# Supervised Learning and the Finite-Temperature String Method for Computing Committor Functions and Reaction Rates

**Muhammad R. Hasyim**[1,*,†], **Clay H. Batton**[1,*,‡], **Kranthi K. Mandadapu**[1,2,§]
[1]Department of Chemical & Biomolecular Engineering, University of California at Berkeley
[2]Chemical Sciences Division, Lawrence Berkeley National Laboratory

## Abstract

A central object in the computational studies of rare events is the committor function. Though costly to compute, the committor function encodes complete mechanistic information of the processes involving rare events, including reaction rates and transition-state ensembles. Under the framework of transition path theory (TPT), recent work [1] proposes an algorithm where a feedback loop couples a neural network that models the committor function with importance sampling, mainly umbrella sampling, which collects data needed for adaptive training. In this work, we show additional modifications are needed to improve the accuracy of the algorithm. The first modification adds elements of supervised learning, which allows the neural network to improve its prediction by fitting to sample-mean estimates of committor values obtained from short molecular dynamics trajectories. The second modification replaces the committor-based umbrella sampling with the finite-temperature string (FTS) method, which enables homogeneous sampling in regions where transition pathways are located. We test our modifications on low-dimensional systems with non-convex potential energy where reference solutions can be found via analytical or the finite element methods, and show how combining supervised learning and the FTS method yields accurate computation of committor functions and reaction rates. We also provide an error analysis for algorithms that use the FTS method, using which reaction rates can be accurately estimated during training with a small number of samples. The methods are then applied to a molecular system in which no reference solution is known, where accurate computations of committor functions and reaction rates can still be obtained.

## 1 Introduction

A fundamental problem in chemistry is to discover the mechanistic pathways governing kinetic processes at the microscopic level. These processes include phase transitions in colloidal systems [2], chemical reactions at aqueous interfaces [3], and protein folding [4]. 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 [5–7], 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.

*These two authors contributed equally to this work.
†`muhammad_hasyim@berkeley.edu`
‡`chbatton@berkeley.edu`
§`kranthi@berkeley.edu`

Several strategies exist for capturing rare barrier-crossing events, one of which is transition path sampling (TPS) [8, 9]; an importance sampling technique for generating an ensemble of transition pathways. An alternative strategy is to rely on transition path theory (TPT) [10, 11], which can outline various computational methods to obtain an average characteristic pathway, e.g., the finite-temperature string (FTS) method [12, 13]. 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 [14].

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) [10, 11, 15]. The complexity in solving the high-dimensional BKE may be reduced by constructing a low-dimensional set of collective variables (CVs) [16], but they are not known *a priori* and require exhaustive trial-and-error to obtain ones that best describe a reaction pathway [17]. 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 but focuses on important regions across the transition path. One way to target these regions is importance sampling [18] 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 [1] 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 [19] 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 work, we improve the algorithm in Ref. [1] 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 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 [13], 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.

Our paper is organized as follows: in Section 2.1, we review the framework of TPT to introduce the BKE and construct an optimization problem from the BKE that is feasible to solve using ML. In Section 2.2, we review the ML algorithm proposed in Ref. [1], 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 2.3 and ending with the review and use of the FTS method for importance sampling in Section 2.4. In Sections 3.1 and 3.2, 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 3.3, we provide an error analysis for algorithms that use the FTS method,

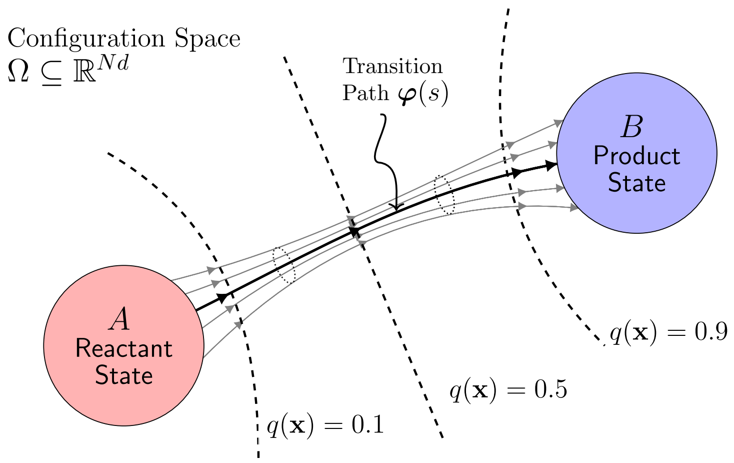


Figure 1: A schematic of transition path theory (TPT). Gray lines are flow lines of the probability flux $\mathbf{J}(\mathbf{x})$, and the transition tube, i.e., the region of high flux, is localized around the transition path $\boldsymbol{\varphi}(s)$. Dashed lines are isocommittor surfaces, with the middle dashed line defining the transition-state ensemble where $q(\mathbf{x}) = 0.5$.

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 4, 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.

## 2 Theory and Algorithms

### 2.1 From Transition Path Theory to Machine Learning

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. Equilibrium properties can be computed via ensemble averages $\langle \dots \rangle = \int_\Omega \mathrm{d}\mathbf{x}\rho(\mathbf{x}) \dots$ over the Boltzmann distribution $\rho(\mathbf{x}) = e^{-\beta V(\mathbf{x})}/Z$ where $\beta = 1/k_\mathrm{B}T$ with $k_\mathrm{B}$ being the Boltzmann constant, $T$ the temperature, and $Z = \int_\Omega \mathrm{d}\mathbf{x}\, e^{-\beta V(\mathbf{x})}$ the partition function. 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 $B \subset \Omega$ [10, 11, 20]; see Fig. 1 for a schematic of the problem. Central to TPT is the calculation of the committor function $q(\mathbf{x})$, which is defined as the probability to first reach $B$ before $A$ given that the system initially starts at $\mathbf{x}_0 = \mathbf{x}$. The formula for $q(\mathbf{x})$ is given by

$$q(\mathbf{x}) = \mathbb{E}\left[h_B\left(\mathbf{x}_\tau\right) \mid \mathbf{x}_0 = \mathbf{x}\right]; \quad \tau = \underset{t\in[0,+\infty)}{\arg\min}\{\mathbf{x}_t \in A \cup B : \mathbf{x}_0 = \mathbf{x}\}\,, \tag{2.1}$$

where $\mathbb{E}[\dots \mid \mathbf{x}_0 = \mathbf{x}]$ is an average over all trajectories starting from $\mathbf{x}$, $\tau$ is the first-passage time, and $h_C(\mathbf{x}) = 1$ if $\mathbf{x} \in C$ and zero otherwise. Using stochastic calculus [21], one may compute the committor as a solution to the steady-state backward Kolmogorov equation (BKE)

$$\nabla_\mathbf{x} \cdot \mathbf{J}(\mathbf{x}) = 0\,, \tag{2.2}$$

with $\mathbf{J}(\mathbf{x}) = \rho(\mathbf{x})\mathbf{D}(\mathbf{x})\nabla_\mathbf{x} q(\mathbf{x})$ being the probability flux and $\mathbf{D}(\mathbf{x})$ being the position-dependent diffusivity matrix, subjected to the boundary conditions

$$q(\mathbf{x}) = 0,\ \mathbf{x} \in \partial A; \quad q(\mathbf{x}) = 1,\ \mathbf{x} \in \partial B\,, \tag{2.3}$$

where $\partial A$ and $\partial B$ are the boundaries of $A$ and $B$ respectively.

