaction rates. Unlike the low-dimensional studies, where the average BKE loss of the neural network can be evaluated via quadrature (Eq. (5.6.9)), numerically exact calculation is not possible in high-dimensional problems and a new scheme is needed. To this end, we choose umbrella sampling with a reweighting procedure to compute the average BKE loss with minimal sampling error. This new scheme utilizes the earlier dataset obtained for the initialization of the neural network as a validation dataset, where umbrella sampling with respect to Eq. (5.7.4) was used to obtain 10^4 configurations from all 32 replicas. Given this dataset, we compute the reweighting factors z_α using the multistate Bennett acceptance ratio (MBAR) method. Note that MBAR is used instead of FEP since it yields estimates of z_α with lower error than FEP, albeit at a higher computational cost [131]. Once the MBAR reweighting factors z_α^{MBAR} are computed, the reaction rate from the neural network can be

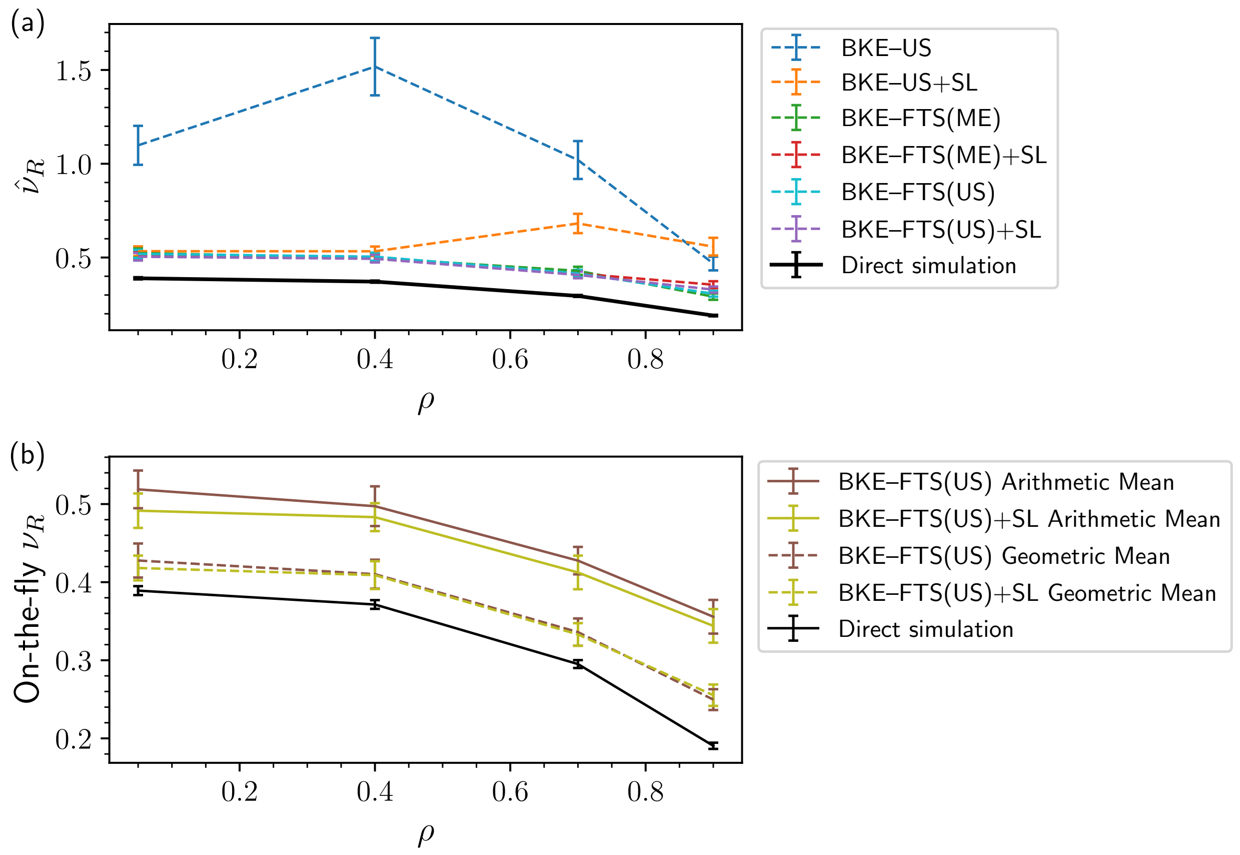


Figure 5.25: (a) The reaction rate $\hat{\nu}_R$ of the neural network per Eq. (5.7.8) as a function of density. (b) Comparison between the arithmetic mean and geometric mean applied to the last 3000 samples from training to direct simulation as a function of density. Error bars are 95% confidence interval.

estimated from a modification of Eq. (5.6.5) for umbrella sampling,

$$\hat{\nu}_R = \left\langle |\nabla_{\mathbf{x}} \hat{q}(\mathbf{x};\boldsymbol{\theta})|^2 \right\rangle = \frac{\displaystyle\sum_{\alpha=1}^{32} \frac{2z_\alpha^{\text{MBAR}}}{|\mathcal{M}^\alpha|} \sum_{\mathbf{x}\in\mathcal{M}^\alpha} \left[\frac{\ell(\mathbf{x};\boldsymbol{\theta})}{c(\mathbf{x};\boldsymbol{\theta})}\right]}{\displaystyle\sum_{\alpha=1}^{32} \frac{z_\alpha^{\text{MBAR}}}{|\mathcal{M}^\alpha|} \sum_{\mathbf{x}\in\mathcal{M}^\alpha} \left[\frac{1}{c(\mathbf{x};\boldsymbol{\theta})}\right]}. \tag{5.7.8}$$

Evaluating Eq. (5.7.8) produces the results seen in Fig. 5.25(a), which are compared to the true reaction rate as estimated by direct molecular simulation. The results in Fig. 5.25(a) mirror the trends seen in Fig. 5.24.

As established by the error analysis in Section 5.6.3, we may avoid costly computation in Eq. (5.7.8) for the BKE–FTS(US) and BKE–FTS(US)+SL methods via the geometric-mean estimate to eliminate sampling error at low batch sizes. The comparison between the arithmetic and geometric mean on the on-the-fly estimates, taken from the last portion of training, is shown in Fig. 5.25(b). Similar to the low-dimensional case, the geometric mean is able to recover estimates of the reaction rate closer to the true reaction rate than the arithmetic mean, demonstrating the generality of the results from the error analysis. Furthermore, the trend between the geometric mean agrees reasonably well with the true reaction rate across all densities. This result supports the points made in Section 5.5 that the BKE–FTS methods are able to account for solvent effects despite using a CV that ignores solvent configurations and thereby predicting the correct trend of the reaction rate as a function of density.

5.8 Conclusion and Future Work

In summary, building on the work of Ref. [10], we have introduced and discussed a set of ML-based algorithms for computing accurate and precise committor functions and reaction rates. Accuracy in computing committor functions is improved by adding elements of supervised learning, where committor values obtained from short molecular trajectories are used to improve the neural network training. On the other hand, accuracy in the estimated reaction rates is significantly improved by incorporating the FTS method, which allows homogeneous sampling across the transition tube necessary for obtaining accurate free energies and reweighting factors. Furthermore, for the FTS method via path-based umbrella sampling as in the BKE–FTS(US) and BKE–FTS(US)+SL method, we provide an error analysis, which shows that the on-the-fly estimates of the average BKE loss obey log-normal statistics. This analysis also shows that the sampling error in the on-the-fly estimates of reaction rates can be removed by computing its geometric mean or median. The different combinations of supervised learning and the FTS method yield five additional algorithms, which were tested against three model systems. Out of the six algorithms, we recommend the BKE–FTS(US)+SL method, which combines all the strengths of supervised learning and the FTS method, in conjunction with the geometric mean/median procedure that allows accurate and precise computation of reaction rates with a small number of samples, e.g., batch size of $O(10^1)$.

Future work involves investigating ways of further increasing the accuracy of the methods on molecular systems. The accuracy could likely be increased through the use of an equivariant neural network [247], with neural networks satisfying equivariance throughout the hidden layers having been shown to yield increased accuracy in predictions of molecular properties over SchNet [248]. Future work should also explore other model systems ranging from ionic association/dissociation in NaCl solution, where the transition pathway involves the association/dissociation of Na^+–Cl^- ionic pairs [192, 249–251], to excitation events in glassy systems, where the transition state is known to have elastic signatures that are crucial for the structural relaxation [252].

Appendix A

Appendix to Lipid Rafts

A.1 Additional Composition Results

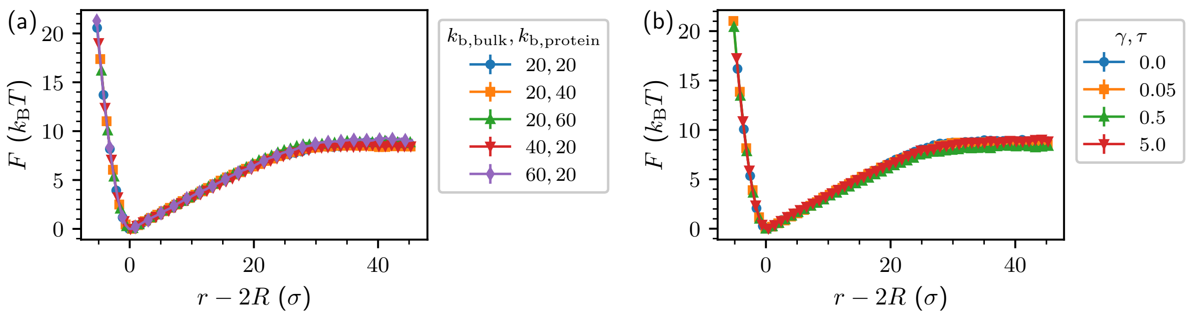


Figure A.1: Additional results for the free energy profiles of two orderphilic proteins at $k_{\rm b}\,(L_{\rm d}) = 20\ k_{\rm B}T$, $k_{\rm b}\,(L_{\rm o}) = 60\ k_{\rm B}T$ for a system with 10^4 total vertices. (a) Varying bending module for proteins of radii 6.4 σ, $J = 0.3\ k_{\rm B}T$, and $\mu_{\rm eq} - \mu = 10^{-4}\ k_{\rm B}T$. (b) Varying γ, τ for proteins of radii 6.4 σ, $J = 0.3\ k_{\rm B}T$, and $\mu_{\rm eq} - \mu = 10^{-4}\ k_{\rm B}T$.

This section contains additional results for Section 3.4. To begin, the free energy with respect to the distance between two orderphilic protein domains is evaluated using REUS and MBAR under variations in the bending rigidity and surface tension in Fig. A.1, in which the free energies have small dependence on those parameters. Additional statistics related to the phase diagram of multiple protein systems of Fig. 3.7(c) is evaluated in Fig. A.2. At $J = 0.3\ k_{\rm B}T$, the use of US and MBAR obtains the free energy with respect to average composition ϕ in Fig. A.2(a) in which it is found that the free energy minimum corresponding to the $\rm L_d$ phase becomes broader upon the addition of orderphilic proteins. From the free energy profiles, the line tension is evaluated. To do so, it is noted the line tension is defined by

$$\lambda = \frac{1}{L}\left(F_{\rm l} - F_{\rm b}\right)\ , \tag{A.1.1}$$

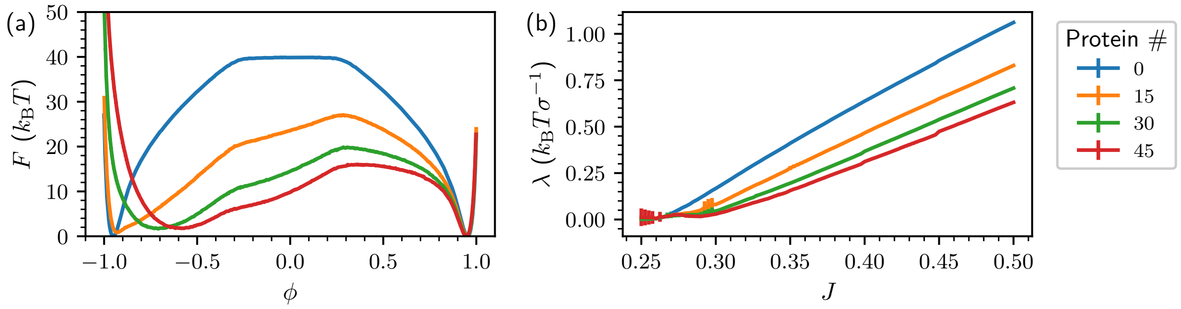


Figure A.2: Additional results for an orderphilic point particle system with 1600 total vertices, $k_b\,(L_d) = 20\ k_BT$, $k_b\,(L_o) = 60\ k_BT$, and reweighted such that each phase has equal probability. (a) Free energy as a function of ϕ for varying number of orderphilic proteins at $J = 0.3\ k_BT$. (b) Line tension as a function of J for varying number of orderphilic proteins.

where L is the length of the interface, F_l is the free energy of a system with an interface between the L_o–L_d phases, and F_b is the free energy of a system in the bulk phase. In a planar periodic L_o–L_d system Eq. (A.1.1) is evaluated as [253]

$$\lambda = \frac{1}{2L}\left[F\left(\phi_{\text{coex}}\right) - \frac{1}{2}\left(F\left(\phi_{L_o}\right) + F\left(\phi_{L_d}\right)\right)\right], \tag{A.1.2}$$

where $F\left(\phi\right)$ is the free energy at composition ϕ with ϕ_{L_o}, ϕ_{L_d}, and ϕ_{coex} corresponding to the compositions in the L_o phase, L_d phase, and coexistence between the phases respectively. Applying Eq. (A.1.2) to the data obtained using US and MBAR yields the line tensions in Fig. A.2(b), in which it is found that the line tension is reduced upon the addition of proteins.

A.2 Additional Bending Results

This section contains additional results for Section 3.5. Additional free energy versus $\bar{H}_{\text{protein}}$ profiles for a single protein system with varying surface tension are evaluated using REUS and MBAR in Fig. A.3. In Fig. A.3(a), the free energy profiles for a system with low k_b is evaluated in which surface tension is found to suppress spontaneous budding by making the flat state preferred. In Fig. A.3(b), the free energy profiles for a system with realistic k_b and governed by the composition energetics of Eqs. (3.5.2) and (3.5.3) is evaluated in which surface tension is found to slightly shift the profiles.

Additional free energy versus protein distance profiles for dimpled two protein systems with varying μ and J are evaluated using REUS and MBAR in Fig. A.4. The trends match the previous $C = 0$ cases seen in Figs. 3.6(b,d) with the addition of repulsion due to the curvature of the dimpled domains.

In Figs. A.5–A.7, additional cluster metrics are evaluated for planar and spherical systems. Figure A.5 corresponds to the results in Figs. 3.12,(b,c) in which C and $\mu_{\text{eq}} - \mu$ are changed

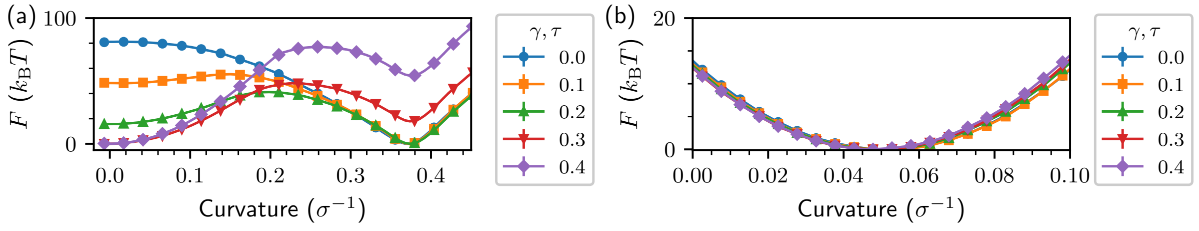


Figure A.3: (a) Free energy versus curvature for various surface and frame tensions at $k_\mathrm{b} = 2.5\ k_\mathrm{B}T$ for a domain of size 10.3 σ at $J = 1\ k_\mathrm{B}T$ in a system with 2500 total vertices. (b) Free energy versus curvature for various surface and frame tensions at $C = 0.05\ \sigma^{-1}$ for a single L_o domain of radius 4.8 σ at $k_\mathrm{b}\,(L_\mathrm{d}) = 20\ k_\mathrm{B}T$, $k_\mathrm{b}\,(L_\mathrm{o}) = 60\ k_\mathrm{B}T$, and composition energy governed by Eq. 3.5.2 with parameters given by Eq. 3.5.3 in a system with 2500 total vertices and 80 $\mathrm{L_o}$ vertices.

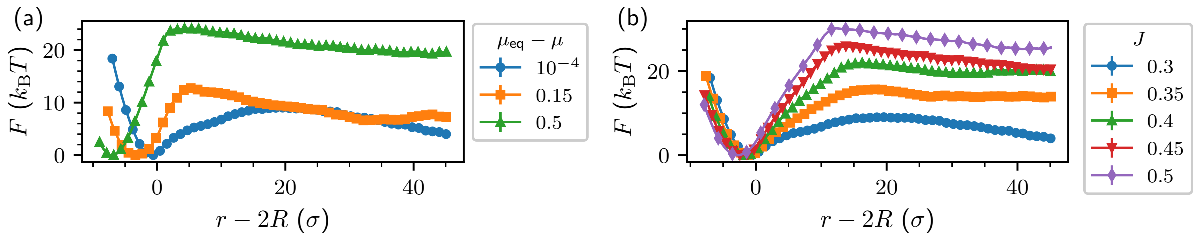


Figure A.4: The following are free energy versus protein distance for two orderphilic proteins with varying parameters. (a) Varying μ at $\gamma, \tau = 0$, $J = 0.3\ k_\mathrm{B}T$, $k_\mathrm{b}\,(L_\mathrm{d}) = 20\ k_\mathrm{B}T$, $k_\mathrm{b}\,(L_\mathrm{o}) = 60\ k_\mathrm{B}T$, and $C = 0.2\ \sigma^{-1}$ for two orderphilic proteins of radii 6.4 σ. (b) Varying J at $\gamma, \tau = 0$, $C = 0.2\ \sigma^{-1}$, $k_\mathrm{b}\,(L_\mathrm{d}) = 20\ k_\mathrm{B}T$, $k_\mathrm{b}\,(L_\mathrm{o}) = 60\ k_\mathrm{B}T$, and $\mu_\mathrm{eq} - \mu = 1\text{e-}4\ k_\mathrm{B}T$ for two orderphilic proteins of radii 6.4 σ.

at varying J for a planar system. Figure A.6 corresponds to the results in Fig. 3.14, in which C and $\mu_\mathrm{eq} - \mu$ are changed at varying surface tension for a planar system. Figure A.7 corresponds to the results in Fig. 3.15, in which C and $\mu_\mathrm{eq} - \mu$ are changed at varying J for a spherical system.

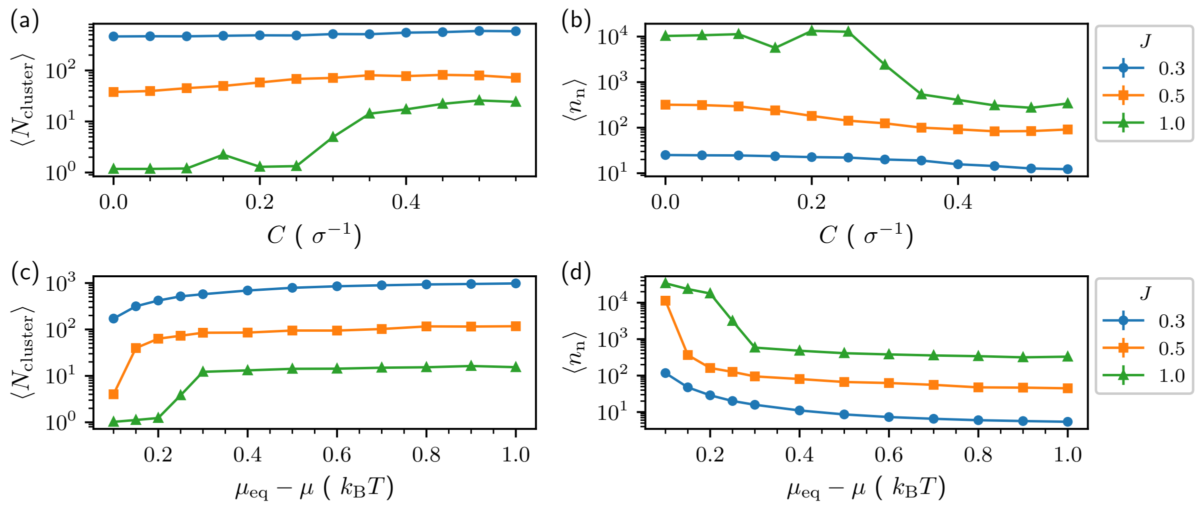


Figure A.5: (a,b) Cluster metrics for a system with 4800 orderphilic point proteins with 40000 total vertices at $k_{\mathrm{b}}\,(L_{\mathrm{d}}) = 20\ k_{\mathrm{B}}T$, $k_{\mathrm{b}}\,(L_{\mathrm{o}}) = 60\ k_{\mathrm{B}}T$, $\mu_{\mathrm{eq}} - \mu = 0.25\ k_{\mathrm{B}}T$, $\gamma, \tau = 0$, and varying C and J. (c,d) Cluster metrics for a system with 4800 orderphilic point proteins with 40000 total vertices at $k_{\mathrm{b}}\,(L_{\mathrm{d}}) = 20\ k_{\mathrm{B}}T$, $k_{\mathrm{b}}\,(L_{\mathrm{o}}) = 60\ k_{\mathrm{B}}T$, $C = 0.3\ \sigma^{-1}$, $\gamma, \tau = 0$, and varying μ and J.

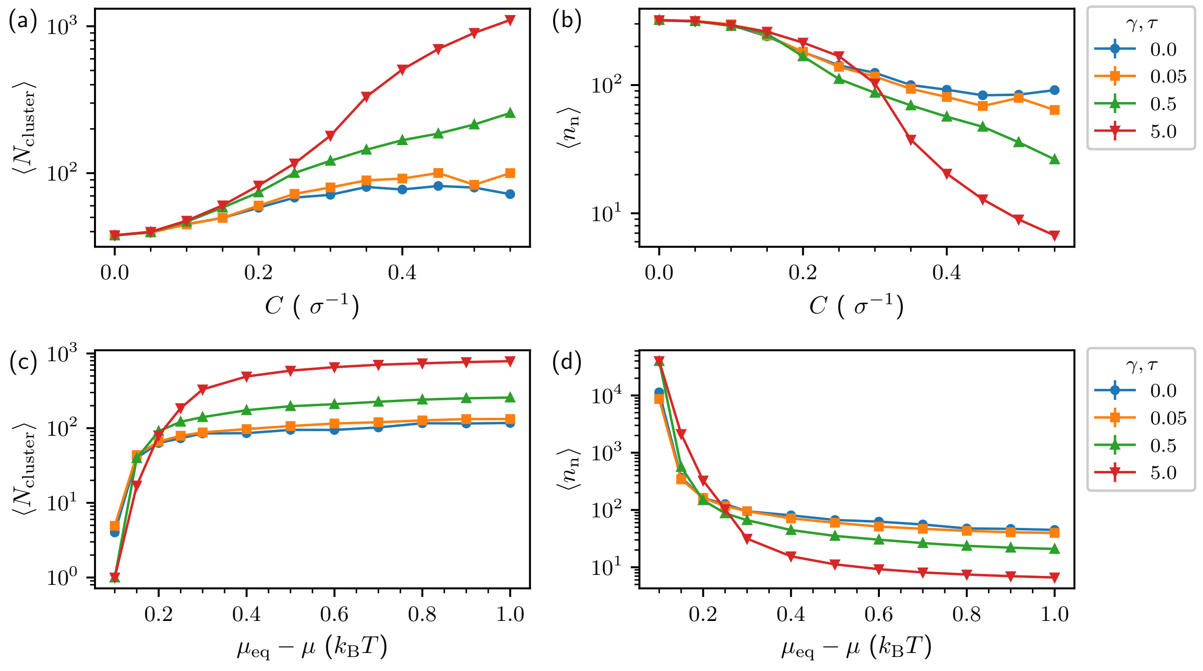


Figure A.6: (a,b) Cluster metrics for a system with 4800 orderphilic point proteins with 40000 total vertices at $k_{\mathrm{b}}\,(L_{\mathrm{d}}) = 20\ k_{\mathrm{B}}T$, $k_{\mathrm{b}}\,(L_{\mathrm{o}}) = 60\ k_{\mathrm{B}}T$, $\mu_{\mathrm{eq}} - \mu = 0.25\ k_{\mathrm{B}}T$, $J = 0.5\ k_{\mathrm{B}}T$, and varying γ, τ and C. (c,d) Cluster metrics for a system with 4800 orderphilic point proteins with 40000 total vertices at $k_{\mathrm{b}}\,(L_{\mathrm{d}}) = 20\ k_{\mathrm{B}}T$, $k_{\mathrm{b}}\,(L_{\mathrm{o}}) = 60\ k_{\mathrm{B}}T$, $C = 0.3\ \sigma^{-1}$, $J = 0.5\ k_{\mathrm{B}}T$, and varying γ, τ, and μ.

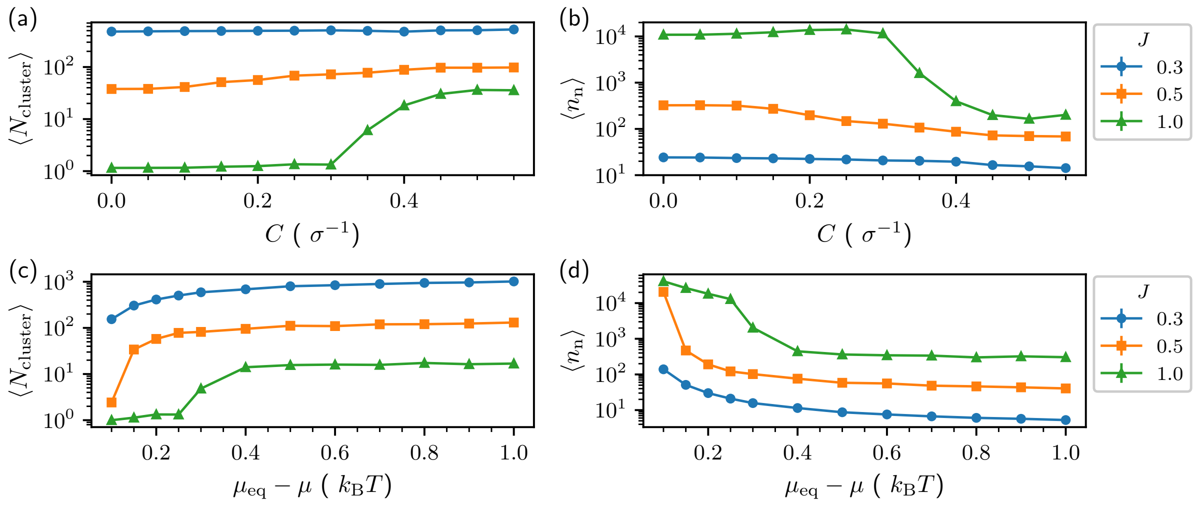


Figure A.7: (a,b) Cluster metrics for a spherical system with 4800 orderphilic point proteins with 40962 total vertices at $k_{\mathrm{b}}\,(L_{\mathrm{d}}) = 20\ k_{\mathrm{B}}T$, $k_{\mathrm{b}}\,(L_{\mathrm{o}}) = 60\ k_{\mathrm{B}}T$, $\mu_{\mathrm{eq}} - \mu = 0.25\ k_{\mathrm{B}}T$, $\gamma, \tau = 0$, and varying C and J. (c,d) Cluster metrics for a spherical system with 4800 orderphilic point proteins with 40962 total vertices at $k_{\mathrm{b}}\,(L_{\mathrm{d}}) = 20\ k_{\mathrm{B}}T$, $k_{\mathrm{b}}\,(L_{\mathrm{o}}) = 60\ k_{\mathrm{B}}T$, $C = 0.3\ \sigma^{-1}$, $\gamma, \tau = 0$, and varying μ and J.

Appendix B

Appendix to Committor Learning

B.1 Computing Reweighting Factors in the Master-Equation Approach

Recall that the reweighting factor z_α in the BKE–FTS(ME) and BKE–FTS(ME)+SL methods are computed by solving the master equation Eq. (5.5.4). One can re-write Eq. (5.5.4) as a matrix equation:

$$\mathbf{K}\mathbf{z} = \mathbf{0}\,, \tag{B.1.1}$$

where $\mathbf{z} = z_\alpha \mathbf{e}_\alpha$, $\mathbf{K} = K_{\alpha\alpha'} \mathbf{e}_\alpha \otimes \mathbf{e}_{\alpha'}$, and $K_{\alpha\alpha'} = k^T_{\alpha\alpha'}$ for $\alpha \neq \alpha'$ and $K_{\alpha\alpha} = -\sum_{\alpha'} k_{\alpha\alpha'}$. Since Eq. (B.1.1) defines $\mathbf{z}$ as the basis vector of the null-space of $\mathbf{K}$, one can use singular value decomposition (SVD) to factorize $\mathbf{K} = \mathbf{U}\boldsymbol{\Sigma}\mathbf{V}^T$, and set the solution $\mathbf{z}$ as the column vector of $\mathbf{V}$ corresponding to the zero singular value. One can then normalize the vector $\mathbf{z}$ to satisfy the constraint $\sum_{\alpha=1}^{M} z_\alpha = 1$.

In extremely short simulation runs, the off-diagonals of $N_{\alpha\alpha'}$ can be zero due to the absence of rejection counts, which may result in estimates of z_α, i.e., elements of column vector of $\mathbf{V}$, that are not strictly positive. To ensure that the algorithm computes the correct column vector, we shift the off-diagonals $k_{\alpha(\alpha+1)}$ and $k_{(\alpha-1)\alpha}$ by a tolerance value of $2 \cdot 10^{-9}$, i.e., slightly lower than the machine epsilon of single-precision floats, and set the tolerance for zero singular-value detection to be 10^{-6}. For this choice of tolerance values, the estimated z_α converge in the limit of large batch sizes to the z_α computed by numerical integration of Eq. (5.5.3); see Fig. B.4. Note that a range of tolerance values 10^{-11}–10^{-8} have also been used with no change to the results.

B.2 Computational Details on Optimization and Sampling

In this section, we provide additional details relevant to both the sampling and optimization steps of all algorithms. The simulation of multiple replicas are distributed with MPI and interfaced with PyTorch [254] for performing optimization[2].

B.2.1 First Study: 1D Quartic Potential

In the first study, umbrella sampling is performed with $M = 20$ replicas with dynamics described by the overdamped Langevin dynamics in Eq. (5.3.1). For committor-based umbrella sampling, bias potential parameters for each replica are set to $\kappa_\alpha = 50$ and $q_\alpha = \frac{\alpha-1}{M-1}$. For the path- or string-based umbrella sampling, the bias strength $\kappa_\alpha^{\|} = 5$, and the choice of $\kappa_\alpha^{\perp}$ is irrelevant since there is no perpendicular direction in 1D. The transition path used as input for the path-based umbrella sampling is obtained by running the FTS method up to 100 iterations. Note that the FTS method is also performed with the same number of replicas, but with dynamics described by Eqs. (4.3.13)-(4.3.14). In all algorithms, the friction coefficient $\gamma = 1$, and step size $\Delta t = 0.005$. The size of α-th batch at every iteration is set to $|\mathcal{M}_k^\alpha| = 16$ and $|\mathcal{R}_k^\alpha| = 16$ for methods employing umbrella sampling and the FTS method, respectively. Each sample $\mathbf{x} \in \mathcal{M}_k^\alpha$ and $\mathbf{x} \in \mathcal{R}_k^\alpha$ is collected every 25 timesteps.

For the supervised learning component, the penalty strength $\lambda_{\text{SL}} = 100$ at all iterations, and empirical committor values are collected at every 40 iterations of the algorithm, i.e., $\tau_{\text{emp}} = 40$. The initial and final iteration index are set to $k_{\text{emp},s} = 10$ and $k_{\text{emp},f} = 2500$, respectively. The number of trajectories for each replica is $H = 100$. The size of α-th mini-batch is $|\mathcal{C}_k^\alpha| = 0.5|\mathcal{C}^\alpha|$, and thus the size of mini-batch during iterations grows as more samples are stored into $\mathcal{C}^\alpha$

For the boundary conditions, the penalty strengths are $\lambda_{\text{A}} = \lambda_{\text{B}} = 10^4$. The reactant and product batches $\mathcal{A}$ and $\mathcal{B}$ are collected prior to the start of each algorithm using dynamics given by Eqs. (4.3.13)-(4.3.14), but with R_α replaced with A and B, respectively. The size $|\mathcal{A}| = |\mathcal{B}| = 250M$ and each sample is also collected every 100 timesteps. During optimization, the mini-batch is randomly sampled without replacement from the original batch $\mathcal{A}$ and $\mathcal{B}$, where the size $|\mathcal{A}_k| = |\mathcal{B}_k| = 125M$.

The chosen optimizer to train the neural network is the Heavy-Ball method [213], which takes in two hyper-parameters as inputs. The first is the step size/learning rate η, while the second is the momentum coefficient μ. Given any function $f(\boldsymbol{\theta})$ to minimize, the Heavy-Ball method updates model parameters $\boldsymbol{\theta}_k$ with the following equation:

$$\mathbf{m}_{k+1} = \mu \mathbf{m}_k + \nabla_{\boldsymbol{\theta}} f(\boldsymbol{\theta}_k)\,, \tag{B.2.1}$$
$$\boldsymbol{\theta}_{k+1} = \boldsymbol{\theta}_k - \eta \mathbf{m}_{k+1}\,, \tag{B.2.2}$$

[2] https://github.com/muhammadhasyim/tps-torch

where $\mathbf{m}_0 = \mathbf{0}$. Note that our notation is consistent with PyTorch's implementation of the Heavy-Ball method. For all methods, $\eta = 5 \cdot 10^{-4}$ and $\mu = 0.95$. The gradient $\nabla_{\boldsymbol{\theta}} f(\boldsymbol{\theta})$ in Eq. (B.2.1) corresponds to, e.g., Eq. (5.3.2) for the BKE–US and BKE–FTS(US) method, and Eq. (5.5.2) for the BKE–FTS(ME) method, with additional mini-batches used as inputs to the gradient computation.

For the FTS method, the penalty strength is set to $\lambda_\mathrm{S} = 0.1M$, where $M = 20$ is the number of replicas. In addition, we replace the SGD step in Eq. (4.3.18) with a momentum-variant called the Nesterov's method [214]. As implemented in PyTorch, which follows the simplified version in [255], the Nesterov's update can be written as

$$\mathbf{m}_{k+1} = \mu^2 \mathbf{m}_k + (1+\mu)\nabla_{\boldsymbol{\varphi}^\alpha} \hat{C}(\{\boldsymbol{\varphi}_k^\alpha\})\,, \tag{B.2.3}$$

$$\boldsymbol{\varphi}_\star^\alpha = \boldsymbol{\varphi}_k^\alpha - \Delta\tau \mathbf{m}_{k+1}\,, \tag{B.2.4}$$

where $\mathbf{m}_0 = \mathbf{0}$. We set the step size/learning rate $\Delta\tau = 10^{-2}$ and momentum coefficient $\mu = 0.9$.

B.2.2 Second Study: Muller-Brown Potential

In the second study, umbrella sampling is performed with $M = 24$ replicas with dynamics given by Metropolis Monte Carlo [39, 256]. The particle is displaced in both directions by a random value between $-\Delta r$ and Δr to yield a new position $\mathbf{x}'$, which is accepted with probability given by $P_\mathrm{acc} = \min\left[1, \exp\left(-\beta(V_\mathrm{MB}(\mathbf{x}') - V_\mathrm{MB}(\mathbf{x}))\right)\right]$. The value of Δr is 0.05 when generating the batches $\mathcal{M}_k^\alpha$ for umbrella sampling, $\mathcal{R}_k^\alpha$ for the FTS method, and $\mathcal{C}_k^\alpha$ for supervised learning, while it is set to 0.01 for sampling the reactant and product states. For committor-based umbrella sampling, q_α is set to be $\frac{\alpha-1}{M-1}$ and $\kappa_\alpha = 10000$ for all α. For the path- or string-based umbrella sampling, we choose bias strength $\kappa_\alpha^{\parallel} = 1100$ and $\kappa_\alpha^{\perp} = 600$ and the transition path used as input is obtained by running the FTS method up to 100 iterations. Note the FTS method also uses the same amount of replicas as umbrella sampling, and the Monte Carlo method to sample configurations inside the Voronoi cells. For Figs. 5.9 and 5.10, the size of α-th batch at every iteration is set to $|\mathcal{M}_k^\alpha| = 16$ and $|\mathcal{R}_k^\alpha| = 4$ for methods employing umbrella sampling and the FTS method, respectively. For Figs. 5.11–B.12, we use a list of batch sizes $[4, 16, 64, 256, 1024]$, where each sample $\mathbf{x} \in \mathcal{M}_k^\alpha$ and $\mathbf{x} \in \mathcal{R}_k^\alpha$ is collected every 25 timesteps.

For the supervised learning component, the penalty strength is set to $\lambda_\mathrm{SL} = 100$ initially. Beginning at iteration 300, λ_SL is increased linearly to 25000 at iteration 10000. Empirical committor values are collected at every 10 iterations of the algorithm, i.e., $\tau_\mathrm{emp} = 10$. The initial and final iteration index where we start and end supervised learning is set to $k_{\mathrm{emp},s} = 10$ and $k_{\mathrm{emp},f} = 1000$. The number of trials for every window $H = 100$. The size of α-th mini-batch is $|\mathcal{C}_k^\alpha| = 0.5|\mathcal{C}^\alpha|$, and thus the size of mini-batch we use during iterations again grows as more samples are stored into $\mathcal{C}^\alpha$ as in the 1D case.

For the boundary conditions, the penalty strengths $\lambda_\mathrm{A} = \lambda_\mathrm{B} = 10^4$. The reactant and product batches $\mathcal{A}$ and $\mathcal{B}$ are collected before the start of each algorithm with dynamics confined to regions A and B, respectively. The size of the number of samples is $|\mathcal{A}| = |\mathcal{B}| = 100M$ and each sample is also collected every 10 timesteps. The mini-batch is randomly sampled without replacement from the original batch $\mathcal{A}$ and $\mathcal{B}$ with $|\mathcal{A}_k| = |\mathcal{B}_k| = 50M$.

The optimizer used to train the neural network in MB systems is Adam [226], which takes in four hyper-parameters as inputs: the first is the step size/learning rate η, the second and third are momentum coefficients β_1 and β_2 that control the change in the momentum and momentum squared respectively, and the fourth parameter ϵ is a term added to improve numerical stability. For any function $f(\boldsymbol{\theta})$ being optimized, the Adam update of model parameters $\boldsymbol{\theta}_k$ can be written as

$$\mathbf{m}_{k+1} = \beta_1 \mathbf{m}_k + (1-\beta_1)\nabla_{\boldsymbol{\theta}} f(\boldsymbol{\theta}_k)\,, \tag{B.2.5}$$

$$\mathbf{v}_{k+1} = \beta_2 \mathbf{v}_k + (1-\beta_2)\left[\nabla_{\boldsymbol{\theta}} f(\boldsymbol{\theta}_k) \odot \nabla_{\boldsymbol{\theta}} f(\boldsymbol{\theta}_k)\right]\,, \tag{B.2.6}$$

$$\hat{\mathbf{m}}_{k+1} = \frac{\mathbf{m}_k}{1-(\beta_1)^k}\,, \tag{B.2.7}$$

$$\hat{\mathbf{v}}_k = \frac{\mathbf{v}_k}{1-(\beta_2)^k}\,, \tag{B.2.8}$$

$$\mathbf{H}_{k+1} = \operatorname{diag}\left[\sqrt{\hat{\mathbf{v}}_k}\right] + \epsilon\,, \tag{B.2.9}$$

$$\boldsymbol{\theta}_{k+1} = \boldsymbol{\theta}_k - \eta(\mathbf{H}_{k+1})^{-1}\hat{\mathbf{m}}_{k+1}\,, \tag{B.2.10}$$

where $\odot$ is the element-wise product between two vectors that yields a new vector of the same dimension, the square root in Eq. (B.2.9) is applied element-wise to the vector, diag $[\dots]$ is a diagonal matrix obtained from elements of an input vector. Initially $\mathbf{m}_0 = \mathbf{0}$ and $\mathbf{v}_0 = \mathbf{0}$. Note that our notation is consistent with PyTorch's implementation of Adam, and for all methods, $\eta = 1 \cdot 10^{-3}$, $\beta_1 = 0.9$, $\beta_2 = 0.999$, and $\epsilon = 10^{-8}$.