Solving the BKE for the committor function allows us to evaluate many quantities including transition paths, 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 $|\mathbf{J}(\mathbf{x})| = \rho(\mathbf{x})\frac{k_\mathrm{B}T}{\gamma}|\nabla_\mathbf{x} q(\mathbf{x})|$ at a chosen level set of the committor function $q(\mathbf{x})$, i.e.,

$$\boldsymbol{\varphi}(s) = \frac{\int_P \mathrm{d}S|\mathbf{J}(\mathbf{x})|\mathbf{x}}{\int_P \mathrm{d}S|\mathbf{J}(\mathbf{x})|} = \frac{\int_P \mathrm{d}S\rho(\mathbf{x})|\nabla_\mathbf{x} q(\mathbf{x})|\mathbf{x}}{\int_P \mathrm{d}S\rho(\mathbf{x})|\nabla_\mathbf{x} q(\mathbf{x})|}\,, \tag{2.4}$$

where $\int_P \mathrm{d}S$ is a surface integral over the level set $P = \{\mathbf{x} \in \Omega : q(\mathbf{x}) = q(\boldsymbol{\varphi}(s))\}$ [10, 11]. 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. 1. 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 $\nu_R$, defined as the frequency with which a system transitions from $A$ to $B$, can be evaluated as [10]

$$\nu_R = \frac{k_\mathrm{B}T}{\gamma}\int_\Omega \mathrm{d}\mathbf{x}\, \rho(\mathbf{x})|\nabla_\mathbf{x} q(\mathbf{x})|^2 = \frac{k_\mathrm{B}T}{\gamma}\left\langle |\nabla_\mathbf{x} q(\mathbf{x})|^2 \right\rangle\,. \tag{2.5}$$

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 [1, 22–24]. To this end, we begin by constructing a variational form of the BKE. Following the standard procedure for elliptic PDEs [25], 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. (2.3). Multiplying Eq. (2.2) by $\delta q(\mathbf{x})$, integrating over $\Omega \setminus A \cup B$, and then integrating by parts yields

$$\int_{\Omega\setminus A\cup B} \mathrm{d}\mathbf{x}\, \delta q(\mathbf{x})\nabla_\mathbf{x}\left[\rho(\mathbf{x})\nabla_\mathbf{x} q(\mathbf{x})\right] = -\int_{\Omega\setminus A\cup B} \mathrm{d}\mathbf{x}\, \rho(\mathbf{x})\, \nabla_\mathbf{x}\delta q(\mathbf{x}) \cdot \nabla_\mathbf{x} q(\mathbf{x}) = 0\,. \tag{2.6}$$

Applying Vainberg's theorem [25] to Eq. (2.6) leads to the following functional:

$$L\left[\tilde{q}\right] = \frac{1}{2}\int_{\Omega\setminus A\cup B} \mathrm{d}\mathbf{x}\rho(\mathbf{x})|\nabla_\mathbf{x}\tilde{q}(\mathbf{x})|^2 = \frac{1}{2}\left\langle |\nabla_\mathbf{x}\tilde{q}(\mathbf{x})|^2 \right\rangle_{\Omega\setminus A\cup B} \tag{2.7}$$

whose extremization over the space of admissible functions $\tilde{q}(\mathbf{x})$ subject to boundary conditions Eq. (2.3) leads to the solution of the BKE. The variational form in Eq. (2.7) therefore transforms the strong form of BKE into a problem of functional optimization, where the committor function satisfies

$$q(\mathbf{x}) = \arg\min_{\tilde{q}} L\left[\tilde{q}\right] \quad \text{s.t.} \quad \tilde{q}(\mathbf{x}) = 0,\ \mathbf{x} \in \partial A; \quad \tilde{q}(\mathbf{x}) = 1,\ \mathbf{x} \in \partial B\,. \tag{2.8}$$

Equation (2.8) 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

$$\ell(\mathbf{x};\boldsymbol{\theta}) = \frac{1}{2}|\nabla_\mathbf{x}\hat{q}(\mathbf{x};\boldsymbol{\theta})|^2 \tag{2.9}$$

and imposing boundary conditions in Eq. (2.3) by the penalty method [26] with the loss functions

$$\ell_\mathrm{A}(\mathbf{x}_A;\boldsymbol{\theta}) = \frac{1}{2}\left(\hat{q}(\mathbf{x}_A;\boldsymbol{\theta})\right)^2\,, \tag{2.10}$$

$$\ell_\mathrm{B}(\mathbf{x}_B;\boldsymbol{\theta}) = \frac{1}{2}\left(\hat{q}(\mathbf{x}_B;\boldsymbol{\theta}) - 1\right)^2\,, \tag{2.11}$$

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:

$$L(\boldsymbol{\theta}) = \left\langle \ell(\mathbf{x};\boldsymbol{\theta}) \right\rangle + \lambda_\mathrm{A}\left\langle \ell_\mathrm{A}(\mathbf{x};\boldsymbol{\theta}) \right\rangle_A + \lambda_\mathrm{B}\left\langle \ell_\mathrm{B}(\mathbf{x};\boldsymbol{\theta}) \right\rangle_B\,. \tag{2.12}$$

Here, $\langle \ldots \rangle_C$ denotes ensemble averaging constrained in a region $C \subset \Omega$, and $\lambda_\mathrm{A}$ and $\lambda_\mathrm{B}$ 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. (2.5) 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. (2.12) 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. (2.12) with averages over samples obtained via molecular dynamics (MD) or Monte Carlo (MC) simulations. In this case, Eq. (2.12) can be evaluated as

$$\hat{L}(\boldsymbol{\theta};\mathcal{S},\mathcal{A},\mathcal{B}) = \frac{1}{|\mathcal{S}|}\sum_{\mathbf{x}\in\mathcal{S}}\ell(\mathbf{x};\boldsymbol{\theta}) + \frac{\lambda_\mathrm{A}}{|\mathcal{A}|}\sum_{\mathbf{x}\in\mathcal{A}}\ell_\mathrm{A}(\mathbf{x};\boldsymbol{\theta}) + \frac{\lambda_\mathrm{B}}{|\mathcal{B}|}\sum_{\mathbf{x}\in\mathcal{B}}\ell_\mathrm{B}(\mathbf{x};\boldsymbol{\theta})\,, \tag{2.13}$$

where $\mathcal{A}$, $\mathcal{B}$ and $\mathcal{S}$ are batches of samples obtained in the reactant state $A$, product state $B$ and configuration space $\Omega$, 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 [1, 22–24] 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 [27, 28]. The main challenge inherent in these approaches is sampling; since the first term in Eq. (2.12) 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. An inadequate sampling of the transition-state ensemble may lead to poor estimates of the average BKE loss function in Eq. (2.9), resulting in an inaccurate computation of committor functions and reaction rates in Eq. (2.5). 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. [1], 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.