For the FTS method, the penalty strength for the Müller-Brown potential is set to $\lambda_S = 0.1M$. The previously mentioned Nesterov's update scheme is also used here, with the step size/learning rate $\Delta\tau = 0.05$ and momentum coefficient $\mu = 0.9$.

B.2.3 Third Study: Solvated Dimer

In the third study, umbrella sampling is performed with $M = 24$ replicas with dynamics described by the overdamped Langevin dynamics in Eq. (5.3.1). For committor-based umbrella sampling, bias potential parameters for each replica are set to $\kappa_\alpha = 100$ for $\rho = 0.05$, 0.4, and 0.7 and $\kappa_\alpha = 50$ for $\rho = 0.9$, and $q_\alpha = \frac{\alpha-1}{M-1}$ for all densities. For the path- or string-based umbrella sampling, the bias strength $\kappa_\alpha^{\parallel} = \kappa_\alpha^{\perp} = 1200$. The transition path used as input for the path-based umbrella sampling is obtained by running the FTS method to 20000 iterations. Note that the FTS method is also performed with the same number of replicas, but with dynamics described by Eqs. (4.3.13)-(4.3.14). In all algorithms, the friction coefficient $\gamma = 1$, and step size $\Delta t = 0.0001$. The size of α-th batch at every iteration is set to $|\mathcal{M}_k^\alpha| = 8$ and $|\mathcal{R}_k^\alpha| = 8$ for methods employing umbrella sampling and the FTS method, respectively. Each sample $\mathbf{x} \in \mathcal{M}_k^\alpha$ and $\mathbf{x} \in \mathcal{R}_k^\alpha$ is collected every 25 timesteps.

For the supervised learning component, the penalty strength is set to $\lambda_{\mathrm{SL}} = 100$ initially. Beginning at iteration 200, λ_{SL} is increased linearly to 1000 at iteration 10000. Empirical committor values are collected at every 10 iterations of the algorithm, i.e., $\tau_{\mathrm{emp}} = 10$. The initial and final iteration index where we start and end supervised learning is set to $k_{\mathrm{emp},s} = 10$ and $k_{\mathrm{emp},f} = 5000$. The number of trials for every window $H = 100$. The size of α-th mini-batch is $|\mathcal{C}_k^\alpha| = 0.5|\mathcal{C}^\alpha|$, and thus the size of mini-batch we use during iterations again grows as more samples are stored into $\mathcal{C}^\alpha$ as in the 1D and 2D cases.

For the boundary conditions, the penalty strengths $\lambda_{\mathrm{A}} = \lambda_{\mathrm{B}} = 10^4$. The reactant and product batches $\mathcal{A}$ and $\mathcal{B}$ are collected before the start of each algorithm with dynamics confined to regions A and B, respectively. The size of the number of samples is $|\mathcal{A}| = |\mathcal{B}| = 100M$ and each sample is also collected every 10 timesteps. The mini-batch is randomly sampled without replacement from the original batch $\mathcal{A}$ and $\mathcal{B}$ with $|\mathcal{A}_k| = |\mathcal{B}_k| = 50M$.

The optimizer used to train the neural network in dimer systems is Adam as previously described in Appendix B.2.2. Initially $\mathbf{m}_0 = \mathbf{0}$ and $\mathbf{v}_0 = \mathbf{0}$. For all densities and methods, $\eta = 1 \cdot 10^{-5}$, $\beta_1 = 0.9$, $\beta_2 = 0.999$, and $\epsilon = 10^{-8}$.

For the FTS method, the penalty strength for the Müller-Brown potential is set to $\lambda_{\mathrm{S}} = 0.1M$. The previously mentioned Nesterov's update scheme is also used here, with the step size/learning rate $\Delta\tau = 0.001$ and momentum coefficient $\mu = 0.9$.

B.3 Comments on the Supervised Learning Loss Function

In this section, we compare the results from the supervised learning scheme used in this work with that of the more standard scheme seen in the literature [232], which utilizes the mean-squared error (MSE) loss function given by Eq. (5.4.3) instead of the supervised-learning loss given by Eq. (5.4.5). Switching the supervised-learning loss yields new algorithms denoted as the BKE–US+MSE, BKE–FTS(ME)+MSE, and the BKE–FTS(US)+MSE methods. The procedure for training the neural network follows that described in Appendix B.2.2, except for the BKE–US+MSE method where λ_{MSE} is increased linearly to 2500.

The results demonstrate that the use of the MSE loss function yields worse accuracy, as shown in the isocommittor lines and L_1-norm error in Fig. B.1. In fact, the L_1-norm error of all methods employing the MSE loss function increases at later iterations. Furthermore, with the exception of the BKE–FTS(ME)+MSE method, both the on-the-fly estimates (Fig. B.2(a)) and ensemble-averaged BKE loss function (Fig. B.2(b)) increase at higher iterations. This suggests that the MSE loss function is prone to overfitting [232], and we provide a sketch for why this occurs. To this end, the gradients of the losses with respect to the neural network

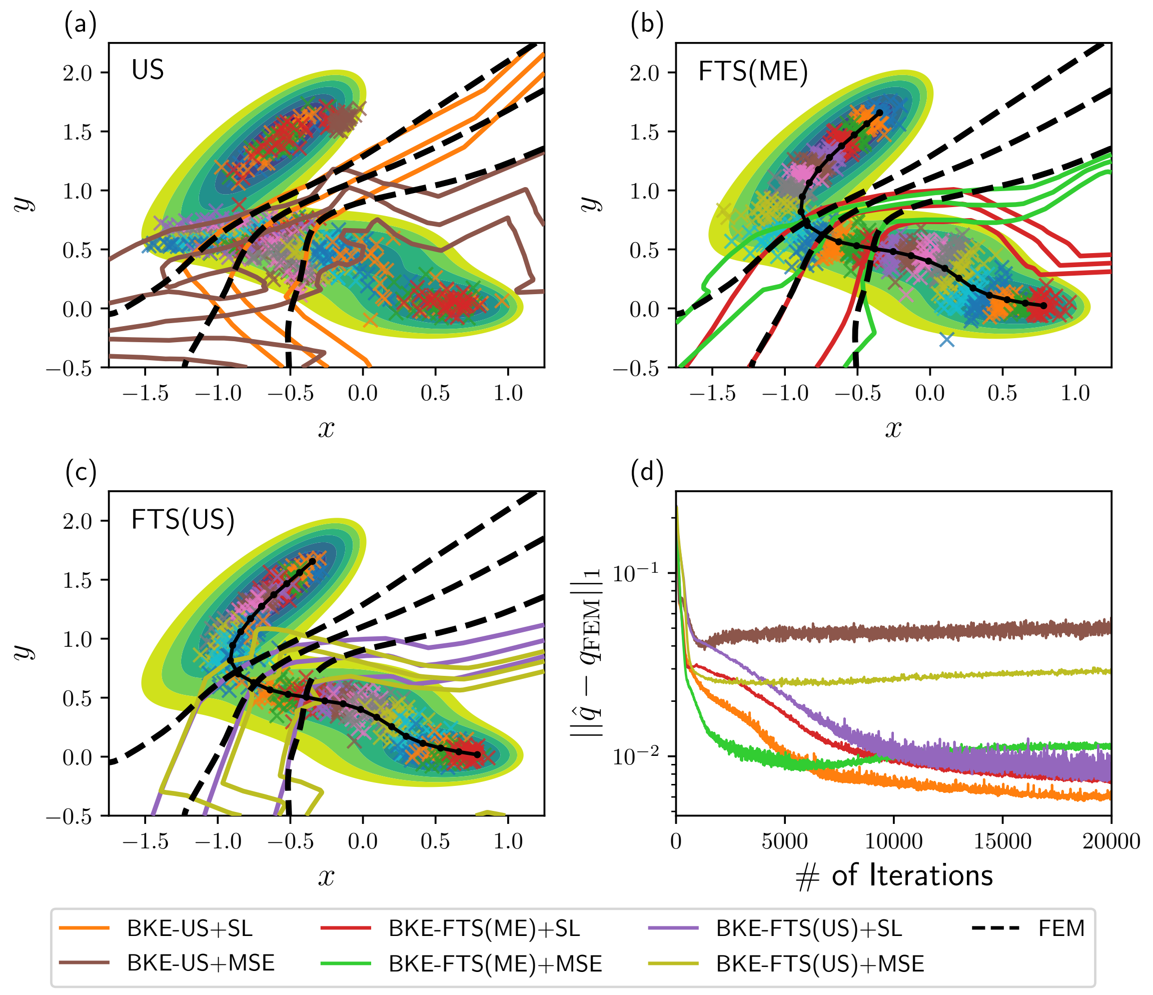


Figure B.1: Isocommittor lines for $q = 0.1$, 0.5, and 0.9 from (a) the BKE–US+SL and BKE–US+MSE method, (b) the BKE–FTS(ME)+SL and BKE–FTS(ME)+SL method, (c) the BKE–FTS(US) and BKE–FTS(US)+MSE method. $\times$ markers denote representative samples obtained from algorithms the SL methods. (d) The L_1-norm error as a function of iterations.

parameters are evaluated. For the MSE loss function in Eq. (5.4.3) this is

$$\nabla_{\boldsymbol{\theta}} \hat{L}_{\mathrm{MSE}}(\boldsymbol{\theta}; \{\mathcal{C}_k^\alpha\}) = \frac{\lambda_{\mathrm{MSE}}}{M} \sum_{\alpha=1}^{M} \frac{1}{|\mathcal{C}_k^\alpha|} \sum_{(q_{\mathrm{emp}}, \mathbf{x}) \in \mathcal{C}_k^\alpha} \nabla_{\boldsymbol{\theta}} \ell_{\mathrm{MSE}}(q_{\mathrm{emp}}, \mathbf{x}; \boldsymbol{\theta}) \tag{B.3.1}$$

$$= \frac{\lambda_{\mathrm{MSE}}}{M} \sum_{\alpha=1}^{M} \frac{1}{|\mathcal{C}_k^\alpha|} \sum_{(q_{\mathrm{emp}}, \mathbf{x}) \in \mathcal{C}_k^\alpha} \left(\hat{q}(\mathbf{x}; \boldsymbol{\theta}) - q_{\mathrm{emp}}\right) \nabla_{\boldsymbol{\theta}} \hat{q}(\mathbf{x}; \boldsymbol{\theta})\,. \tag{B.3.2}$$

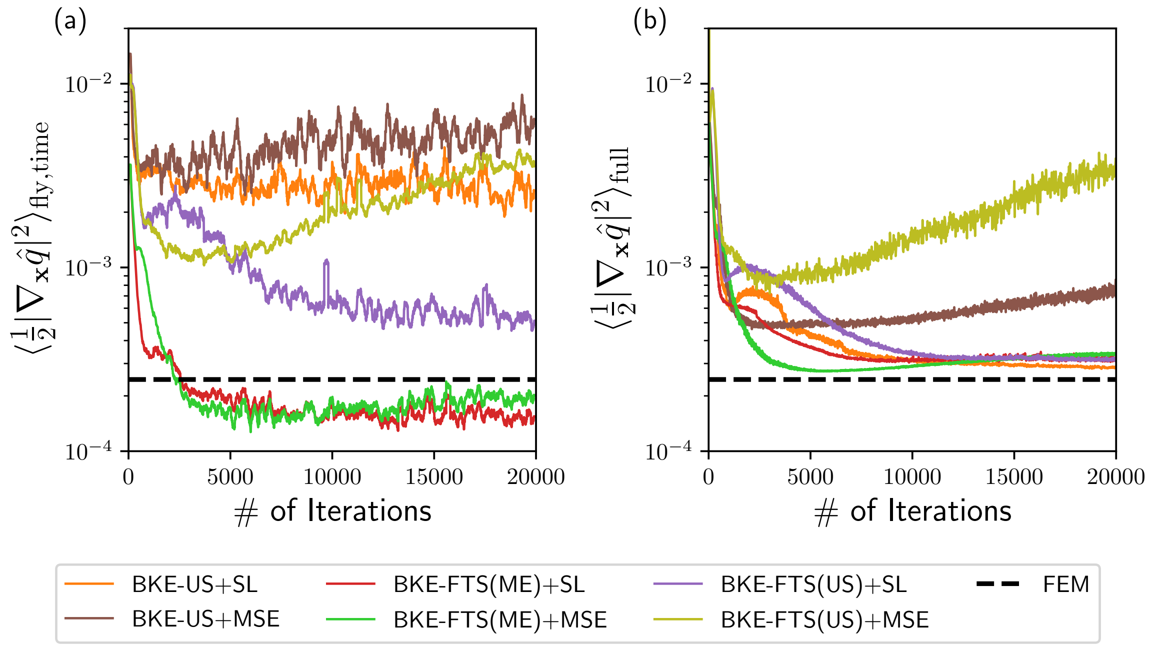


Figure B.2: (a) The filtered on-the-fly estimate of the BKE loss obtained at every iteration, with the filtering window set to 200 iterations. (b) The ensemble-averaged loss per Eq. (5.6.9) obtained at every iteration.

Note that the error $\hat{q}(\mathbf{x};\boldsymbol{\theta}) - q_{\mathrm{emp}}$ for every $\mathbf{x}$ is correlated point-wise with the model's gradient $\nabla_{\boldsymbol{\theta}}\hat{q}(\mathbf{x};\boldsymbol{\theta})$, which causes large point-wise errors to have more weight in the gradient descent direction. If the global minimum is reached, this results in fitting every datapoint in $\mathbf{x}$ perfectly, despite the statistical noise in the data. In comparison the gradient of the supervised-learning loss Eq. (5.4.5) is

$$\nabla_{\boldsymbol{\theta}}\hat{L}_{\mathrm{SL}}(\boldsymbol{\theta};\{\mathcal{C}_k^\alpha\}) = \frac{\lambda_{\mathrm{SL}}}{M}\sum_{\alpha=1}^{M}\nabla_{\boldsymbol{\theta}}\ell_{\mathrm{ME}}(\mathcal{C}_k^\alpha;\boldsymbol{\theta}) \tag{B.3.3}$$

$$= \frac{\lambda_{\mathrm{SL}}}{M}\sum_{\alpha=1}^{M}\left[\frac{1}{|\mathcal{C}_k^\alpha|}\sum_{(q_{\mathrm{emp}},\mathbf{x})\in\mathcal{C}_k^\alpha}(\hat{q}(\mathbf{x};\boldsymbol{\theta}) - q_{\mathrm{emp}})\right]\left[\frac{1}{|\mathcal{C}_k^\alpha|}\sum_{(q_{\mathrm{emp}},\mathbf{x})\in\mathcal{C}_k^\alpha}\nabla_{\boldsymbol{\theta}}\hat{q}(\mathbf{x};\boldsymbol{\theta})\right], \tag{B.3.4}$$

where the error $\hat{q}(\mathbf{x};\boldsymbol{\theta}) - q_{\mathrm{emp}}$ and model gradient $\nabla_{\boldsymbol{\theta}}\hat{q}(\mathbf{x};\boldsymbol{\theta})$ are now individually averaged with respect to samples in $\mathcal{C}_k^\alpha$. This averaging is crucial as it reduces the statistical noise in the empirical committor function $q_{\mathrm{emp}}(\mathbf{x})$. To see this, we first write $q_{\mathrm{emp}}(\mathbf{x})$ in terms of the exact committor function $q(\mathbf{x})$ as

$$q_{\mathrm{emp}}(\mathbf{x}) = q(\mathbf{x}) + \epsilon(\mathbf{x}), \tag{B.3.5}$$

where $\epsilon(\mathbf{x})$ is some noise. It is expected that $\epsilon(\mathbf{x})$ has zero mean and some unknown variance related to the number of trajectories used in the estimate. In the limit of large batch sizes, we can approximate the average over samples with an ensemble average. We then have for a single replica α

$$\begin{aligned}\ell_{\mathrm{ME}}(\mathcal{C}^\alpha;\boldsymbol{\theta}) &= \frac{1}{2}\left[\frac{1}{|\mathcal{C}^\alpha|}\sum_{(q_{\mathrm{emp}},\mathbf{x})\in\mathcal{C}^\alpha}(\hat{q}(\mathbf{x};\boldsymbol{\theta})-q_{\mathrm{emp}})\right]^2 &\text{(B.3.6)}\\ &\approx \frac{1}{2}\left(\langle\hat{q}(\mathbf{x};\boldsymbol{\theta})-q_{\mathrm{emp}}(\mathbf{x})\rangle_\alpha\right)^2 &\text{(B.3.7)}\\ &\approx \frac{1}{2}\left(\langle\hat{q}(\mathbf{x};\boldsymbol{\theta})-q(\mathbf{x})\rangle_\alpha-\langle\epsilon(\mathbf{x})\rangle_\alpha\right)^2 &\text{(B.3.8)}\\ &\approx \frac{1}{2}\left(\langle\hat{q}(\mathbf{x};\boldsymbol{\theta})-q(\mathbf{x})\rangle_\alpha\right)^2\,, &\text{(B.3.9)}\end{aligned}$$

where $\langle ... \rangle_\alpha$ is the ensemble average with respect to replica α. Note that the noise has been approximately canceled due to the effective matching of negative and positive error terms. In practice, the locality of the replicas in both umbrella sampling and the FTS method likely ensures that $\epsilon(\mathbf{x})$ is slowly varying. This leads to the annihilation of noise at the level of summing over batches from every replica without the need for higher quality $q_{\mathrm{emp}}(\mathbf{x})$.

Returning to the gradient of the supervised learning loss function given in Eq. (B.3.4), we have for large mini-batch sizes

$$\nabla_{\boldsymbol{\theta}}\hat{L}_{\mathrm{SL}}(\boldsymbol{\theta};\{\mathcal{C}_k^\alpha\}) \approx \frac{\lambda_{\mathrm{SL}}}{M}\sum_{\alpha=1}^{M}\langle\hat{q}(\mathbf{x};\boldsymbol{\theta})-q(\mathbf{x})\rangle_\alpha\langle\nabla_{\boldsymbol{\theta}}\hat{q}(\mathbf{x};\boldsymbol{\theta})\rangle_\alpha\,, \qquad \text{(B.3.10)}$$

in which the replica average of the gradient is coupled to a noise-reduced measure of the error. A global minimum is achieved when

$$\langle\hat{q}(\mathbf{x};\boldsymbol{\theta})\rangle_\alpha = \langle q(\mathbf{x})\rangle_\alpha \quad \forall\alpha\,. \qquad \text{(B.3.11)}$$

While this condition can be satisfied for $\hat{q}(\mathbf{x};\boldsymbol{\theta}) \neq q(\mathbf{x})$ in the region sampled by replica α, the additional loss terms in Eq. (5.2.7) and continuity between replicas seem to prevent trivial solutions in practice.

In summary, compared to the standard mean-squared loss, the chosen supervised-learning loss function avoids overfitting. This is likely due to the polling of empirical committor estimates, which leads to a reduction in the effect of noise on the optimization.

B.4 Examining the Sampling Error in Reweighting Factors

In this section, we examine how sampling error in the reweighting factors z_α estimated from all algorithms is reduced in the limit of large batch sizes. For a given batch size, z_α is computed

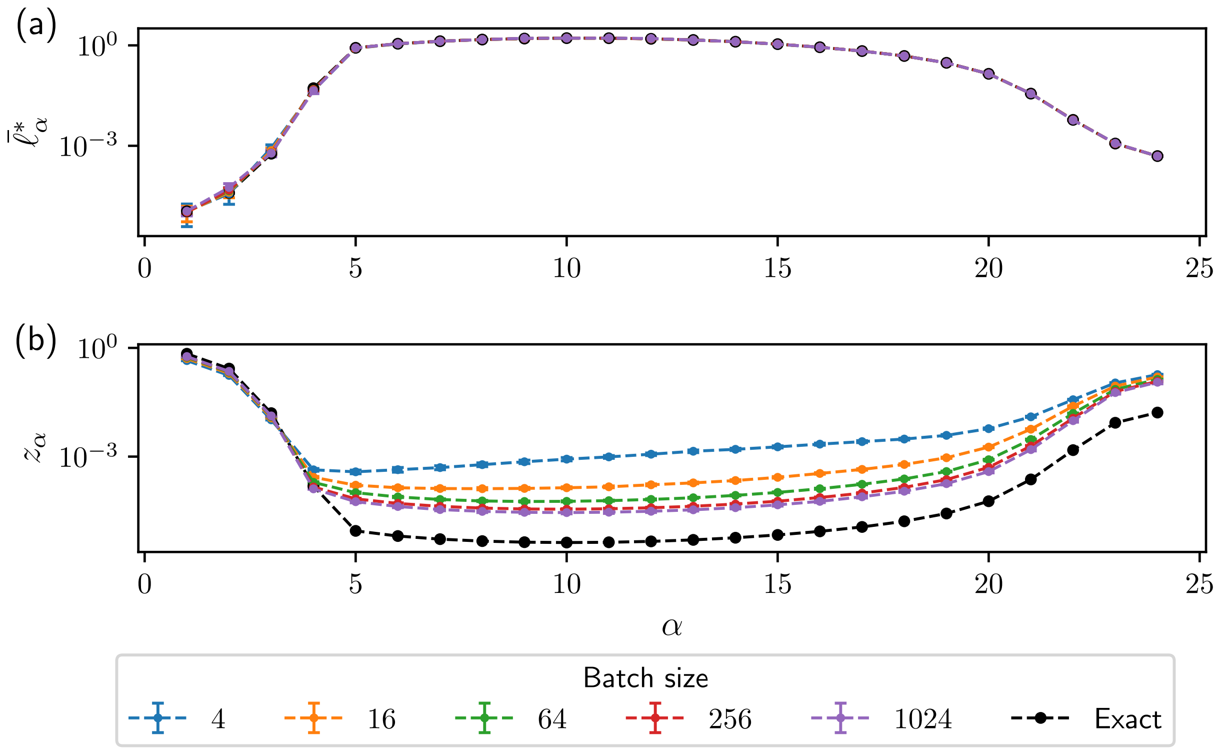


Figure B.3: (a) The sample mean of the BKE loss from each replica $\bar{\ell}^*_\alpha = \frac{1}{|\mathcal{M}^\alpha_k|}\sum_{\mathbf{x}\in\mathcal{M}^\alpha_k} \frac{\ell(\mathbf{x};\boldsymbol{\theta}_k)}{c(\mathbf{x};\boldsymbol{\theta}_k)}$, and (b) the estimated reweighting factor z_α from each replica α, in comparison to the values obtained by numerical integration ('Exact') for committor-based umbrella sampling. This is done using a fixed neural network for all batch sizes that is obtained from the BKE–US+SL method.

over many iterations of each algorithm while keeping the neural network fixed. Afterwards, the mean of z_α computed from all iterations is compared to the z_α computed from numerical integration of Eq. (5.3.3) for umbrella sampling, and Eq. (5.5.3) for the master-equation approach.

Figure B.3(b) shows z_α for committor-based umbrella sampling, where inaccurate estimates are obtained for $\alpha \in [5, 24]$. This result arises due to a lack of overlap in samples obtained from adjacent replicas since $\alpha = 5$ coincides with the beginning of non-overlap between samples from the reactant state (1-4) and the transition state, which begins at $\alpha = 5$. The inaccuracy in z_α can be contrasted with the sample-mean quantity $\bar{\ell}^*_\alpha = \frac{1}{|\mathcal{M}^\alpha_k|}\sum_{\mathbf{x}\in\mathcal{M}^\alpha_k} \frac{\ell(\mathbf{x};\boldsymbol{\theta}_k)}{c(\mathbf{x};\boldsymbol{\theta}_k)}$ (Fig. B.3(a)), which shows uniform convergence beginning with the smallest batch size. From these results, we may conclude that the large sampling error of the on-the-fly estimates from the BKE–US and BKE–US(SL) method arises from inaccurate reweighting factors due to the lack of overlap in samples between neighboring replicas, and the accuracy may only be improved with prohibitively large batch sizes for training.

Figure B.4 shows both z_α and the sample mean of the BKE loss from each replica $\bar{\ell}_\alpha = \frac{1}{|\mathcal{R}_k^\alpha|}\sum_{\mathbf{x}\in\mathcal{R}_k^\alpha} \ell(\mathbf{x};\boldsymbol{\theta}_k)$, as obtained from the FTS method with master equation. We see that the quantity $\bar{\ell}_\alpha$ converges quickly and uniformly, but the error in the reweighting factor z_α, which is the largest for $\alpha \in [11, 24]$, only diminishes at batch sizes that are too large and impractical to use for neural network training, i.e., at $O(4 \cdot 10^3)$. Meanwhile, z_α computed from path-based umbrella sampling (Fig. B.5(b)) achieves convergence at relatively smaller batch sizes, i.e., at $O(10^2)$, with similar quick convergence for the corresponding $\bar{\ell}^*_\alpha$ (Fig. B.5(a)). This demonstrates the advantage of using path-based umbrella sampling for computing accurate reweighting factors, and thus the utility of the BKE–FTS(US) and BKE–FTS(US)+SL method in obtaining accurate on-the-fly estimates of reaction rates at a wide range of batch sizes.

B.5 Additional Figures for Examining Log-Normal Behavior

This section contains additional figures for the probability density functions (PDFs) of all quantities of interest in Section 5.6.3 for all replicas. The histograms for the forward free-energy differences are given in Fig. B.6. The histograms for the backward free-energy differences are given in Fig. B.7. The histograms for the reweighting factors are given in Fig. B.8. The histograms for $\ln \bar{\ell}^*_\alpha$ and $\ln \bar{1}^*_\alpha$ are given in Figs. B.9 and B.10, respectively. The histograms for $\ln z_\alpha \bar{\ell}^*_\alpha$ and $\ln z_\alpha \bar{1}^*_\alpha$ are given in Figs. B.11 and B.12, respectively. For all histograms, data is obtained by sampling a fixed neural network obtained from the BKE–FTS(US)+SL method at a batch size of 1024. Dashed blue lines correspond to log-normal distributions fitted using the method of moments [236], while the vertical dotted orange and solid black lines correspond to the mean of the histograms and the corresponding ensemble average computed by numerical integration, respectively.

The existence of tails in these PDFs is dependent upon the choice of bias potential parameters that are needed for the path-based umbrella sampling. For instance, Figs. B.13 and B.14 show the histograms for $\ln z_\alpha \bar{\ell}^*_\alpha$ and $\ln z_\alpha \bar{1}^*_\alpha$ when the bias potential parameters are changed from the ones in Appendix B.2.2 to $\kappa_\alpha^\parallel = 2200$ and $\kappa_\alpha^\perp = 300$, where we see that PDFs that originally possess tails, e.g., $\alpha \in [14, 18]$ for $\ln z_\alpha \bar{\ell}^*_\alpha$, are log-normal. It is also expected that any tails in the distributions are suppressed as the batch size is further increased.

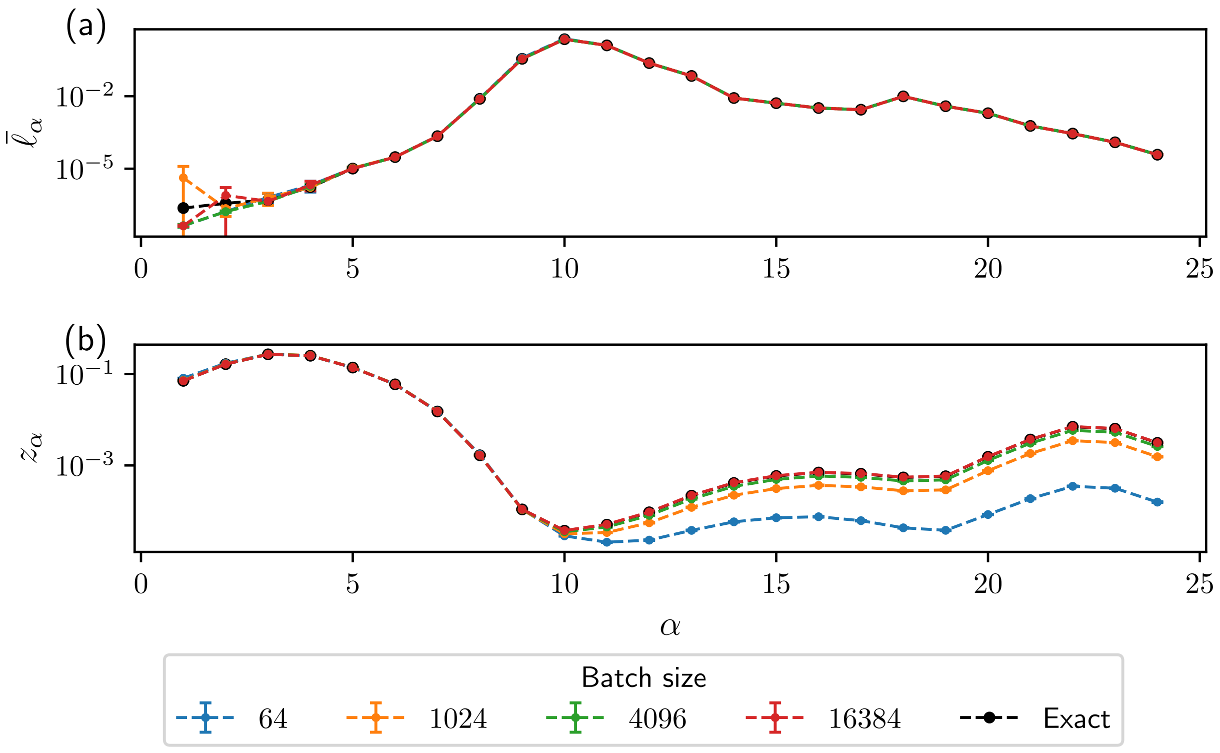


Figure B.4: (a) The sample mean of the BKE loss from each replica $\bar{\ell}_\alpha = \frac{1}{|\mathcal{R}_k^\alpha|} \sum_{\mathbf{x} \in \mathcal{R}_k^\alpha} \ell(\mathbf{x}; \boldsymbol{\theta}_k)$, and (b) the estimated reweighting factor z_α from each replica α, in comparison to the values obtained by numerical integration ('Exact') for the FTS method with master equation. This is done using a fixed neural network for all batch sizes that is obtained from the BKE–FTS(ME)+SL method.

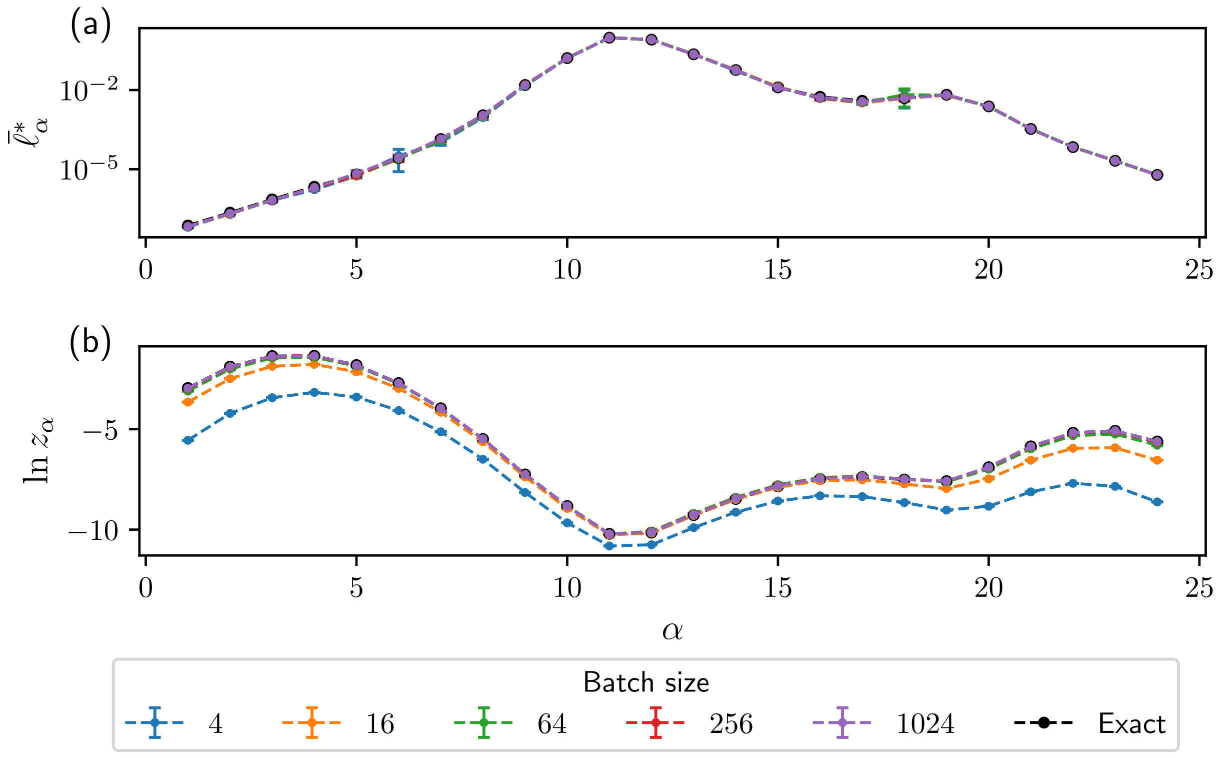


Figure B.5: (a) The sample mean of the BKE loss from each replica $\bar{\ell}^*_\alpha = \frac{1}{|\mathcal{M}^\alpha_k|}\sum_{\mathbf{x}\in\mathcal{M}^\alpha_k}\frac{\ell(\mathbf{x};\boldsymbol{\theta}_k)}{c(\mathbf{x};\boldsymbol{\theta}_k)}$, and (b) the estimated reweighting factor $\ln z_\alpha$ from each replica α, in comparison to the values obtained by numerical integration ('Exact') for the path-based umbrella sampling. This is done using a fixed neural network for all batch sizes that is obtained from the BKE–FTS(US)+SL method.

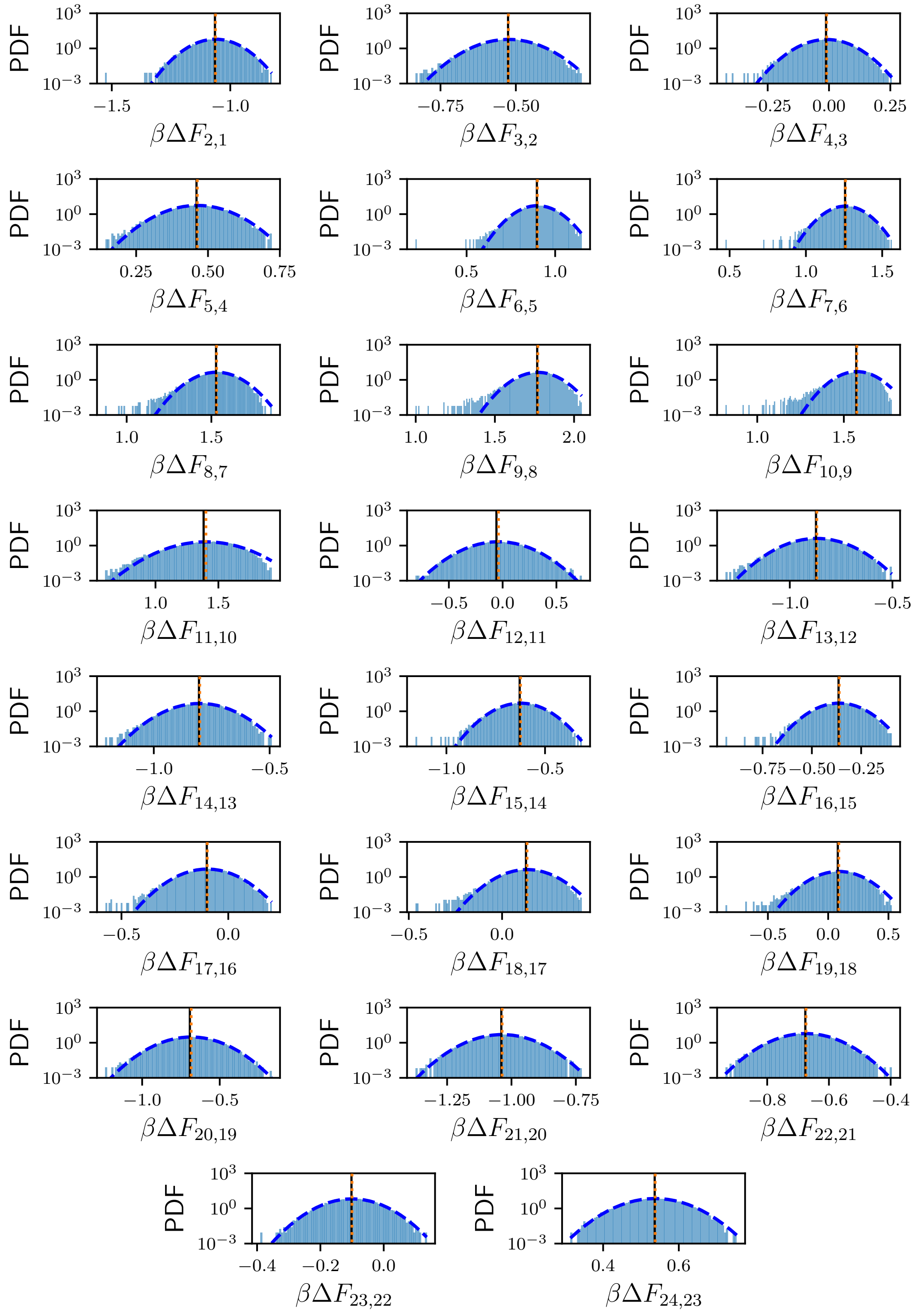


Figure B.6: Probability density functions of the forward free-energy differences $\beta\Delta F_{(\alpha+1),\alpha}$.