## 2.2 Solving the BKE with Umbrella Sampling and Feedback Loops

In this section, we review the algorithm in Ref. [1] 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. 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 $\kappa$ is the bias strength. This bias leads to modified equations of motion

$$\mathbf{x}_{t+1} = \mathbf{x}_t - \gamma^{-1}\nabla_\mathbf{x}\left[V(\mathbf{x}_t) + W(\mathbf{x}_t;\boldsymbol{\theta})\right]\Delta t + \sqrt{2k_\mathrm{B}T\Delta t\gamma^{-1}}\mathbf{w}_t\,, \tag{2.14}$$

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 $\kappa$, 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. [1], 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. (2.14).

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) [29], Heavy-Ball [30], or Adam [31], to obtain model parameters $\boldsymbol{\theta}$ that extremize Eq. (2.13). 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 $\alpha$-th batch $\mathcal{M}_k^\alpha$ consists of samples obtained from a short MD/MC trajectory run of the $\alpha$-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.

**Algorithm 1:** The BKE–US Method [1]

```
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. (2.14).
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. (2.19).
8     Compute $\nabla_{\boldsymbol{\theta}} \hat{L}(\boldsymbol{\theta}_k; \{(\mathcal{M}_k^\alpha, z_\alpha)\}, \mathcal{A}_k, \mathcal{B}_k)$
       with Eq. (2.15).
9     Update $\boldsymbol{\theta}_k \to \boldsymbol{\theta}_{k+1}$ with optimizer.
```

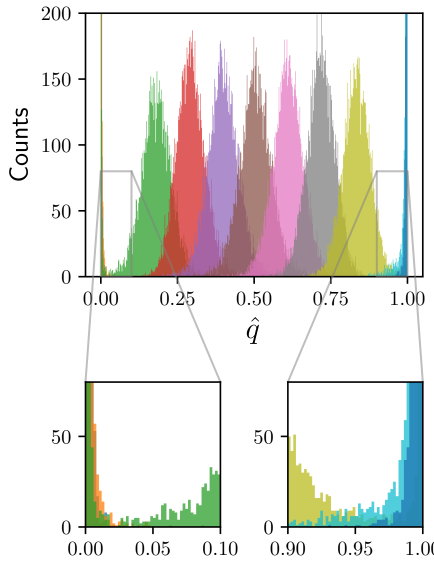


Figure 2: (Left) Pseudo-code corresponding to the BKE–US method. Lines 2-6 are the sampling steps, Lines 7-9 are the optimization steps, and a feedback loop couples the sampling and optimization steps together. Note that the sampling of configuration $\mathbf{x}_m^\alpha$ in Line 4 utilizes a fixed simulation 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. 9(b, top) for the corresponding histograms in configuration space.

Second, a formula for $\nabla_{\boldsymbol{\theta}} \hat{L}$ is needed for the optimizer and is obtained using a reweighting procedure [32] to compute the unbiased sample averages from biased samples. This yields

$$\nabla_{\boldsymbol{\theta}} \hat{L}\left(\boldsymbol{\theta}_k; \{(\mathcal{M}_k^\alpha, z_\alpha)\}, \mathcal{A}_k, \mathcal{B}_k\right) = \frac{\sum_{\alpha=1}^{M} \frac{z_\alpha}{|\mathcal{M}_k^\alpha|} \sum_{\mathbf{x} \in \mathcal{M}_k^\alpha} \left[\frac{\nabla_{\boldsymbol{\theta}} \ell(\mathbf{x}; \boldsymbol{\theta}_k)}{c(\mathbf{x}; \boldsymbol{\theta}_k)}\right]}{\sum_{\alpha=1}^{M} \frac{z_\alpha}{|\mathcal{M}_k^\alpha|} \sum_{\mathbf{x} \in \mathcal{M}_k^\alpha} \left[\frac{1}{c(\mathbf{x}; \boldsymbol{\theta}_k)}\right]} + \frac{\lambda_{\mathrm{A}}}{|\mathcal{A}_k|} \sum_{\mathbf{x} \in \mathcal{A}_k} \nabla_{\boldsymbol{\theta}} \ell_{\mathrm{A}}(\mathbf{x}; \boldsymbol{\theta}_k) + \frac{\lambda_{\mathrm{B}}}{|\mathcal{B}_k|} \sum_{\mathbf{x} \in \mathcal{B}_k} \nabla_{\boldsymbol{\theta}} \ell_{\mathrm{B}}(\mathbf{x}; \boldsymbol{\theta}_k)\,, \tag{2.15}$$

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, $z_\alpha$ is a reweighting factor given by the relative partition function

$$z_\alpha = \frac{Z_\alpha}{\sum_{\alpha'=1}^{M} Z_{\alpha'}} = \frac{\int \mathrm{d}\mathbf{x}\, e^{-\beta[V(\mathbf{x}) + W_\alpha(\mathbf{x}; \boldsymbol{\theta})]}}{\sum_{\alpha'=1}^{M} \int \mathrm{d}\mathbf{x}\, e^{-\beta[V(\mathbf{x}) + W_{\alpha'}(\mathbf{x}; \boldsymbol{\theta})]}}\,, \tag{2.16}$$

where $Z_\alpha$ is the partition function of the $\alpha$-th replica. Given the batches of samples $\{\mathcal{M}_k^\alpha\}_{\alpha=1}^{M}$, various free-energy methods [33] can be used to compute $Z_\alpha$ via the free-energy $F_\alpha = -\frac{1}{\beta} \ln Z_\alpha$. In this work, we use free-energy perturbation (FEP) [34] where the estimator for $z_\alpha$ is derived from the following exact identity:

$$\frac{z_\alpha}{z_{\alpha'}} = e^{-\beta \Delta F_{\alpha,\alpha'}} = \left\langle \frac{\phi_\alpha(\mathbf{x}; \boldsymbol{\theta})}{\phi_{\alpha'}(\mathbf{x}; \boldsymbol{\theta})} \right\rangle_{\alpha'}\,, \tag{2.17}$$

where $\langle \dots \rangle_{\alpha'}$ is an ensemble average over the distribution $\rho_{\alpha'} \propto e^{-\beta[V(\mathbf{x}) + W_{\alpha'}(\mathbf{x}; \boldsymbol{\theta})]}$ obeyed by the $\alpha'$-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 $\alpha'$-th replica, Eq. (2.17) can be estimated as

$$\frac{z_\alpha}{z_{\alpha'}} \approx \frac{1}{|\mathcal{M}_k^{\alpha'}|} \sum_{\mathbf{x}\in\mathcal{M}_k^{\alpha'}} \frac{\phi_\alpha(\mathbf{x};\boldsymbol{\theta}_k)}{\phi_{\alpha'}(\mathbf{x};\boldsymbol{\theta}_k)}\,. \tag{2.18}$$