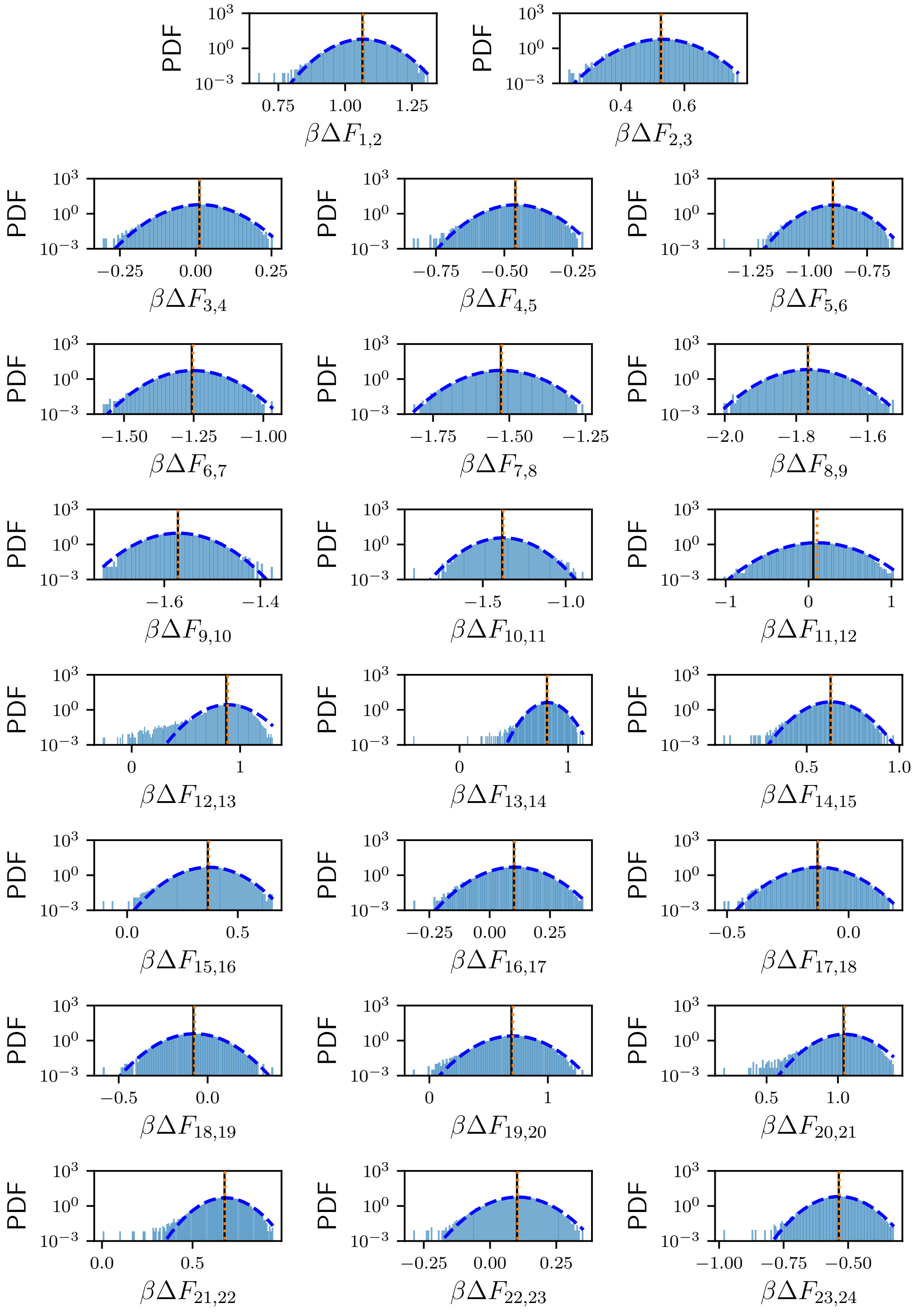


Figure B.7: Probability density functions of the backward free-energy differences $\beta\Delta F_{(\alpha-1),\alpha}$.

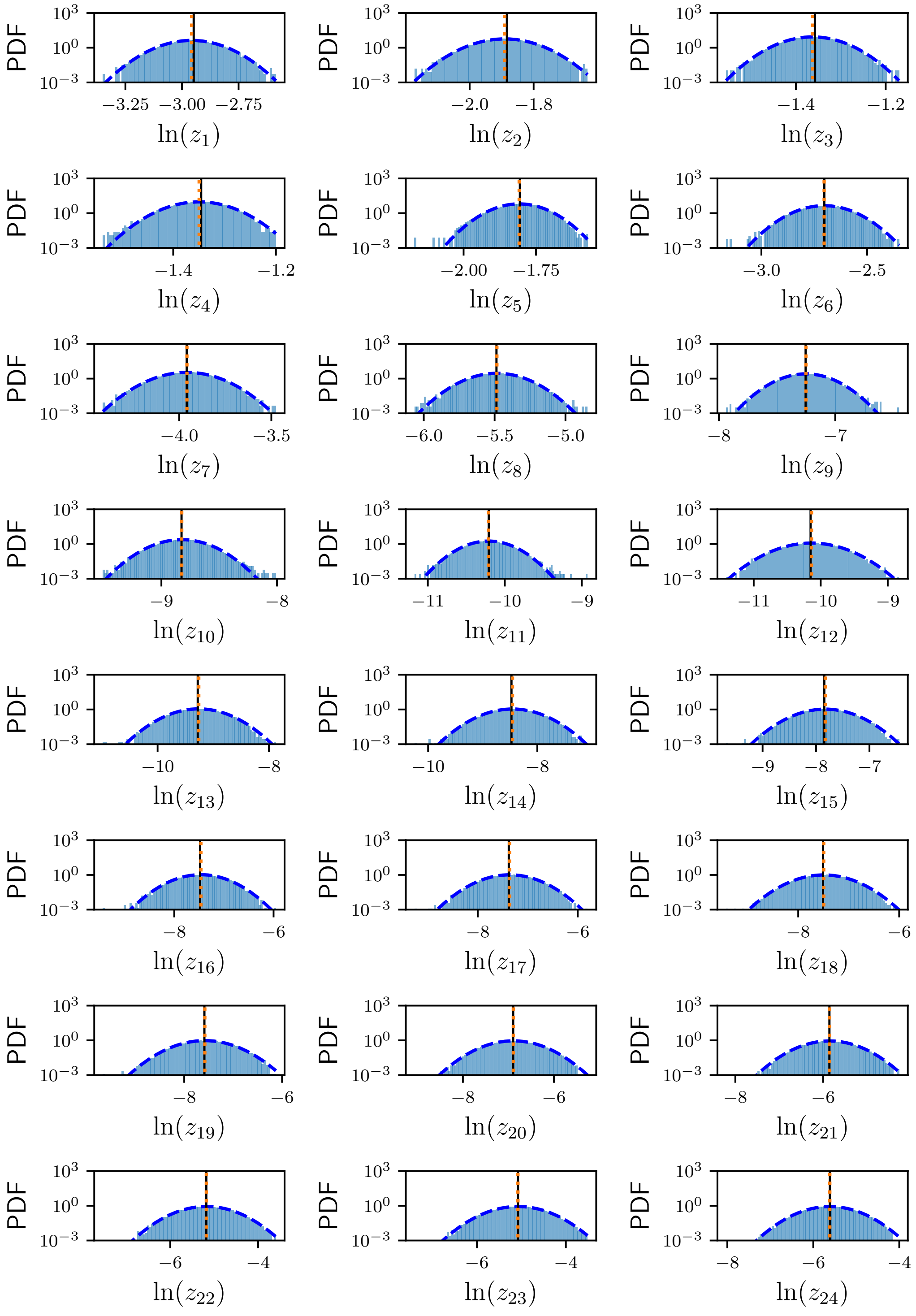


Figure B.8: Probability density functions of the log of reweighting factors $\ln z_\alpha$.

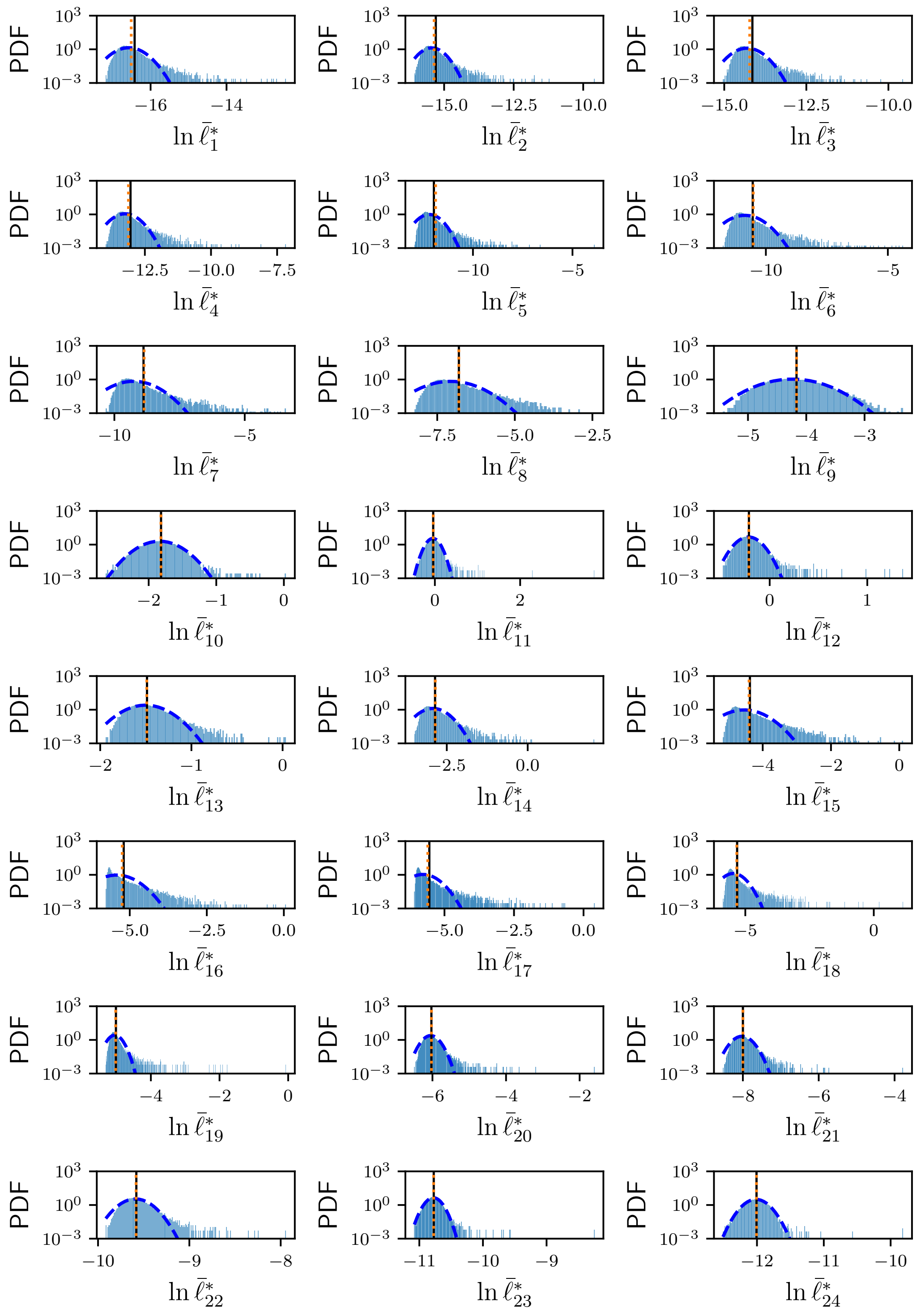


Figure B.9: Probability density functions of $\ln \bar{\ell}_\alpha^*$.

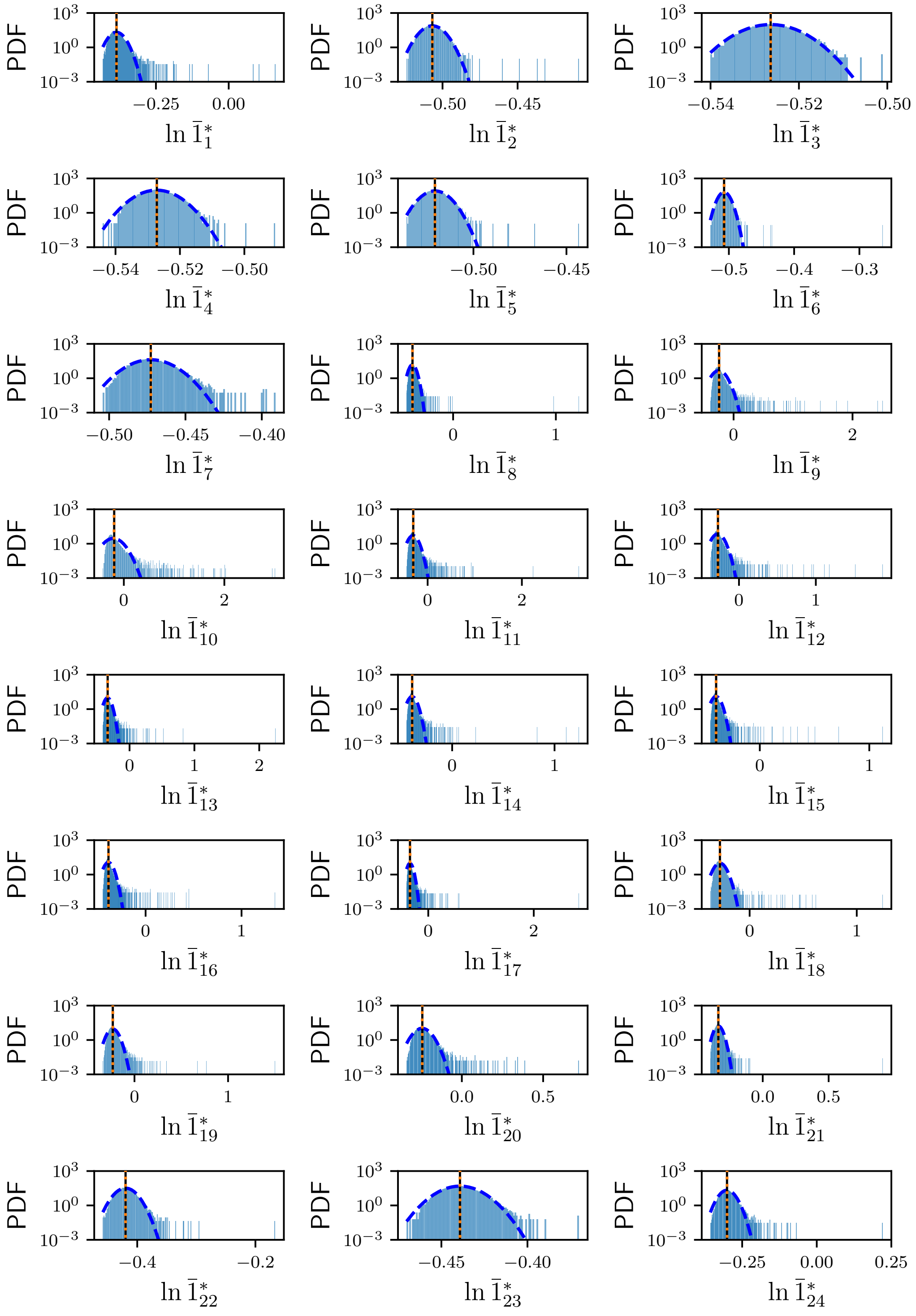


Figure B.10: Probability density functions of $\ln \bar{1}^*_\alpha$.

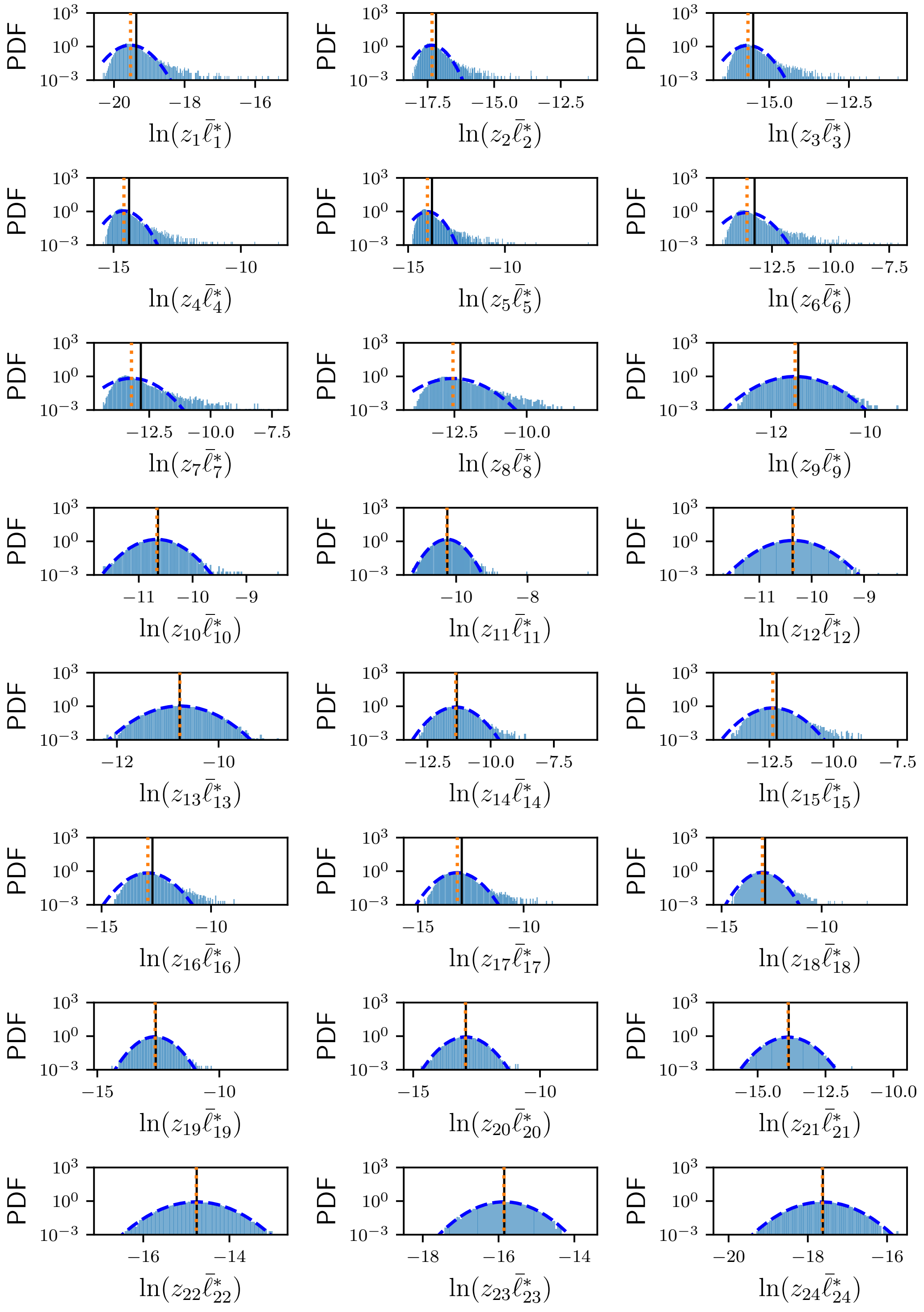


Figure B.11: Probability density functions of $\ln z_\alpha \bar{\ell}^*_\alpha$.

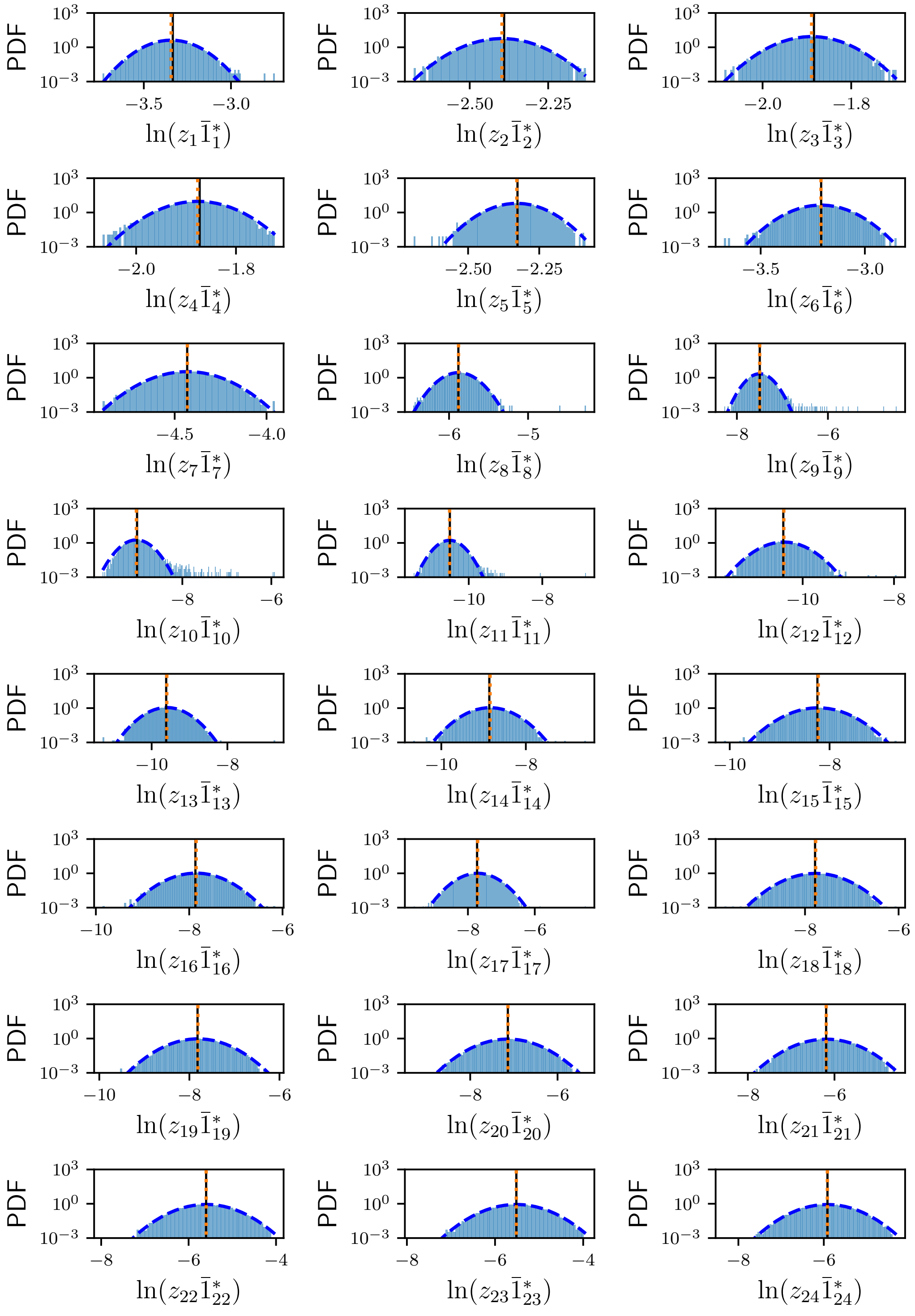


Figure B.12: Probability density functions of $\ln z_\alpha \bar{1}_\alpha^*$.

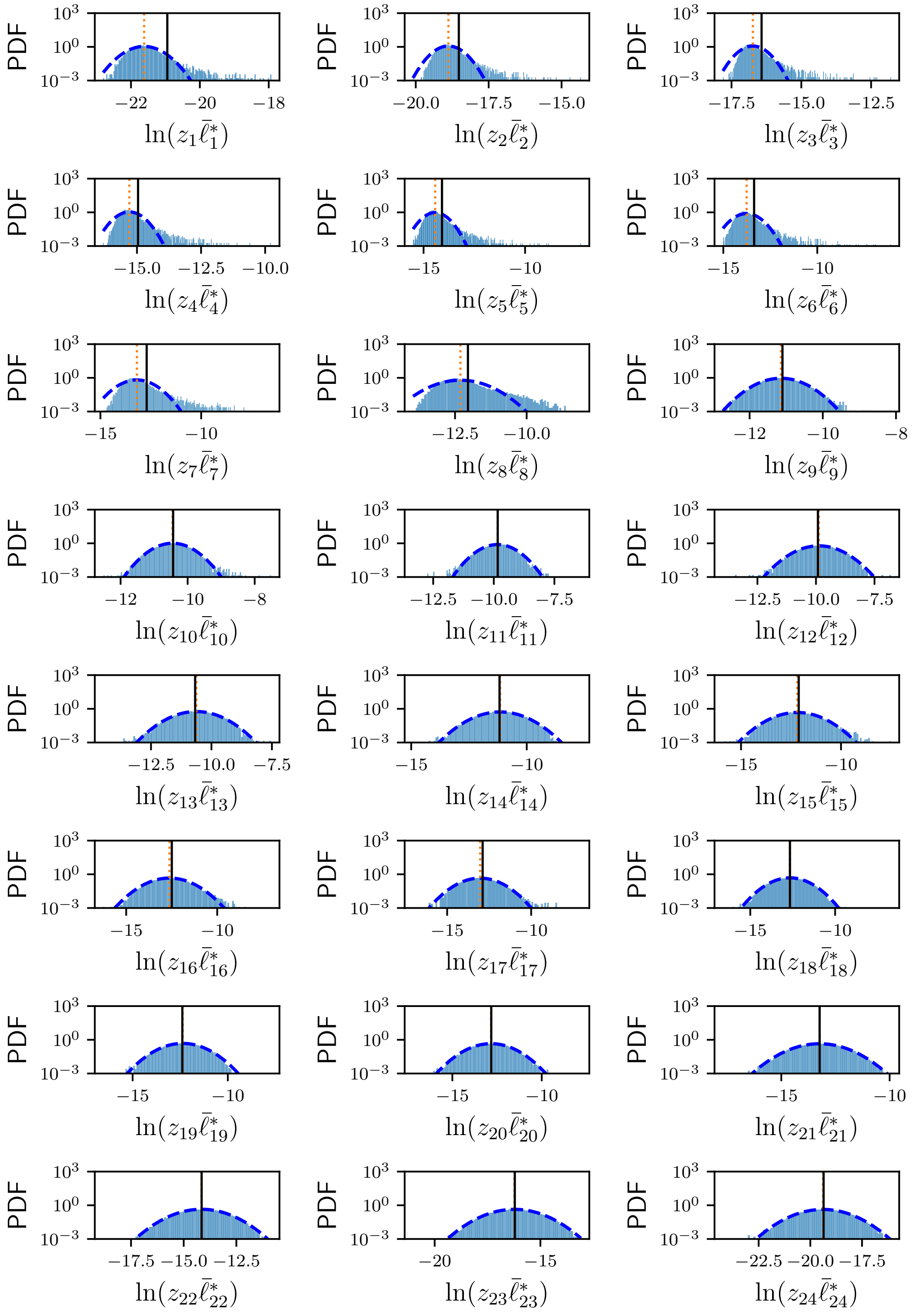


Figure B.13: Probability density functions of $\ln z_\alpha \bar{\ell}^*_\alpha$, where data is obtained from umbrella sampling with bias strengths $\kappa^{\parallel}_\alpha = 2200$ and $\kappa^{\perp}_\alpha = 300$.

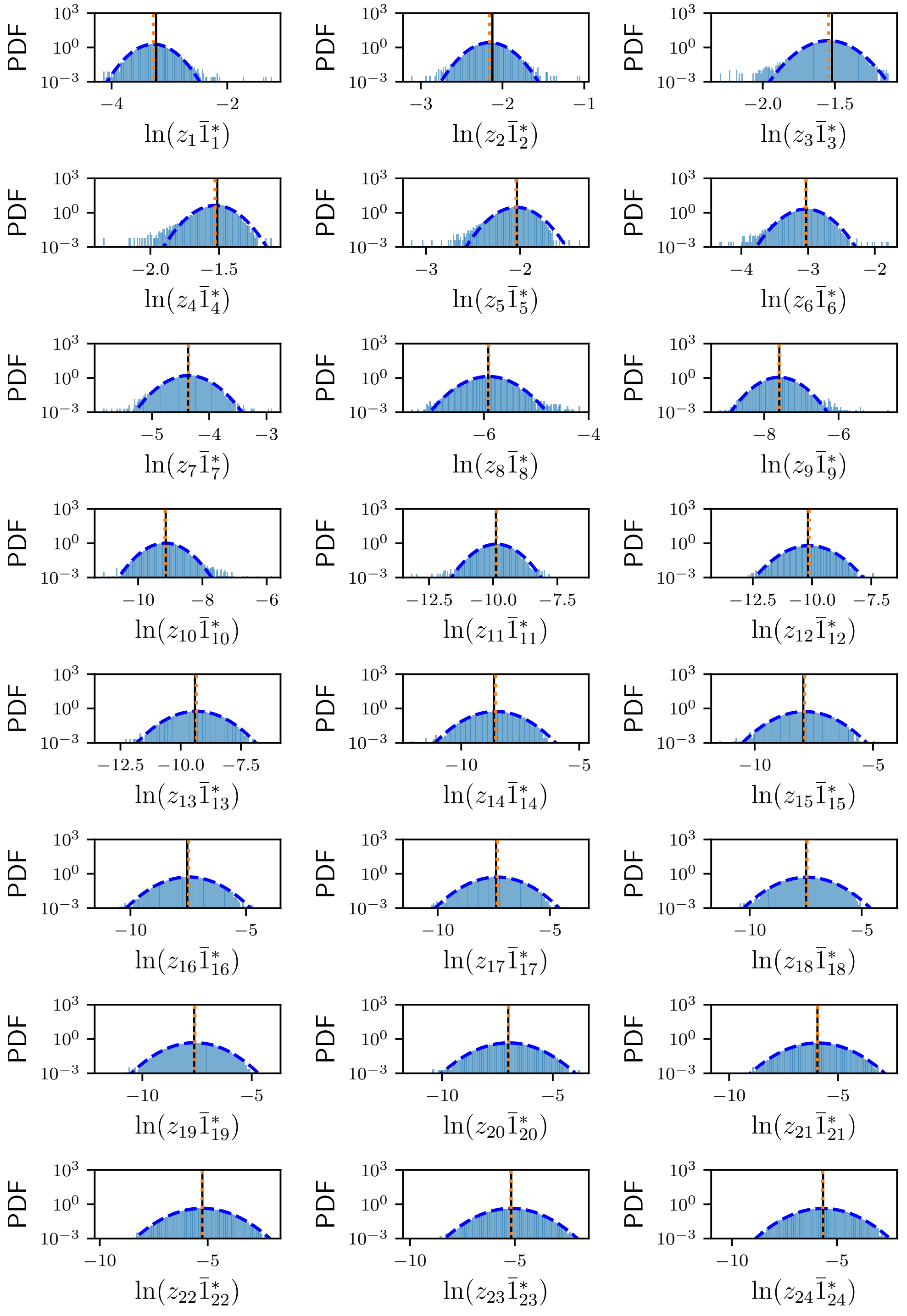


Figure B.14: Probability density functions of $\ln z_\alpha \bar{1}^*_\alpha$, where data is obtained from umbrella sampling with bias strengths $\kappa^{\parallel}_\alpha = 2200$ and $\kappa^{\perp}_\alpha = 300$.

Bibliography

[1]D. Chandler, *Introduction to Modern Statistical Mechanics* (Oxford University Press, 1987).

[2]J. Hénin, T. Lelièvre, M. R. Shirts, O. Valsson, and L. Delemotte, *Enhanced sampling methods for molecular dynamics simulations*, 2022, arXiv:2202.04164.

[3]D. N. Arnold, *A Concise Introduction to Numerical Analysis*, `https://www-users.cse.umn.edu/~arnold/597.00-01/nabook.pdf`, 2001.

[4]S. J. Singer and G. L. Nicolson, "The fluid mosaic model of the structure of cell membranes", Science **175**, 720–731 (1972).

[5]J. Yu, D. A. Fischman, and T. L. Steck, "Selective solubilization of proteins and phospholipids from red blood cell membranes by nonionic detergents", Journal of Supramolecular Structure **1**, 233–248 (1973).

[6]K. Simons and E. Ikonen, "Functional rafts in cell membranes", Nature **387**, 569–572 (1997).

[7]S. Katira, K. K. Mandadapu, S. Vaikuntanathan, B. Smit, and D. Chandler, "Pre-transition effects mediate forces of assembly between transmembrane proteins", eLife **5**, e13150 (2016).

[8]Y. Khoo, J. Lu, and L. Ying, "Solving for high-dimensional committor functions using artificial neural networks", Research in the Mathematical Sciences **6**, 1–13 (2019).

[9]Q. Li, B. Lin, and W. Ren, "Computing committor functions for the study of rare events using deep learning", The Journal of Chemical Physics **151**, 054112 (2019).

[10]G. M. Rotskoff, A. R. Mitchell, and E. Vanden-Eijnden, "Active importance sampling for variational objectives dominated by rare events: consequences for optimization and generalization", Proceedings of Machine Learning Research **145**, 757–780 (2022).

[11]K. Jacobson, E. D. Sheets, and R. Simson, "Revisiting the fluid mosaic model of membranes", Science **268**, 1441–1442 (1995).

[12]G. L. Nicolson and G. Ferreira de Mattos, "A brief introduction to some aspects of the fluid–mosaic model of cell membrane structure and its importance in membrane lipid replacement", Membranes **11**, `10.3390/membranes11120947` (2021).

[13]L. Lizana, B. Bauer, and O. Orwar, "Controlling the rates of biochemical reactions and signaling networks by shape and volume changes", Proceedings of the National Academy of Sciences **105**, 4099–4104 (2008).

[14]T. D. Perez, M. Tamada, M. P. Sheetz, and W. J. Nelson, "Immediate-early signaling induced by e-cadherin engagement and adhesion", Journal of Biological Chemistry **283**, 5014–5022 (2008).

[15]E. Ikonen, "Cellular cholesterol trafficking and compartmentalization", Nature Reviews Molecular Cell Biology **9**, 125–138 (2008).

[16]A. Callan-Jones and P. Bassereau, "Curvature-driven membrane lipid and protein distribution", Current Opinion in Solid State and Materials Science **17**, 143–150 (2013).

[17]L.-G. Wu, E. Hamid, W. Shin, and H.-C. Chiang, "Exocytosis and endocytosis: modes, functions, and coupling mechanisms", Annual Review of Physiology **76**, 301–331 (2014).

[18]S. C. Harrison, "Viral membrane fusion", Virology **479-480**, 60th Anniversary Issue, 498–507 (2015).

[19]C. L. Jackson, L. Walch, and J.-M. Verbavatz, "Lipids and their trafficking: an integral part of cellular organization", Developmental Cell **39**, 139–153 (2016).

[20]J. Huang and A. D. MacKerell Jr, "Charmm36 all-atom additive protein force field: validation based on comparison to nmr data", Journal of Computational Chemistry **34**, 2135–2145 (2013).

[21]J. P. M. Jämbeck and A. P. Lyubartsev, "Derivation and systematic validation of a refined all-atom force field for phosphatidylcholine lipids", The Journal of Physical Chemistry B **116**, PMID: 22352995, 3164–3179 (2012).

[22]C. J. Dickson, B. D. Madej, Å. A. Skjevik, R. M. Betz, K. Teigen, I. R. Gould, and R. C. Walker, "Lipid14: the amber lipid force field", Journal of Chemical Theory and Computation **10**, PMID: 24803855, 865–879 (2014).

[23]I. R. Cooke, K. Kremer, and M. Deserno, "Tunable generic model for fluid bilayer membranes", Physical Review E **72**, 011506 (2005).

[24]S. J. Marrink, H. J. Risselada, S. Yefimov, D. P. Tieleman, and A. H. de Vries, "The martini force field: coarse grained model for biomolecular simulations", The Journal of Physical Chemistry B **111**, PMID: 17569554, 7812–7824 (2007).

[25]J. Grime and J. J. Madsen, *Efficient simulation of tunable lipid assemblies across scales and resolutions*, 2019, arXiv:1910.05362.

[26]W. Helfrich, "Elastic properties of lipid bilayers: theory and possible experiments", Zeitschrift für Naturforschung C **28**, 693–703 (1973).

[27]U. Seifert, "Configurations of fluid membranes and vesicles", Advances in Physics **46**, 13–137 (1997).

[28]A. Sahu, R. A. Sauer, and K. K. Mandadapu, "Irreversible thermodynamics of curved lipid membranes", Phys. Rev. E **96**, 042409 (2017).

[29]N. Goldenfeld, *Lectures on Phase Transitions and the Renormalization Group* (CRC Press, 2019).

[30]R. Dimova, S. Aranda, N. Bezlyepkina, V. Nikolov, K. A. Riske, and R. Lipowsky, "A practical guide to giant vesicles. probing the membrane nanoregime via optical microscopy", Journal of Physics: Condensed Matter **18**, S1151–S1176 (2006).

[31]A. Stukowski, "Visualization and analysis of atomistic simulation data with OVITO-the Open Visualization Tool", Modelling and Simulatoin in Materials Science and Engineering **18**, `{10.1088/0965-0393/18/1/015012}` (2010).

[32]R. Phillips, "Membranes by the numbers", in *Physics of biological membranes*, edited by P. Bassereau and P. Sens (Springer International Publishing, Cham, 2018), pp. 73–105.

[33]H. Diamant, "Model-free thermodynamics of fluid vesicles", Phys. Rev. E **84**, 061123 (2011).

[34]K. Crane, *Discrete Differential Geometry: An Applied Introduction*, `https://www.cs.cmu.edu/~kmcrane/Projects/DDG/paper.pdf`, 2022.

[35]K. Tapp, *Differential Geometry of Curves and Surfaces* (Springer Cham, 2016).

[36]E. Kreyszig, *Differential Geometry* (Dover Publications, 1991).

[37]J.-S. Ho and A. Baumgärtner, "Simulations of fluid self-avoiding membranes", Europhysics Letters (EPL) **12**, 295–300 (1990).

[38]G. Gompper and D. M. Kroll, "Random surface discretizations and the renormalization of the bending rigidity", Journal de Physique I **6**, 1305–1320 (1996).

[39]D. Frenkel and B. Smit, *Understanding Molecular Simulation: From Algorithms to Applications*, 2nd ed. (Elsevier, 2001).

[40]M. Tuckerman, *Statistical Mechanics: Theory and Molecular Simulation* (Oxford University Press, 2010).

[41]M. P. Allen and D. J. Tildesley, *Computer Simulation of Liquids*, 2nd ed. (Oxford University Press, 2017).

[42]D. Boulatov, V. Kazakov, I. Kostov, and A. Migdal, "Analytical and numerical study of a model of dynamically triangulated random surfaces", Nuclear Physics B **275**, 641–686 (1986).

[43]A. Billoire and F. David, "Scaling properties of randomly triangulated planar random surfaces: a numerical study", Nuclear Physics B **275**, 617–640 (1986).

[44]A. Baumgärtner and J.-S. Ho, "Crumpling of fluid vesicles", Phys. Rev. A **41**, 5747–5750 (1990).

[45]D. H. Boal and M. Rao, "Topology changes in fluid membranes", Phys. Rev. A **46**, 3037–3045 (1992).

[46]W. W. Wood, "Monte carlo calculations for hard disks in the isothermal-isobaric ensemble", The Journal of Chemical Physics **48**, 415–434 (1968).

[47]I. McDonald, "Monte carlo calculations for one- and two-component fluids in the isothermal-isobaric ensemble", Chemical Physics Letters **3**, 241–243 (1969).

[48]M. S. Shell, *CHE210D: Principles of modern molecular simulation methods*, [Online; accessed 6-November-2017], 2012.

[49]J. A. Anderson, E. Jankowski, T. L. Grubb, M. Engel, and S. C. Glotzer, "Massively parallel monte carlo for many-particle simulations on gpus", Journal of Computational Physics **254**, 27–38 (2013).

[50]C. L. Phillips, J. A. Anderson, and S. C. Glotzer, "Pseudo-random number generation for brownian dynamics and dissipative particle dynamics simulations on gpu devices", Journal of Computational Physics **230**, 7191–7201 (2011).

[51]J. A. Anderson, J. Glaser, and S. C. Glotzer, "HOOMD-blue: A Python package for high-performance molecular dynamics and hard particle Monte Carlo simulations", Computational Materials Science **173**, 109363 (2020).

[52]Y. Kantor and D. R. Nelson, "Phase transitions in flexible polymeric surfaces", Phys. Rev. A **36**, 4020–4032 (1987).

[53]H. S. Seung and D. R. Nelson, "Defects in flexible membranes with crystalline order", Phys. Rev. A **38**, 1005–1018 (1988).