The accuracy of Eq. (2.18) quickly deteriorates if samples obtained between the $\alpha$-th and $\alpha'$-th replicas do not overlap [35]. To mitigate this issue, we can employ a strategy called stratification [36], where the forward and backward free-energy differences per Eq. (2.18) between adjacent replicas are used to compute the overall free-energy difference of replica $\alpha$ in reference to replica $\gamma$. This strategy yields the following formula:

$$z_\alpha = \frac{z^\star_\alpha}{\sum_{\alpha=1}^{M} z^\star_\alpha}; \quad z^\star_\alpha = \begin{cases} \prod_{i=\gamma}^{\alpha-1} e^{-\beta\Delta F_{(i+1),i}} \approx \prod_{i=\gamma}^{\alpha-1} \left( \frac{1}{|\mathcal{M}_k^i|} \sum_{\mathbf{x}\in\mathcal{M}_k^i} \frac{\phi_{i+1}(\mathbf{x};\boldsymbol{\theta}_k)}{\phi_i(\mathbf{x};\boldsymbol{\theta}_k)} \right) & \alpha > \gamma \\ \prod_{i=\alpha}^{\gamma-1} e^{-\beta\Delta F_{(i-1),i}} \approx \prod_{i=\alpha}^{\gamma-1} \left( \frac{1}{|\mathcal{M}_k^i|} \sum_{\mathbf{x}\in\mathcal{M}_k^i} \frac{\phi_{i-1}(\mathbf{x};\boldsymbol{\theta}_k)}{\phi_i(\mathbf{x};\boldsymbol{\theta}_k)} \right) & \alpha < \gamma \\ 1 & \alpha = \gamma \end{cases}, \tag{2.19}$$

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 1 (Fig. 2, left). Note that Ref. [1] 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. (2.15). Additionally, Ref. [1] 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 $z_\alpha$. 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 [36]. One may choose as initial guess $q_\alpha = (\alpha-1)/(M-1)$ with equal biasing strengths, which is the setting recommended in Ref. [1], 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 2(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. 9(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.

### 2.3 Adding Elements of Supervised Learning

To begin with, the accuracy of the BKE–US method (Algorithm 1) 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 [37–39]. 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