[54]J. Li, M. Dao, C. Lim, and S. Suresh, "Spectrin-level modeling of the cytoskeleton and optical tweezers stretching of the erythrocyte", Biophysical Journal **88**, 3707–3719 (2005).

[55]C. Itzykson and J.-M. Drouffe, *Statistical Field Theory: Strong Coupling, Monte Carlo methods, Conformal Field Theory, and Random Systems*, Vol. 2 (Cambridge University Press, 1991).

[56]M. Botsch, L. Kobbelt, M. Pauly, P. Alliez, and B. Lévy, *Polygon Mesh Processing* (CRC Press, 2010).

[57]P. Papadopoulos, *ME 280A: Introduction to the Finite Element Method*, `https://csml.berkeley.edu/Notes/ME280A.pdf`, 2015.

[58]H. P. Langtangen and K.-A. Mardal, *Introduction to Numerical Methods for Variational Problems*, Vol. 21 (Springer Nature, 2019).

[59]M. Meyer, M. Desbrun, P. Schröder, and A. H. Barr, "Discrete differential-geometry operators for triangulated 2-manifolds", in Visualization and mathematics iii, edited by H.-C. Hege and K. Polthier (2003), pp. 35–57.

[60]S. J. Colley, *Vector Calculus*, Fourth (Pearson, 2011).

[61]G. H. Golub and C. F. Van Loan, *Matrix Computations*, Fourth (JHU Press, 2013).

[62]O. C. Zienkiewicz, R. L. Taylor, and J. Z. Zhu, *The Finite Element Method: Its Basis and Fundamentals*, Seventh (Elsevier, 2005).

[63]M. de Berg, O. Cheong, M. van Kreveld, and M. Overmars, *Computational Geometry: Algorithms and Applications*, Third (Elsevier, 2008).

[64]E. Atilgan and S. X. Sun, "Shape transitions in lipid membranes and protein mediated vesicle fusion and fission", The Journal of Chemical Physics **126**, 03B604 (2007).

[65]F. Jülicher, "The morphology of vesicles of higher topological genus: conformal degeneracy and conformal modes", Journal de Physique II **6**, 1797–1824 (1996).

[66]C. Loop, "Smooth subdivision surfaces based on triangles", (1987).

[67]M. Deserno, *Fluid lipid membranes–a primer*, 2007.

[68]J. S. Rowlinson and B. Widom, *Molecular Theory of Capillarity* (Courier Corporation, 2013).

[69]M. Deserno, *Notes on Differential Geometry*, 2004.

[70]A. Zee, *Quantum Field Theory in a Nutshell*, 2nd ed., Vol. 7 (Princeton University Press, 2010).

[71]D. J. Smith, J. B. Klauda, and A. J. Sodt, "Simulation best practices for lipid membranes", Living Journal of Computational Molecular Science **1**, 5966 (2018).

[72]M. Frigo and S. G. Johnson, "The design and implementation of FFTW3", Proceedings of the IEEE **93**, Special issue on "Program Generation, Optimization, and Platform Adaptation", 216–231 (2005).

[73]H. Shiba and H. Noguchi, "Estimation of the bending rigidity and spontaneous curvature of fluid membranes in simulations", Phys. Rev. E **84**, 031926 (2011).

[74]E. G. Brandt, A. R. Braun, J. N. Sachs, J. F. Nagle, and O. Edholm, "Interpretation of fluctuation spectra in lipid bilayer simulations", Biophysical Journal **100**, 2104–2111 (2011).

[75]W. Cai, T. C. Lubensky, P. Nelson, and T. Powers, "Measure factors, tension, and correlations of fluid membranes", Journal de Physique II **4**, 931–949 (1994).

[76]F. Schmid, "Are stress-free membranes really "tensionless"?", EPL (Europhysics Letters) **95**, 28008 (2011).

[77]O. Farago, "Mechanical surface tension governs membrane thermal fluctuations", Physical Review E **84**, 051914 (2011).

[78]M. Durand, "Frame tension governs the thermal fluctuations of a fluid membrane: new evidence", Soft Matter **18**, 3891–3901 (2022).

[79]H. Shiba, H. Noguchi, and J.-B. Fournier, "Monte carlo study of the frame, fluctuation and internal tensions of fluctuating membranes with fixed area", Soft Matter **12**, 2373–2380 (2016).

[80]E. Sezgin, I. Levental, S. Mayor, and C. Eggeling, "The mystery of membrane organization: composition, regulation and roles of lipid rafts", Nature Reviews Molecular Cell Biology **18**, 361–374 (2017).

[81]T. Baumgart, S. T. Hess, and W. W. Webb, "Imaging coexisting fluid domains in biomembrane models coupling curvature and line tension", Nature **425**, 821–824 (2003).

[82]J. H. Hurley, E. Boura, L.-A. Carlson, and B. Różycki, "Membrane budding", Cell **143**, 875–887 (2010).

[83]H. Sunshine and M. L. Iruela-Arispe, “Membrane lipids and cell signaling”, Current opinion in lipidology **28**, 408–413 (2017).

[84]M. I. Bukrinsky, N. Mukhamedova, and D. Sviridov, “Lipid rafts and pathogens: the art of deception and exploitation: thematic review series: biology of lipid rafts”, Journal of Lipid Research **61**, 601–610 (2020).

[85]A. B. Ouweneel, M. J. Thomas, and M. G. Sorci-Thomas, “The ins and outs of lipid rafts: functions in intracellular cholesterol homeostasis, microparticles, and cell membranes: thematic review series: biology of lipid rafts”, Journal of Lipid Research **61**, 676–686 (2020).

[86]P. Sengupta, D. Holowka, and B. Baird, “Fluorescence resonance energy transfer between lipid probes detects nanoscopic heterogeneity in the plasma membrane of live cells”, Biophysical Journal **92**, 3564–3574 (2007).

[87]Y. Zhou, C.-O. Wong, K.-j. Cho, D. van der Hoeven, H. Liang, D. P. Thakur, J. Luo, M. Babic, K. E. Zinsmaier, M. X. Zhu, H. Hu, K. Venkatachalam, and J. F. Hancock, “Membrane potential modulates plasma membrane phospholipid dynamics and k-ras signaling”, Science **349**, 873–876 (2015).

[88]C. Eggeling, C. Ringemann, R. Medda, G. Schwarzmann, K. Sandhoff, S. Polyakova, V. N. Belov, B. Hein, C. Von Middendorff, A. Schönle, et al., “Direct observation of the nanoscale dynamics of membrane lipids in a living cell”, Nature **457**, 1159–1162 (2009).

[89]E. Sezgin, F. Schneider, S. Galiani, I. Urbančič, D. Waithe, B. C. Lagerholm, and C. Eggeling, “Measuring nanoscale diffusion dynamics in cellular membranes with super-resolution sted–fcs”, Nature Protocols **14**, 1054–1083 (2019).

[90]M. J. Gerl, J. L. Sampaio, S. Urban, L. Kalvodova, J.-M. Verbavatz, B. Binnington, D. Lindemann, C. A. Lingwood, A. Shevchenko, C. Schroeder, and K. Simons, “Quantitative analysis of the lipidomes of the influenza virus envelope and MDCK cell apical membrane”, Journal of Cell Biology **196**, 213–221 (2012).

[91]H. Ogiso, M. Taniguchi, and T. Okazaki, “Analysis of lipid-composition changes in plasma membrane microdomains[s]”, Journal of Lipid Research **56**, 1594–1605 (2015).

[92]M. M. Lozano, J. S. Hovis, F. R. Moss, and S. G. Boxer, “Dynamic reorganization and correlation among lipid raft components”, Journal of the American Chemical Society **138**, PMID: 27447959, 9996–10001 (2016).

[93]N. Komura, K. G. Suzuki, H. Ando, M. Konishi, M. Koikeda, A. Imamura, R. Chadda, T. K. Fujiwara, H. Tsuboi, R. Sheng, et al., “Raft-based interactions of gangliosides with a gpi-anchored receptor”, Nature Chemical Biology **12**, 402–410 (2016).

[94]M. Kinoshita, K. G. Suzuki, N. Matsumori, M. Takada, H. Ano, K. Morigaki, M. Abe, A. Makino, T. Kobayashi, K. M. Hirosawa, T. K. Fujiwara, A. Kusumi, and M. Murata, “Raft-based sphingomyelin interactions revealed by new fluorescent sphingomyelin analogs”, Journal of Cell Biology **216**, 1183–1204 (2017).

[95] I. Levental, K. R. Levental, and F. A. Heberle, "Lipid rafts: controversies resolved, mysteries remain", Trends in Cell Biology **30**, 341–353 (2020).

[96] S. L. Veatch and S. L. Keller, "Organization in lipid membranes containing cholesterol", Physical Review Letters **89**, 268101 (2002).

[97] T. Baumgart, A. T. Hammond, P. Sengupta, S. T. Hess, D. A. Holowka, B. A. Baird, and W. W. Webb, "Large-scale fluid/fluid phase separation of proteins and lipids in giant plasma membrane vesicles", Proceedings of the National Academy of Sciences **104**, 3165–3170 (2007).

[98] S. L. Veatch, P. Cicuta, P. Sengupta, A. Honerkamp-Smith, D. Holowka, and B. Baird, "Critical fluctuations in plasma membrane vesicles", ACS Chemical Biology **3**, 287–293 (2008).

[99] T. R. Shaw, S. Ghosh, and S. L. Veatch, "Critical phenomena in plasma membrane organization and function", Annual Review of Physical Chemistry **72**, PMID: 33710910, 51–72 (2021).

[100] C. E. Cornell, A. Mileant, N. Thakkar, K. K. Lee, and S. L. Keller, "Direct imaging of liquid domains in membranes by cryo-electron tomography", Proceedings of the National Academy of Sciences **117**, 19713–19719 (2020).

[101] F. A. Heberle, M. Doktorova, H. L. Scott, A. D. Skinkle, M. N. Waxham, and I. Levental, "Direct label-free imaging of nanodomains in biomimetic and biological membranes by cryogenic electron microscopy", Proceedings of the National Academy of Sciences **117**, 19943–19952 (2020).

[102] J. H. Lorent and I. Levental, "Structural determinants of protein partitioning into ordered membrane domains and lipid rafts", Chemistry and Physics of Lipids **192**, ORNL workshop on Biomembranes, 23–32 (2015).

[103] J. H. Lorent, B. Diaz-Rohrer, X. Lin, K. Spring, A. A. Gorfe, K. R. Levental, and I. Levental, "Structural determinants and functional consequences of protein affinity for membrane rafts", Nature Communications **8**, 1–10 (2017).

[104] D. Chandler, "Interfaces and the driving force of hydrophobic assembly", Nature **437**, 640–647 (2005).

[105] Q. Lin and E. London, "Altering hydrophobic sequence lengths shows that hydrophobic mismatch controls affinity for ordered lipid domains (rafts) in the multitransmembrane strand protein perfringolysin o", Journal of Biological Chemistry **288**, 1340–1352 (2013).

[106] B. B. Diaz-Rohrer, K. R. Levental, K. Simons, and I. Levental, "Membrane raft association is a determinant of plasma membrane localization", Proceedings of the National Academy of Sciences **111**, 8500–8505 (2014).

[107] A. R. Honerkamp-Smith, P. Cicuta, M. D. Collins, S. L. Veatch, M. den Nijs, M. Schick, and S. L. Keller, "Line tensions, correlation lengths, and critical exponents in lipid membranes near critical points", Biophysical Journal **95**, 236–246 (2008).

[108]P. Hohenberg and A. Krekhov, "An introduction to the ginzburg–landau theory of phase transitions and nonequilibrium patterns", Physics Reports **572**, An introduction to the Ginzburg–Landau theory of phase transitions and nonequilibrium patterns, 1–42 (2015).

[109]E. Ising, "Beitrag zur theorie des ferro-und paramagnetismus", PhD thesis (Grefe & Tiedemann, 1924).

[110]G. Gompper and D. M. Kroll, "Network models of fluid, hexatic and polymerized membranes", Journal of Physics: Condensed Matter **9**, 8795–8834 (1997).

[111]P. B. Sunil Kumar and M. Rao, "Novel monte carlo approach to the dynamics of fluids: single-particle diffusion, correlation functions, and phase ordering of binary fluids", Physical Review Letters **77**, 1067–1070 (1996).

[112]P. B. S. Kumar and M. Rao, "Shape instabilities in the dynamics of a two-component fluid membrane", Physical Review Letters **80**, 2489–2492 (1998).

[113]A. J. García-Sáez, S. Chiantia, and P. Schwille, "Effect of line tension on the lateral organization of lipid membranes", Journal of Biological Chemistry **282**, 33537–33544 (2007).

[114]A. Tian, C. Johnson, W. Wang, and T. Baumgart, "Line tension at fluid membrane domain boundaries measured by micropipette aspiration", Physical Review Letters **98**, 208102 (2007).

[115]N. Fuller, C. R. Benatti, and R. P. Rand, "Curvature and bending constants for phosphatidylserine containing membranes", Biophysical Journal **85**, 1667–1674 (2003).

[116]J. Zimmerberg and M. M. Kozlov, "How proteins produce cellular membrane curvature", Nature Reviews Molecular Cell Biology **7**, 9–19 (2006).

[117]B. Kollmitzer, P. Heftberger, M. Rappolt, and G. Pabst, "Monolayer spontaneous curvature of raft-forming membrane lipids", Soft Matter **9**, 10877–10884 (2013).

[118]T. R. Graham and M. M. Kozlov, "Interplay of proteins and lipids in generating membrane curvature", Current Opinion in Cell Biology **22**, Membranes and organelles, 430–436 (2010).

[119]H. T. McMahon and E. Boucrot, "Membrane curvature at a glance", Journal of Cell Science **128**, 1065–1070 (2015).

[120]I. K. Jarsch, F. Daste, and J. L. Gallop, "Membrane curvature in cell biology: An integration of molecular mechanisms", Journal of Cell Biology **214**, 375–387 (2016).

[121]W. Rawicz, K. Olbrich, T. McIntosh, D. Needham, and E. Evans, "Effect of chain length and unsaturation on elasticity of lipid bilayers", Biophysical Journal **79**, 328–339 (2000).

[122]J. Song and R. Waugh, "Bending rigidity of sopc membranes containing cholesterol", Biophysical Journal **64**, 1967–1970 (1993).

[123]A. Roux, D. Cuvelier, P. Nassoy, J. Prost, P. Bassereau, and B. Goud, "Role of curvature and phase transition in lipid sorting and fission of membrane tubules", The EMBO Journal **24**, 1537–1545 (2005).

[124] J. Henriksen, A. Rowat, E. Brief, Y. Hsueh, J. Thewalt, M. Zuckermann, and J. Ipsen, “Universal behavior of membranes with sterols”, Biophysical Journal **90**, 1639–1649 (2006).

[125] A. Tian, B. R. Capraro, C. Esposito, and T. Baumgart, “Bending stiffness depends on curvature of ternary lipid mixture tubular membranes”, Biophysical Journal **97**, 1636–1646 (2009).

[126] B. Sorre, A. Callan-Jones, J.-B. Manneville, P. Nassoy, J.-F. Joanny, J. Prost, B. Goud, and P. Bassereau, “Curvature-driven lipid sorting needs proximity to a demixing point and is aided by proteins”, Proceedings of the National Academy of Sciences **106**, 5622–5626 (2009).

[127] R. S. Gracià, N. Bezlyepkina, R. L. Knorr, R. Lipowsky, and R. Dimova, “Effect of cholesterol on the rigidity of saturated and unsaturated membranes: fluctuation and electrodeformation analysis of giant vesicles”, Soft Matter **6**, 1472–1482 (2010).

[128] G. M. Torrie and J. P. Valleau, “Monte carlo free energy estimates using non-boltzmann sampling: application to the sub-critical lennard-jones fluid”, Chemical Physics Letters **28**, 578–581 (1974).

[129] G. Torrie and J. Valleau, “Nonphysical sampling distributions in Monte Carlo free-energy estimation: Umbrella sampling”, Journal of Computational Physics **23**, 187–199 (1977).

[130] J. Kästner, “Umbrella sampling”, WIREs Computational Molecular Science **1**, 932–942 (2011).

[131] M. R. Shirts and J. D. Chodera, “Statistically optimal analysis of samples from multiple equilibrium states”, The Journal of Chemical Physics **129**, 124105 (2008).

[132] M. R. Shirts, *Reweighting from the mixture distribution as a better way to describe the Multistate Bennett Acceptance Ratio*, 2017, arXiv:1704.00891.

[133] R. W. Zwanzig, “High-temperature equation of state by a perturbation method. I. Nonpolar gases”, The Journal of Chemical Physics **22**, 1420–1426 (1954).

[134] G. J. McLachlan, S. X. Lee, and S. I. Rathnayake, “Finite mixture models”, Annual Review of Statistics and Its Application **6**, 355–378 (2019).

[135] P. S. Varilly, *Fluctuations in Water and their Relation to the Hydrophobic Effect* (University of California, Berkeley, 2011).

[136] Z. Tan, E. Gallicchio, M. Lapelosa, and R. M. Levy, “Theory of binless multi-state free energy estimation with applications to protein-ligand binding”, The Journal of Chemical Physics **136**, 144102 (2012).

[137] E. H. Thiede, B. Van Koten, J. Weare, and A. R. Dinner, “Eigenvector method for umbrella sampling enables error analysis”, The Journal of Chemical Physics **145**, 084115 (2016).

[138] J. D. Chodera, W. C. Swope, J. W. Pitera, C. Seok, and K. A. Dill, “Use of the weighted histogram analysis method for the analysis of simulated and parallel tempering simulations”, Journal of Chemical Theory and Computation **3**, 26–41 (2007).

[139]H. R. Kunsch, "The jackknife and the bootstrap for general stationary observations", The Annals of Statistics **17**, 1217–1241 (1989).

[140]B. Efron and R. J. Tibshirani, *An Introduction to the Bootstrap* (CRC Press, 1994).

[141]Y. Sugita and Y. Okamoto, "Replica-exchange molecular dynamics method for protein folding", Chemical Physics Letters **314**, 141–151 (1999).

[142]Y. Sugita, A. Kitao, and Y. Okamoto, "Multidimensional replica-exchange method for free-energy calculations", The Journal of Chemical Physics **113**, 6042–6051 (2000).

[143]J. D. Chodera and M. R. Shirts, "Replica exchange and expanded ensemble simulations as gibbs sampling: simple improvements for enhanced mixing", The Journal of Chemical Physics **135**, 194110 (2011).

[144]K. Binder and D. P. Landau, "Finite-size scaling at first-order phase transitions", Physical Review B **30**, 1477–1485 (1984).

[145]C. Borgs and R. Kotecký, "A rigorous theory of finite-size scaling at first-order phase transitions", Journal of Statistical Physics **61**, 79–119 (1990).

[146]R. P. Brent, *Algorithms for Minimization Without Derivatives* (Courier Corporation, 2013).

[147]M. E. Fisher and A. E. Ferdinand, "Interfacial, boundary, and size effects at critical points", Physical Review Letters **19**, 169–172 (1967).

[148]A. P. Willard and D. Chandler, "The role of solvent fluctuations in hydrophobic assembly", The Journal of Physical Chemistry B **112**, 6187–6192 (2008).

[149]S. L. Veatch and S. L. Keller, "Separation of liquid phases in giant vesicles of ternary mixtures of phospholipids and cholesterol", Biophysical Journal **85**, 3074–3083 (2003).

[150]K. Bacia, P. Schwille, and T. Kurzchalia, "Sterol structure determines the separation of phases and the curvature of the liquid-ordered phase in model membranes", Proceedings of the National Academy of Sciences **102**, 3272–3277 (2005).

[151]M. Andes-Koback and C. D. Keating, "Complete budding and asymmetric division of primitive model cells to produce daughter vesicles with different interior and membrane compositions", Journal of the American Chemical Society **133**, 9545–9555 (2011).

[152]R. Lipowsky, "Budding of membranes induced by intramembrane domains", Journal de Physique II **2**, 1825–1840 (1992).