**Algorithm 2:** 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. (2.14).
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. (2.20).
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. (2.19).
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. (2.15) and (2.24).
12     Update $\boldsymbol{\theta}_k \to \boldsymbol{\theta}_{k+1}$ with optimizer.
```

Figure 3: Pseudo-code for the BKE–US+SL method.

configuration $\mathbf{x}$. The quantity $q_{\mathrm{emp}}(\mathbf{x})$ can be obtained from a sample-mean estimator of Eq. (2.1):

$$q_{\mathrm{emp}}(\mathbf{x}) = \frac{1}{H} \sum_{i=1}^{H} h_B\left(\mathbf{x}_\tau;\ \mathbf{x}_0 = \mathbf{x}\right) , \tag{2.20}$$

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 $\tau$. This estimator obeys the binomial distribution and its variance scales as $\frac{1}{H}$ [14]. 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, which will be introduced in Section 2.4, 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 $q_{\mathrm{emp}}(\mathbf{x})$. To this end, a loss function in supervised learning is typically postulated as the squared error for every configuration $\mathbf{x}$:

$$\ell_{\mathrm{MSE}}(q_{\mathrm{emp}}, \mathbf{x}; \boldsymbol{\theta}) = \frac{1}{2}\left(\hat{q}(\mathbf{x}; \boldsymbol{\theta}) - q_{\mathrm{emp}}\right)^2 . \tag{2.21}$$

Suppose that $q_{\mathrm{emp}}(\mathbf{x})$ is computed from configurations sampled by different replicas during importance sampling. For every $\alpha$-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. Denoting the collection of batches as $\{\mathcal{C}^\alpha\}_{\alpha=1}^{M}$, and given Eq. (2.21), the objective function as a mean-squared error has the form

$$\hat{L}_{\mathrm{MSE}}(\boldsymbol{\theta}; \{\mathcal{C}^\alpha\}) = \frac{\lambda_{\mathrm{MSE}}}{M} \sum_{\alpha=1}^{M} \frac{1}{|\mathcal{C}^\alpha|} \sum_{(q_{\mathrm{emp}}, \mathbf{x}) \in \mathcal{C}^\alpha} \ell_{\mathrm{MSE}}(q_{\mathrm{emp}}, \mathbf{x}; \boldsymbol{\theta}) , \tag{2.22}$$

where $\lambda_{\mathrm{MSE}}$ 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. (2.15).

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. (2.22) 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 $\alpha$-th replica:

$$\ell_{\mathrm{ME}}(\mathcal{C}^\alpha;\boldsymbol{\theta}) = \frac{1}{2}\left[\frac{1}{|\mathcal{C}^\alpha|}\sum_{(q_{\mathrm{emp}},\mathbf{x})\in\mathcal{C}^\alpha}\left(\hat{q}(\mathbf{x};\boldsymbol{\theta}) - q_{\mathrm{emp}}\right)\right]^2 . \tag{2.23}$$

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

$$\hat{L}_{\mathrm{SL}}(\boldsymbol{\theta};\{\mathcal{C}^\alpha\}) = \frac{\lambda_{\mathrm{SL}}}{M}\sum_{\alpha=1}^{M}\ell_{\mathrm{ME}}(\mathcal{C}^\alpha;\boldsymbol{\theta}) , \tag{2.24}$$

where $\lambda_{\mathrm{SL}}$ is the penalty strength. Equation (2.24) 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. (2.22)) and modified (Eq. (2.24)) objective functions for a two-dimensional system can be found in Appendix B.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 $\alpha$. 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 $\tau_{\mathrm{emp}}$ iterations to reduce the computational cost. The pseudocode combining supervised learning with the BKE–US method is described in Algorithm 1 (Fig. 3), and is herein referred to as the BKE–US+SL method.

## 2.4 Replacing Feedback Loops 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 $z_\alpha$, 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 an algorithm called the finite-temperature string (FTS) method [12, 13], 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. In what follows, we review the FTS method in Section 2.4.1 and describe new algorithms for solving the BKE in Section 2.4.2; see also Ref. [13] for additional details on the FTS method. Readers who are familiar with the FTS method may skip Section 2.4.1 and read Section 2.4.2 directly for details on solving the BKE with the FTS method.

### 2.4.1 Review of the Finite-Temperature String Method

The FTS method is an algorithm for obtaining a transition path $\boldsymbol{\varphi}(s)$, as defined in Eq. (2.4), using sampling and optimization techniques. It emerges from an approximation of the committor function $q(\mathbf{x})$, which 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

$$q(\mathbf{x}) \approx f(s_\gamma(\mathbf{x})) , \tag{2.25}$$

$$s_\gamma(\mathbf{x}) \equiv \arg\min_{s\in[0,L]} \frac{1}{2}|\mathbf{x} - \boldsymbol{\varphi}(s)|^2 , \tag{2.26}$$

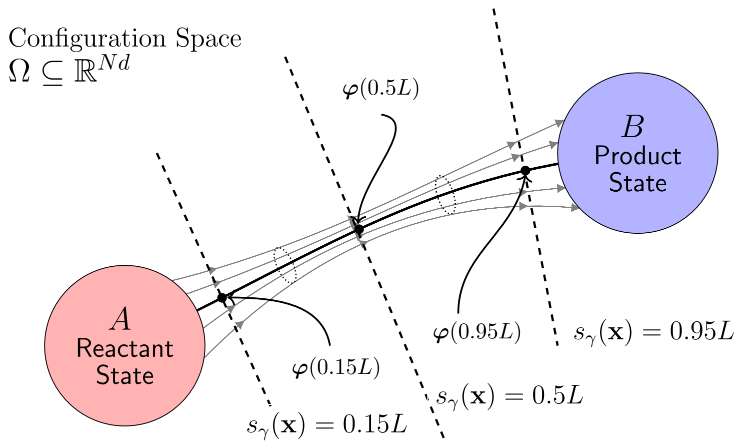

Configuration Space
$\Omega \subseteq \mathbb{R}^{Nd}$
$\boldsymbol{\varphi}(0.5L)$
$B$
Product State
$A$
Reactant State
$\boldsymbol{\varphi}(0.15L)$
$\boldsymbol{\varphi}(0.95L)$
$s_\gamma(\mathbf{x}) = 0.95L$
$s_\gamma(\mathbf{x}) = 0.5L$
$s_\gamma(\mathbf{x}) = 0.15L$

Figure 4: 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 $\frac{\mathrm{d}\boldsymbol{\varphi}(s)}{\mathrm{d}s}$.

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. (2.26) to obtain the following equation:

$$\frac{\mathrm{d}\boldsymbol{\varphi}(s)}{\mathrm{d}s} \cdot (\mathbf{x} - \boldsymbol{\varphi}(s)) = 0\,, \tag{2.27}$$

which is a linear equation in $\mathbf{x}$, indicating the set of all configurations satisfying Eq. (2.27) for fixed value of $s \in [0, L]$ is a hyperplane; see Fig. 4 for illustration. On the other hand, the operation of fixing a configuration $\mathbf{x}$, and finding $s$ that satisfies Eq. (2.27) defines a mapping between configurations $\mathbf{x} \in \Omega$ and the variable $s \in [0, L]$. This mapping is what we denote as $s_\gamma(\mathbf{x})$.

Given $s_\gamma(\mathbf{x})$ in Eq. (2.26), the problem of finding $\boldsymbol{\varphi}(s)$ can be posed as an optimization problem. To this end, using Eq. (2.25), Eq. (2.4) can be approximated as an integral over the hyperplane defined by $s_\gamma(\mathbf{x})$:

$$\boldsymbol{\varphi}(s) \approx \frac{\int_{\tilde{P}} \mathrm{d}S \rho(\mathbf{x}) f'(s_\gamma(\mathbf{x})) |\nabla_{\mathbf{x}} s_\gamma(\mathbf{x})| \mathbf{x}}{\int_{\tilde{P}} \mathrm{d}S \rho(\mathbf{x}) f'(s_\gamma(\mathbf{x})) |\nabla_{\mathbf{x}} s_\gamma(\mathbf{x})|}\,, \tag{2.28}$$

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. (2.28) can be rewritten as

$$\boldsymbol{\varphi}(s) \approx \frac{\int_{\tilde{P}} \mathrm{d}S \rho(\mathbf{x}) |\nabla_{\mathbf{x}} s_\gamma(\mathbf{x})| \mathbf{x}}{\int_{\tilde{P}} \mathrm{d}S \rho(\mathbf{x}) |\nabla_{\mathbf{x}} s_\gamma(\mathbf{x})|}\,. \tag{2.29}$$

Using the identity [40]

$$\int_{\tilde{P}} \mathrm{d}S = \int_{\Omega} \mathrm{d}\mathbf{x} \delta(s_\gamma(\mathbf{x}) - s) |\nabla_{\mathbf{x}} s_\gamma(\mathbf{x})|\,, \tag{2.30}$$

with $\delta(s_\gamma(\mathbf{x}) - s)$ as the Dirac delta function, Eq. (2.29) can be rewritten as

$$\boldsymbol{\varphi}(s) \approx \frac{\int_{\Omega} \mathrm{d}\mathbf{x} \rho(\mathbf{x}) \delta(s_\gamma(\mathbf{x}) - s) |\nabla_{\mathbf{x}} s_\gamma(\mathbf{x})|^2 \mathbf{x}}{\int_{\Omega} \mathrm{d}\mathbf{x} \rho(\mathbf{x}) \delta(s_\gamma(\mathbf{x}) - s) |\nabla_{\mathbf{x}} s_\gamma(\mathbf{x})|^2} = \frac{\langle \delta(s_\gamma(\mathbf{x}) - s) |\nabla_{\mathbf{x}} s_\gamma(\mathbf{x})|^2 \mathbf{x} \rangle}{\langle \delta(s_\gamma(\mathbf{x}) - s) |\nabla_{\mathbf{x}} s_\gamma(\mathbf{x})|^2 \rangle}\,. \tag{2.31}$$

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. [13] for a proof), Eq. (2.31) can be simplified into a conditional average given by

$$\boldsymbol{\varphi}(s) \approx \frac{\langle \delta(s_\gamma(\mathbf{x}) - s) \mathbf{x} \rangle}{\langle \delta(s_\gamma(\mathbf{x}) - s) \rangle} = \langle \mathbf{x} \mid s_\gamma(\mathbf{x}) = s \rangle\,. \tag{2.32}$$

Lastly, one may use variational techniques to show that Eq. (2.32) is the result of extremizing the following functional [13, 41]:

$$C[\boldsymbol{\varphi}] = \int_0^L \mathrm{d}s \left\langle \frac{1}{2} |\boldsymbol{\varphi}(s) - \mathbf{x}|^2 \delta(s_\gamma(\mathbf{x}) - s) \right\rangle \tag{2.33}$$

such that

$$\left|\frac{\mathrm{d}\boldsymbol{\varphi}(s)}{\mathrm{d}s}\right| = 1 \,. \tag{2.34}$$

Equation (2.34) is the definition of arc-length parameterization, which sets a constraint on the possible paths that extremize Eq. (2.33).

Equations (2.33) and (2.34) 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. (2.34), i.e., $|\boldsymbol{\varphi}^{\alpha+1} - \boldsymbol{\varphi}^\alpha| = |\boldsymbol{\varphi}^\alpha - \boldsymbol{\varphi}^{\alpha-1}|$, $\forall\alpha \in \{1,\dots,M\}$, Eq. (2.33) can be approximated as

$$C(\{\boldsymbol{\varphi}^\alpha\}) = \sum_{\alpha=1}^{M} \Delta s \left\langle \frac{1}{2}|\boldsymbol{\varphi}^\alpha - \mathbf{x}|^2 \delta(s_\gamma(\mathbf{x}) - s_\alpha) \right\rangle \,, \tag{2.35}$$

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 $\Delta s$ 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. [13]):

$$h_{R_\alpha}(\mathbf{x}) = \begin{cases} \frac{1}{\Delta s} & \mathbf{x} \in R_\alpha(\{\boldsymbol{\varphi}^\alpha\}) = \{\mathbf{x} \in \Omega : |\mathbf{x} - \boldsymbol{\varphi}^\alpha| < |\mathbf{x} - \boldsymbol{\varphi}^{\alpha'}| \;\; \forall \alpha' \neq \alpha\} \\ 0 & \text{otherwise} \end{cases} \,, \tag{2.36}$$

where $R_\alpha$ denotes a Voronoi cell centered at node $\boldsymbol{\varphi}^\alpha$. With these steps, Eq. (2.35) can then be expressed as a least-squares function:

$$C(\{\boldsymbol{\varphi}^\alpha\}) = \sum_{\alpha=1}^{M} \Delta s \left\langle \frac{1}{2}|\boldsymbol{\varphi}^\alpha - \mathbf{x}|^2 h_{R_\alpha}(\mathbf{x}) \right\rangle = \sum_{\alpha=1}^{M} \left\langle \frac{1}{2}|\boldsymbol{\varphi}^\alpha - \mathbf{x}|^2 \right\rangle_{R_\alpha(\{\boldsymbol{\varphi}^\alpha\})} \tag{2.37}$$

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

The ensemble averages in Eq. (2.37) 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. 5(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:

$$\mathbf{x}_\star^\alpha = \mathbf{x}_t^\alpha - \gamma^{-1}\nabla_{\mathbf{x}} V(\mathbf{x}_t^\alpha)\Delta t + \sqrt{2\Delta t k_{\mathrm{B}} T \gamma^{-1}}\,\mathbf{w}_t^\alpha \,, \tag{2.38}$$

$$\mathbf{x}_{t+1}^\alpha = \begin{cases} \mathbf{x}_\star^\alpha & \text{if } \mathbf{x}_\star^\alpha \in R_\alpha \\ \mathbf{x}_t^\alpha & \text{otherwise} \end{cases} \,, \tag{2.39}$$

where $\mathbf{w}_t^\alpha$ is a random variable with zero-mean and unit variance. Note that Eq. (2.38) can be replaced with an MC step. Introducing $\mathcal{R}^\alpha$ as the batch of samples obtained from the $\alpha$-th replica, Eq. (2.37) can be estimated as

$$\hat{C}(\{\boldsymbol{\varphi}^\alpha\}; \{\mathcal{R}^\alpha\}) = \sum_{\alpha=1}^{M} \frac{1}{|\mathcal{R}^\alpha|} \sum_{\mathbf{x}\in\mathcal{R}^\alpha} \frac{1}{2}|\boldsymbol{\varphi}^\alpha - \mathbf{x}|^2 \,. \tag{2.40}$$

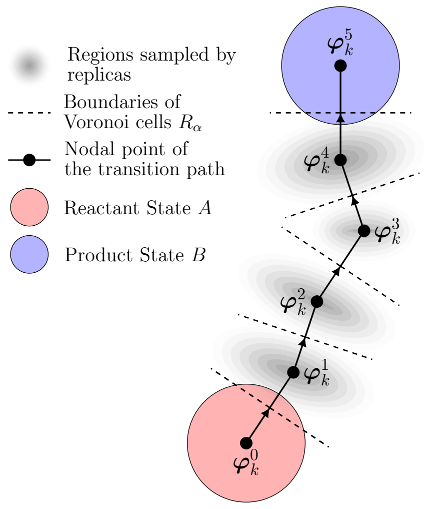
To avoid large displacements in neighboring nodal points, a penalty function is added to Eq. (2.40), which yields

$$\hat{C}(\{\boldsymbol{\varphi}^\alpha\}; \{\mathcal{R}^\alpha\}) = \sum_{\alpha=1}^{M} \frac{1}{|\mathcal{R}^\alpha|} \sum_{\mathbf{x}\in\mathcal{R}^\alpha} \frac{1}{2}|\boldsymbol{\varphi}^\alpha - \mathbf{x}|^2 + \frac{\lambda_{\mathrm{S}}}{2} \sum_{\alpha=1}^{M-1} |\boldsymbol{\varphi}^{\alpha+1} - \boldsymbol{\varphi}^\alpha|^2 \,, \tag{2.41}$$

$$\text{s.t.} \quad |\boldsymbol{\varphi}^{\alpha+1} - \boldsymbol{\varphi}^\alpha| = |\boldsymbol{\varphi}^\alpha - \boldsymbol{\varphi}^{\alpha-1}| \,, \tag{2.42}$$

where $\lambda_{\mathrm{S}}$ is the penalty strength.

The FTS method minimizes Eq. (2.41) using a closed feedback loop between the replica dynamics, e.g., Eqs. (2.38) and (2.39), 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 MD/MC



**Algorithm 3:** The BKE–FTS(ME) Method

```
Data: Initial conditions $\boldsymbol{\theta}_0$, $\{\varphi_0^\alpha\}$. 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 $k = 0, \ldots, K$ do
2      for $\alpha = 1, \ldots, M$ in parallel do
3          for $m = 1, \ldots, |\mathcal{R}_k^\alpha|$ do
4              Sample $\mathbf{x}_m^\alpha$ with MD/MC simulation
                constrained in the Voronoi cell
                $R_\alpha(\{\varphi_k^\alpha\})$, e.g., Eqs. (2.38)
                and (2.39).
5              Store $\mathbf{x}_m^\alpha$ in batch $\mathcal{R}_k^\alpha$.
6      $\varphi_\alpha^{k+1} \leftarrow$ Eqs. (2.43) and (2.44).
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. (2.48).
9      Compute $\nabla_{\boldsymbol{\theta}} \hat{L}(\boldsymbol{\theta}_k; \{(\mathcal{R}_k^\alpha, z_\alpha)\}, \mathcal{A}_k, \mathcal{B}_k)$
        with Eq. (2.46).
10     Update $\boldsymbol{\theta}_k \to \boldsymbol{\theta}_{k+1}$ with optimizer.
```

Figure 5: (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.

trajectory run from the $\alpha$-th replica. This data is then used in a two-part gradient descent update, where the first part corresponds to the following update:

$$\varphi_\star^\alpha = \varphi_k^\alpha - \Delta\tau \nabla_{\varphi^\alpha} \hat{C}(\{\varphi_k^\alpha\}; \{\mathcal{R}_k^\alpha\}), \tag{2.43}$$

with $\Delta\tau$ the step size. Note that one can replace Eq. (2.43) with an implicit update for increased stability or a momentum-variant, such as the Heavy-Ball [30] and the Nesterov method [42], for accelerated convergence. The second part enforces the constraint Eq. (2.42) with a reparameterization of the path using linear interpolation:

$$\varphi_{k+1}^\alpha = \varphi_\star^{a(\alpha)-1} + \left( L_M \frac{\alpha - 1}{M - 1} - L_{a(\alpha)-1} \right) \frac{\varphi_\star^{a(\alpha)} - \varphi_\star^{a(\alpha)-1}}{\left| \varphi_\star^{a(\alpha)} - \varphi_\star^{a(\alpha)-1} \right|}, \tag{2.44}$$

where $L_\alpha = \sum_{\alpha'=2}^{\alpha} |\varphi_\star^{\alpha'} - \varphi_\star^{\alpha'-1}|$ is the length of the path up to node $\varphi_\star^\alpha$, 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 $\varphi(s)$.

### 2.4.2 Solving the BKE with the Finite-Temperature String Method

With the FTS method described in Section 2.4.1, we now proceed to construct new algorithms for minimizing the loss in Eq. (2.13). 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 loss functions and their gradients. 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.

**The FTS method with master equation:** The first algorithm that we construct involves updating the transition path, represented as a set of nodal points, 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. (2.43)–(2.44), as well as the neural network parameters $\boldsymbol{\theta}_k$ by computing the gradient of the loss in Eq. (2.13). 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. (2.15)–(2.16), where the bias potentials $W_\alpha$ are replaced with hard-wall potentials constraining each replica to its Voronoi cell, i.e.,

$$W_\alpha(\mathbf{x};\{\boldsymbol{\varphi}^\alpha\}) = \begin{cases} 0 & \mathbf{x}\in R_\alpha \\ \infty & \text{otherwise} \end{cases} . \tag{2.45}$$

This yields

$$\begin{aligned}\nabla_{\boldsymbol{\theta}}\hat{L}\left(\boldsymbol{\theta}_k;\{(\mathcal{R}_k^\alpha, z_\alpha)\},\mathcal{A}_k,\mathcal{B}_k\right) = &\sum_{\alpha=1}^M \frac{z_\alpha}{|\mathcal{R}_k^\alpha|}\sum_{\mathbf{x}\in\mathcal{R}_k^\alpha}\nabla_{\boldsymbol{\theta}}\ell(\mathbf{x};\boldsymbol{\theta}_k) + \frac{\lambda_{\mathrm{A}}}{|\mathcal{A}_k|}\sum_{\mathbf{x}\in\mathcal{A}_k}\nabla_{\boldsymbol{\theta}}\ell_{\mathrm{A}}(\mathbf{x};\boldsymbol{\theta}_k) \\ &+ \frac{\lambda_{\mathrm{B}}}{|\mathcal{B}_k|}\sum_{\mathbf{x}\in\mathcal{B}_k}\nabla_{\boldsymbol{\theta}}\ell_{\mathrm{B}}(\mathbf{x};\boldsymbol{\theta}_k)\,,\end{aligned} \tag{2.46}$$

where the reweighting factors $z_\alpha$ are

$$z_\alpha = \frac{\int_{R_\alpha}\mathrm{d}\mathbf{x}\, e^{-\beta V(\mathbf{x})}}{\int_{\bigcup_{\alpha=1}^M R_\alpha}\mathrm{d}\mathbf{x}\, e^{-\beta V(\mathbf{x})}} = \frac{\int_{R_\alpha}\mathrm{d}\mathbf{x}\, e^{-\beta V(\mathbf{x})}}{\int_\Omega \mathrm{d}\mathbf{x}\, e^{-\beta V(\mathbf{x})}} = \int_{R_\alpha}\mathrm{d}\mathbf{x}\,\rho(\mathbf{x})\,. \tag{2.47}$$

Equation (2.47) indicates $z_\alpha$ is the equilibrium probability of finding $\mathbf{x}$ to be in a Voronoi cell $R_\alpha$. 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 [13]. To this end, let $N_{\alpha\alpha'}$ be the number of times that the $\alpha$-th replica attempts to exit its Voronoi cell $R_\alpha$ 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. (2.38)–(2.39). Let $k_{\alpha\alpha'}$ be the rate at which the system transitions between $R_\alpha$ to $R_{\alpha'}$. Denoting $N_{\text{steps}}^\alpha$ as the total simulation length of the $\alpha$-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 $R_\alpha$:

$$\sum_{\alpha'=1}^M z_{\alpha'}k_{\alpha'\alpha} = \sum_{\alpha'=1}^M z_\alpha k_{\alpha\alpha'}, \quad \forall\alpha\in\{1,\dots,M\}\,, \tag{2.48}$$

which can be solved to obtain $z_\alpha$; see Appendix A for more details, and also Section III of Ref. [43] for a more detailed discussion of Eq. (2.48). Equations (2.46) and (2.48) constitute the new algorithm, and will herein be referred to as the BKE–FTS(ME) method, whose pseudocode is described in Algorithm 3 (Fig. 5, right).

**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. 9(b)), which better ensures low-variance estimation from reweighting. Accuracy can also be improved by running longer simulations, i.e., larger $N_{\text{steps}}^\alpha$, since they lead to more accurate estimates of the rates $k_{\alpha\alpha'}$, thereby reducing the error in the estimated reweighting factor $z_\alpha$. Despite this, the error in $z_\alpha$ 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, $z_\alpha$ computed from umbrella sampling is amenable to error analysis [32, 44], 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 $z_\alpha$ is based on umbrella sampling and FEP (Eq. (2.19)). 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

**Algorithm 4:** 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. (2.14) and Eq. (2.49).
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. (2.19).
8     Compute $\nabla_{\boldsymbol{\theta}} \hat{L}(\boldsymbol{\theta}_k; \{(\mathcal{M}_k^\alpha, z_\alpha)\}, \mathcal{A}_k, \mathcal{B}_k)$ with Eq. (2.15).
9     Update $\boldsymbol{\theta}_k \to \boldsymbol{\theta}_{k+1}$ with optimizer.
```