[153]F. Jülicher and R. Lipowsky, "Domain-induced budding of vesicles", Physical Review Letters **70**, 2964–2967 (1993).

[154]T. S. Ursell, W. S. Klug, and R. Phillips, "Morphology and interaction between lipid domains", Proceedings of the National Academy of Sciences **106**, 13301–13306 (2009).

[155]T. R. Weikl, M. M. Kozlov, and W. Helfrich, "Interaction of conical membrane inclusions: effect of lateral tension", Physical Review E **57**, 6988–6995 (1998).

[156]T. H. Cormen, C. E. Leiserson, R. L. Rivest, and C. Stein, *Introduction to Algorithms* (MIT press, 2009).

[157] J. Cornet, N. Destainville, and M. Manghi, “Domain formation in bicomponent vesicles induced by composition-curvature coupling”, The Journal of Chemical Physics **152**, 244705 (2020).

[158] R. L. Burden, J. D. Faires, and A. M. Burden, *Numerical Analysis*, 10th ed. (Cengage Learning, 2015).

[159] F. A. Heberle, R. S. Petruzielo, J. Pan, P. Drazba, N. Kučerka, R. F. Standaert, G. W. Feigenson, and J. Katsaras, “Bilayer thickness mismatch controls domain size in model membranes”, Journal of the American Chemical Society **135**, PMID: 23391155, 6853–6859 (2013).

[160] C. E. Cornell, A. D. Skinkle, S. He, I. Levental, K. R. Levental, and S. L. Keller, “Tuning length scales of small domains in cell-derived membranes and synthetic model membranes”, Biophysical Journal **115**, 690–701 (2018).

[161] W.-C. Tsai and G. W. Feigenson, “Lowering line tension with high cholesterol content induces a transition from macroscopic to nanoscopic phase domains in model biomembranes”, Biochimica et Biophysica Acta (BBA) - Biomembranes **1861**, 478–485 (2019).

[162] T. A. Enoki, J. Wu, F. A. Heberle, and G. W. Feigenson, “Investigation of the domain line tension in asymmetric vesicles prepared via hemifusion”, Biochimica et Biophysica Acta (BBA) - Biomembranes **1863**, 183586 (2021).

[163] L. Scheidegger, L. Stricker, P. J. Beltramo, and J. Vermant, “Domain size regulation in phospholipid model membranes using oil molecules and hybrid lipids”, The Journal of Physical Chemistry B **0**, PMID: 35895895, null (0).

[164] K. Lum, D. Chandler, and J. D. Weeks, “Hydrophobicity at small and large length scales”, The Journal of Physical Chemistry B **103**, 4570–4577 (1999).

[165] P. Sens and M. S. Turner, “Budded membrane microdomains as tension regulators”, Physical Review E **73**, 031918 (2006).

[166] N. Walani, J. Torres, and A. Agrawal, “Endocytic proteins drive vesicle growth via instability in high membrane tension environment”, Proceedings of the National Academy of Sciences **112**, E1423–E1432 (2015).

[167] D. Bucher, F. Frey, K. A. Sochacki, S. Kummer, J.-P. Bergeest, W. J. Godinez, H.-G. Kräusslich, K. Rohr, J. W. Taraska, U. S. Schwarz, et al., “Clathrin-adaptor ratio and membrane tension regulate the flat-to-curved transition of the clathrin coat during endocytosis”, Nature Communications **9**, 1–13 (2018).

[168] E. Gutlederer, T. Gruhn, and R. Lipowsky, “Polymorphism of vesicles with multi-domain patterns”, Soft Matter **5**, 3303–3311 (2009).

[169] J. Hu, T. Weikl, and R. Lipowsky, “Vesicles with multiple membrane domains”, Soft Matter **7**, 6092–6102 (2011).

[170]S. L. Goh, J. J. Amazon, and G. W. Feigenson, "Toward a better raft model: modulated phases in the four-component bilayer, dspc/dopc/popc/chol", Biophysical Journal **104**, 853–862 (2013).

[171]J. J. Amazon, S. L. Goh, and G. W. Feigenson, "Competition between line tension and curvature stabilizes modulated phase patterns on the surface of giant unilamellar vesicles: a simulation study", Physical Review E **87**, 022708 (2013).

[172]J. J. Amazon and G. W. Feigenson, "Lattice simulations of phase morphology on lipid bilayers: renormalization, membrane shape, and electrostatic dipole interactions", Physical Review E **89**, 022702 (2014).

[173]G. Gueguen, N. Destainville, and M. Manghi, "Mixed lipid bilayers with locally varying spontaneous curvature and bending", The European Physical Journal E **37**, 1–13 (2014).

[174]N. Destainville, M. Manghi, and J. Cornet, "A rationale for mesoscopic domain formation in biomembranes", Biomolecules **8** (2018).

[175]A. Diz-Muñoz, D. A. Fletcher, and O. D. Weiner, "Use the force: membrane tension as an organizer of cell shape and motility", Trends in Cell Biology **23**, 47–53 (2013).

[176]P. Sens and J. Plastino, "Membrane tension and cytoskeleton organization in cell motility", Journal of Physics: Condensed Matter **27**, 273103 (2015).

[177]B. B. Machta, S. Papanikolaou, J. P. Sethna, and S. L. Veatch, "Minimal model of plasma membrane heterogeneity requires coupling cortical actin to criticality", Biophysical Journal **100**, 1668–1677 (2011).

[178]S. Arumugam, E. P. Petrov, and P. Schwille, "Cytoskeletal pinning controls phase separation in multicomponent lipid membranes", Biophysical Journal **108**, 1104–1113 (2015).

[179]C. King, P. Sengupta, A. Y. Seo, and J. Lippincott-Schwartz, "Er membranes exhibit phase behavior at sites of organelle contact", Proceedings of the National Academy of Sciences **117**, 7225–7235 (2020).

[180]L. A. Gheber and M. Edidin, "A model for membrane patchiness: lateral diffusion in the presence of barriers and vesicle traffic", Biophysical Journal **77**, 3163–3175 (1999).

[181]L. Foret, "A simple mechanism of raft formation in two-component fluid membranes", Europhysics Letters (EPL) **71**, 508–514 (2005).

[182]M. S. Turner, P. Sens, and N. D. Socci, "Nonequilibrium raftlike membrane domains under continuous recycling", Physical Review Letters **95**, 168301 (2005).

[183]T. Das, T. K. Maiti, and S. Chakraborty, "Nanodomain stabilization dynamics in plasma membranes of biological cells", Physical Review E **83**, 021909 (2011).

[184]M. Girard and T. Bereau, "Regulating lipid composition rationalizes acyl tail saturation homeostasis in ectotherms", Biophysical Journal **119**, 892–899 (2020).

[185]Y. K. Cherniavskyi, A. Fathizadeh, R. Elber, and D. P. Tieleman, "Computer simulations of a heterogeneous membrane with enhanced sampling techniques", The Journal of Chemical Physics **153**, 144110 (2020).

[186]M. Girard and T. Bereau, “Computer simulations of lipid regulation by molecular semigrand canonical ensembles”, Biophysical Journal **120**, 2370–2373 (2021).

[187]P. J. Lu and D. A. Weitz, “Colloidal particles: Crystals, glasses, and gels”, Annual Review of Condensed Matter Physics **4**, 217–233 (2013).

[188]P. Jungwirth and B. Winter, “Ions at aqueous interfaces: From water surface to hydrated proteins”, Annual Review of Physical Chemistry **59**, 343–366 (2008).

[189]K. A. Dill, S. B. Ozkan, M. S. Shell, and T. R. Weikl, “The protein folding problem”, Annual Review of Biophysics **37**, 289–316 (2008).

[190]S. Plimpton, “Fast parallel algorithms for short-range molecular dynamics”, Journal of Computational Physics **117**, 1–19 (1995).

[191]S. Pronk, S. Páll, R. Schulz, P. Larsson, P. Bjelkmar, R. Apostolov, M. R. Shirts, J. C. Smith, P. M. Kasson, D. van der Spoel, et al., “GROMACS 4.5: A high-throughput and highly parallel open source molecular simulation toolkit”, Bioinformatics **29**, 845–854 (2013).

[192]P. L. Geissler, C. Dellago, D. Chandler, J. Hutter, and M. Parrinello, “Autoionization in liquid water”, Science **291**, 2121–2124 (2001).

[193]I. J. Ford, “Statistical mechanics of nucleation: a review”, Proceedings of the Institution of Mechanical Engineers, Part C: Journal of Mechanical Engineering Science **218**, 883–899 (2004).

[194]D. Erdemir, A. Y. Lee, and A. S. Myerson, “Nucleation of crystals from solution: classical and two-step models”, Accounts of Chemical Research **42**, PMID: 19402623, 621–629 (2009).

[195]C. H. Bennett, “Molecular dynamics and transition state theory: the simulation of infrequent events”, in *Algorithms for Chemical Computations* (1977) Chap. 4, pp. 63–97.

[196]D. Chandler, “Statistical mechanics of isomerization dynamics in liquids and the transition state approximation”, The Journal of Chemical Physics **68**, 2959–2970 (1978).

[197]R. A. Kuharski, D. Chandler, J. A. Montgomery, F. Rabii, and S. J. Singer, “Stochastic molecular dynamics study of cyclohexane isomerization”, The Journal of Physical Chemistry **92**, 3261–3267 (1988).

[198]B. J. Berne, M. Borkovec, and J. E. Straub, “Classical and modern methods in reaction rate theory”, The Journal of Physical Chemistry **92**, 3711–3725 (1988).

[199]B. Peters, “Reaction coordinates and mechanistic hypothesis tests”, Annual Review of Physical Chemistry **67**, 669–690 (2016).

[200]C. Dellago, P. G. Bolhuis, F. S. Csajka, and D. Chandler, “Transition path sampling and the calculation of rate constants”, The Journal of Chemical Physics **108**, 1964–1977 (1998).

[201]P. G. Bolhuis, D. Chandler, C. Dellago, and P. L. Geissler, “Transition path sampling: Throwing ropes over rough mountain passes, in the dark”, Annual Review of Physical Chemistry **53**, 291–318 (2002).

[202]W. E and E. Vanden-Eijnden, "Towards a theory of transition paths", Journal of Statistical Physics **123**, 503–523 (2006).

[203]W. E and E. Vanden-Eijnden, "Transition-path theory and path-finding algorithms for the study of rare events", Annual Review of Physical Chemistry **61**, 391–420 (2010).

[204]W. E, W. Ren, and E. Vanden-Eijnden, "Finite temperature string method for the study of rare events", The Journal of Physical Chemistry B **109**, 6688–6693 (2005).

[205]E. Vanden-Eijnden and M. Venturoli, "Revisiting the finite temperature string method for the calculation of reaction tubes and free energies", The Journal of Chemical Physics **130**, 194103 (2009).

[206]B. Peters, "Using the histogram test to quantify reaction coordinate error", The Journal of Chemical Physics **125**, 241101 (2006).

[207]A. M. Berezhkovskii and A. Szabo, "Diffusion along the splitting/commitment probability reaction coordinate", The Journal of Physical Chemistry B **117**, 13115–13119 (2013).

[208]R. Durrett, *Stochastic Calculus: A Practical Introduction* (CRC Press, 1996).

[209]E. Weinan, T. Li, and E. Vanden-Eijnden, *Applied Stochastic Analysis*, Vol. 199 (American Mathematical Soc., 2021).

[210]E. Vanden-Eijnden, "Transition Path Theory", in *Computer Simulations in Condensed Matter Systems: From Materials to Chemical Biology* (Springer, 2006), pp. 453–493.

[211]S. Osher and R. Fedkiw, "Implicit Functions", in *Level Set Methods and Dynamic Implicit Surfaces* (Springer New York, New York, NY, 2003).

[212]T. Hastie and W. Stuetzle, "Principal curves", Journal of the American Statistical Association **84**, 502–516 (1989).

[213]B. Polyak, "Some methods of speeding up the convergence of iteration methods", USSR Computational Mathematics and Mathematical Physics **4**, 1–17 (1964).

[214]Y. Nesterov, "A method for unconstrained convex minimization problem with the rate of convergence $O(1/k^2)$", Soviet Mathematics Doklady **27**, 372–376 (1983).

[215]K. Müller and L. D. Brown, "Location of saddle points and minimum energy paths by a constrained simplex optimization procedure", Theoretica Chimica Acta **53**, 75–93 (1979).

[216]A. Logg, K.-A. Mardal, and G. Wells, *Automated Solution of Differential Equations by the Finite Element Method: The FEniCS Book* (Springer, 2012).

[217]M. Alnæs, J. Blechta, J. Hake, A. Johansson, B. Kehlet, A. Logg, C. Richardson, J. Ring, M. E. Rognes, and G. N. Wells, "The FEniCS project version 1.5", Archive of Numerical Software **3** (2015).

[218]W. Kabsch, "A solution for the best rotation to relate two sets of vectors", Acta Crystallographica Section A: Crystal Physics, Diffraction, Theoretical and General Crystallography **32**, 922–923 (1976).

[219]L. Onsager, "Initial recombination of ions", Phys. Rev. **54**, 554–557 (1938).

[220]L. Maragliano, A. Fischer, E. Vanden-Eijnden, and G. Ciccotti, "String method in collective variables: Minimum free energy paths and isocommittor surfaces", The Journal of Chemical Physics **125**, 024106 (2006).

[221]H. Li, Y. Khoo, Y. Ren, and L. Ying, *A semigroup method for high dimensional committor functions based on neural network*, 2021, arXiv:2012.06727.

[222]J. Nocedal and S. Wright, *Numerical Optimization* (Springer-Verlag New York, 2006).

[223]A. Ma and A. R. Dinner, "Automatic method for identifying reaction coordinates in complex systems", The Journal of Physical Chemistry B **109**, PMID: 16851762, 6769–6779 (2005).

[224]B. Peters and B. L. Trout, "Obtaining reaction coordinates by likelihood maximization", The Journal of Chemical Physics **125**, 054108 (2006).

[225]H. Robbins and S. Monro, "A stochastic approximation method", Annals of Mathematical Statistics **22**, 400–407 (1951).

[226]D. P. Kingma and J. Ba, *Adam: A method for stochastic optimization*, 2017, arXiv:1412.6980.

[227]T. Lelièvre, M. Rousset, and G. Stoltz, *Free Energy Computations* (Imperial College Press, 2010).

[228]D. Wu and D. A. Kofke, "Phase-space overlap measures. I. Fail-safe bias detection in free energies calculated by molecular simulation", The Journal of Chemical Physics **123**, 54103 (2005).

[229]A. Pohorille, C. Jarzynski, and C. Chipot, "Good practices in free-energy calculations", The Journal of Physical Chemistry B **114**, 10235–10253 (2010).

[230]S. S. Du, X. Zhai, B. Poczos, and A. Singh, *Gradient descent provably optimizes over-parameterized neural networks*, 2019, arXiv:1810.02054.

[231]S. S. Du, J. Lee, H. Li, L. Wang, and X. Zhai, "Gradient descent finds global minima of deep neural networks", Proceedings of Machine Learning Research **97**, 1675–1685 (2019).

[232]M. Hardt and B. Recht, *Patterns, Predictions, and Actions: A story about machine learning* (https://mlstory.org, 2021), arXiv:2102.05242.

[233]E. Vanden-Eijnden and M. Venturoli, "Markovian milestoning with Voronoi tessellations", The Journal of Chemical Physics **130**, 194101 (2009).

[234]M. R. Shirts and V. S. Pande, "Comparison of efficiency and bias of free energies computed by exponential averaging, the Bennett acceptance ratio, and thermodynamic integration", The Journal of Chemical Physics **122**, 144107 (2005).

[235]K. Zinovjev and I. Tuñón, "Adaptive finite temperature string method in collective variables", The Journal of Physical Chemistry A **121**, 9764–9772 (2017).

[236]K. Pearson, "Method of moments and method of maximum likelihood", Biometrika **28**, 34–59 (1936).

[237]C. Forbes, M. Evans, N. Hastings, and B. Peacock, *Statistical Distributions*, 4th ed. (John Wiley & Sons, 2011).

[238]N. A. Marlow, "A normal limit theorem for power sums of independent random variables", Bell System Technical Journal **46**, 2081–2089 (1967).

[239]E. Barouch, G. Kaufman, and M. Glasser, "On sums of lognormal random variables", Studies in Applied Mathematics **75**, 37–55 (1986).

[240]N. Beaulieu, A. Abu-Dayya, and P. McLane, "Estimating the distribution of a sum of independent lognormal random variables", IEEE Transactions on Communications **43**, 2869 (1996).

[241]N. B. Mehta, J. Wu, A. F. Molisch, and J. Zhang, "Approximating a sum of random variables with a lognormal", IEEE Transactions on Wireless Communications **6**, 2690–2699 (2007).

[242]S. Asmussen and L. Rojas-Nandayapa, "Asymptotics of sums of lognormal random variables with gaussian copula", Statistics and Probability Letters **78**, 2709–2714 (2008).

[243]C. Dellago, P. G. Bolhuis, and D. Chandler, "On the calculation of reaction rate constants in the transition path ensemble", The Journal of Chemical Physics **110**, 6617–6625 (1999).

[244]J. D. Weeks, D. Chandler, and H. C. Andersen, "Role of repulsive forces in determining the equilibrium structure of simple liquids", The Journal of Chemical Physics **54**, 5237–5247 (1971).

[245]K. T. Schütt, H. E. Sauceda, P.-J. Kindermans, A. Tkatchenko, and K.-R. Müller, "Schnet–a deep learning architecture for molecules and materials", The Journal of Chemical Physics **148**, 241722 (2018).

[246]M. Fey and J. E. Lenssen, *Fast graph representation learning with PyTorch Geometric*, 2019, arXiv:1903.02428.

[247]N. Thomas, T. Smidt, S. Kearnes, L. Yang, L. Li, K. Kohlhoff, and P. Riley, *Tensor field networks: rotation- and translation-equivariant neural networks for 3d point clouds*, 2018, arXiv:1802.08219.

[248]S. Batzner, A. Musaelian, L. Sun, M. Geiger, J. P. Mailoa, M. Kornbluth, N. Molinari, T. E. Smidt, and B. Kozinsky, *E(3)-equivariant graph neural networks for data-efficient and accurate interatomic potentials*, 2021, arXiv:2101.03164.

[249]A. Laio and M. Parrinello, "Escaping free-energy minima", Proceedings of the National Academy of Sciences **99**, 12562–12566 (2002).

[250]P. L. Geissler, C. Dellago, and D. Chandler, "Kinetic pathways of ion pair dissociation in water", The Journal of Physical Chemistry B **103**, 3706–3710 (1999).

[251] A. J. Ballard and C. Dellago, “Toward the mechanism of ionic dissociation in water”, The Journal of Physical Chemistry B **116**, 13490–13497 (2012).

[252] M. R. Hasyim and K. K. Mandadapu, “A theory of localized excitations in supercooled liquids”, J. Chem. Phys. **155**, 044504 (2021).

[253] C. Y. Lee and H. L. Scott, “The surface tension of water: a monte carlo calculation using an umbrella sampling algorithm”, The Journal of Chemical Physics **73**, 4591–4596 (1980).

[254] A. Paszke, S. Gross, F. Massa, A. Lerer, J. Bradbury, G. Chanan, T. Killeen, Z. Lin, N. Gimelshein, L. Antiga, A. Desmaison, A. Kopf, E. Yang, Z. DeVito, M. Raison, A. Tejani, S. Chilamkurthy, B. Steiner, L. Fang, J. Bai, and S. Chintala, “PyTorch: An imperative style, high-performance deep learning library”, Advances in Neural Information Processing Systems **32**, 8024–8035 (2019).

[255] Y. Bengio, N. Boulanger-Lewandowski, and R. Pascanu, “Advances in optimizing recurrent networks”, IEEE International Conference on Acoustics, Speech and Signal Processing, 8624–8628 (2013).

[256] N. Metropolis and S. Ulam, “The Monte Carlo method”, Journal of the American Statistical Association **44**, 335–341 (1949).