Figure 6: Pseudo-code for the BKE–FTS(US) method.

**Algorithm 5:** 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. (2.38)–(2.39).
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. (2.20).
8              Store $(q_\mathrm{emp}, \mathbf{x}^\alpha)$ in batch $\mathcal{C}^\alpha$.
9      $\boldsymbol{\varphi}_\alpha^{k+1} \leftarrow$ Eqs. (2.43)–(2.44).
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. (2.48).
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. (2.24) and (2.46).
13     Update $\boldsymbol{\theta}_k \to \boldsymbol{\theta}_{k+1}$ with optimizer.
```

Figure 7: Pseudo-code for the BKE–FTS(ME)+SL method.

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 $\alpha$-th replica can be written as

$$W_\alpha(\mathbf{x}; \{\boldsymbol{\varphi}^\alpha\}) = \frac{1}{2}\kappa_\alpha^\parallel (\mathbf{x} - \boldsymbol{\varphi}^\alpha)\mathbf{P}_\alpha^\parallel (\mathbf{x} - \boldsymbol{\varphi}^\alpha) + \frac{1}{2}\kappa_\alpha^\perp (\mathbf{x} - \boldsymbol{\varphi}^\alpha)\mathbf{P}_\alpha^\perp (\mathbf{x} - \boldsymbol{\varphi}^\alpha)\,, \tag{2.49}$$

where $\kappa_\alpha^\parallel$ and $\kappa_\alpha^\perp$ are the bias strengths for the parallel and transverse direction, respectively. 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. [45] 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. (2.15) and Eq. (2.19) 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

**Algorithm 6:** 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. (2.14) and (2.49).
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. (2.20).
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. (2.19).
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. (2.15) and (2.24).
12     Update $\boldsymbol{\theta}_k \to \boldsymbol{\theta}_{k+1}$ with optimizer.
```

Figure 8: Pseudo-code for the BKE–FTS(US)+SL method.

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 4 (Fig. 6). 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 $z_\alpha$ estimated from FEP are amenable to error analysis [44]. As shown later in Section 3.3, we provide an error analysis of the 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.

**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 2.3 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 2.3, it can be evaluated at a rate $\tau_{\mathrm{emp}}$ 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. (2.24) 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 5 (Fig. 7) and Algorithm 6 (Fig. 8), respectively.

**Limitations of the FTS Method:** The proposed methods for solving the BKE with the FTS method inherit the limitations of the FTS method itself. 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. [13] proposes a sufficiently general CV, denoted as $\Theta$, 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

$$\Theta(\mathbf{x}_\mathrm{S};\mathbf{R}^*,\mathbf{b}^*) = \mathbf{R}^*\left(\mathbf{x}_\mathrm{S}-\mathbf{b}^*\right)\,, \tag{2.50}$$

$$(\mathbf{R}^*,\mathbf{b}^*) = \underset{(\mathbf{R},\mathbf{b})}{\arg\min}\left|\mathbf{R}\left(\mathbf{x}_\mathrm{S}-\mathbf{b}\right)-\boldsymbol{\varphi}^\alpha\right|\,, \tag{2.51}$$

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. (2.51), 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. (2.51) 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 [46]. 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. (2.50), it may not be sufficient at high densities where the solvent molecules/particles move in a highly correlated fashion during the transition, i.e., solvent reorganization. In this situation, the BKE–FTS methods can still use the FTS method with the CV as given in Eq. (2.50) to train neural networks that are implicitly aware of the solvent reorganization, 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. (2.50) is used in Section 4 to compute committor functions and reaction rates in a solvated dimer system with relatively high accuracy.

The FTS method is also ill-suited for problems involving multiple reaction pathways. This problem can possibly be addressed by evolving multiple independent strings that are repulsive with respect to each other, as is done in an extension of the string method in the CV space termed the climbing multistring method [47], but it remains to be extended to the FTS method. Other methods more amendable to studying processes with multiple reaction pathways, such as Markov State Models [48–50], could also be considered in future work.

## 3 Computational Studies in Low-Dimensional Systems

In this section, we test Algorithms 1–6 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 [39]:

$$\hat{q}(\mathbf{x};\boldsymbol{\theta}=\{\mathbf{W}_1,\mathbf{w}_2,\mathbf{b}\}) = \sigma\left(\mathbf{w}_2\cdot\mathrm{ReLU}(\mathbf{W}_1\mathbf{x}+\mathbf{b}_1)\right)\,, \tag{3.1}$$

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 [30] and Adam [31] for the 1D and 2D system, respectively; see Appendix B 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

$$I(\boldsymbol{\theta};\{\mathbf{x}_0^\alpha\}) = \frac{1}{M}\sum_{\alpha=1}^{M}\left(\hat{q}(\mathbf{x}_0^\alpha;\boldsymbol{\theta})-\frac{\alpha-1}{M-1}\right)^2\,, \tag{3.2}$$

where a gradient descent algorithm is used with a stepsize of $0.001$ until $I(\boldsymbol{\theta})\leq 10^{-3}$. Here, $\mathbf{x}_0^\alpha$ is the initial configuration of the $\alpha$-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 $\mathbf{x}_0^\mathrm{B}$:

$$\mathbf{x}_0^\alpha = \left(1-\frac{\alpha-1}{M-1}\right)\mathbf{x}_0^\mathrm{A}+\left(\frac{\alpha-1}{M-1}\right)\mathbf{x}_0^\mathrm{B}\,. \tag{3.3}$$

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. (3.2) 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.

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. (2.9). The latter is proportional to the reaction rate in Eq. (2.5). The $L_1$-norm error is defined over the region spanned by the transition tube, $T_\Lambda = \{\mathbf{x} \in \Omega : |\mathbf{J}(\mathbf{x})| \geq \Lambda\}$ where $\Lambda$ is a cut-off value, and normalized by the volume of the region. This yields

$$||\hat{q} - q||_1 = \frac{1}{\int_{T_\Lambda} \mathrm{d}\mathbf{x}} \int_{T_\Lambda} \mathrm{d}\mathbf{x} |\hat{q}(\mathbf{x};\boldsymbol{\theta}) - q(\mathbf{x})| \,. \tag{3.4}$$

In all algorithms, an on-the-fly estimate of the ensemble average of Eq. (2.9) is computed at the $k$-th iteration with the following formula:

$$\left\langle \frac{1}{2}|\nabla_{\mathbf{x}}\hat{q}(\mathbf{x};\boldsymbol{\theta}_{\mathrm{k}})|^2 \right\rangle_{\text{fly}} = \begin{cases} \dfrac{\sum_{\alpha=1}^{M} \dfrac{z_\alpha}{|\mathcal{M}_k^\alpha|} \sum_{\mathbf{x}\in\mathcal{M}_k^\alpha} \left[\dfrac{\ell(\mathbf{x};\boldsymbol{\theta}_k)}{c(\mathbf{x};\boldsymbol{\theta}_k)}\right]}{\sum_{\alpha=1}^{M} \dfrac{z_\alpha}{|\mathcal{M}_k^\alpha|} \sum_{\mathbf{x}\in\mathcal{M}_k^\alpha} \left[\dfrac{1}{c(\mathbf{x};\boldsymbol{\theta}_k)}\right]} & \text{for umbrella sampling} \\ \sum_{\alpha=1}^{M} \dfrac{z_\alpha}{|\mathcal{R}_k^\alpha|} \sum_{\mathbf{x}\in\mathcal{R}_k^\alpha} \ell(\mathbf{x};\boldsymbol{\theta}_k) & \text{for the master equation} \end{cases} , \tag{3.5}$$

where the reweighting factors $z_\alpha$ are evaluated using Eq. (2.19) for umbrella sampling and Eq. (2.48) for the master equation, respectively. The estimate in Eq. (3.5) is then compared to the average BKE loss function that is evaluated using reference solutions.

## 3.1 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

$$q_{\text{exact}}(\mathbf{x}) = \frac{\int_{-1}^{\mathbf{x}} \mathrm{d}\mathbf{x}' e^{15V(\mathbf{x}')}}{\int_{-1}^{1} \mathrm{d}\mathbf{x}' e^{15V(\mathbf{x}')}} \,. \tag{3.6}$$

Using Eq. (3.6), 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

$$\left\langle \frac{1}{2}|\nabla_{\mathbf{x}}q_{\text{exact}}(\mathbf{x})|^2 \right\rangle = \frac{1}{2}\left( Z \left( \int_{-1}^{1} \mathrm{d}\mathbf{x}\, e^{\beta V(\mathbf{x})} \right)\right)^{-1} \approx 10^{-6} \,. \tag{3.7}$$

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

Figure 9(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. 9(b), top). As discussed in Section 2.2, 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. 9(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. 9(c) shows that the on-the-fly estimates from the BKE–US and BKE–US+SL methods exhibit large 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

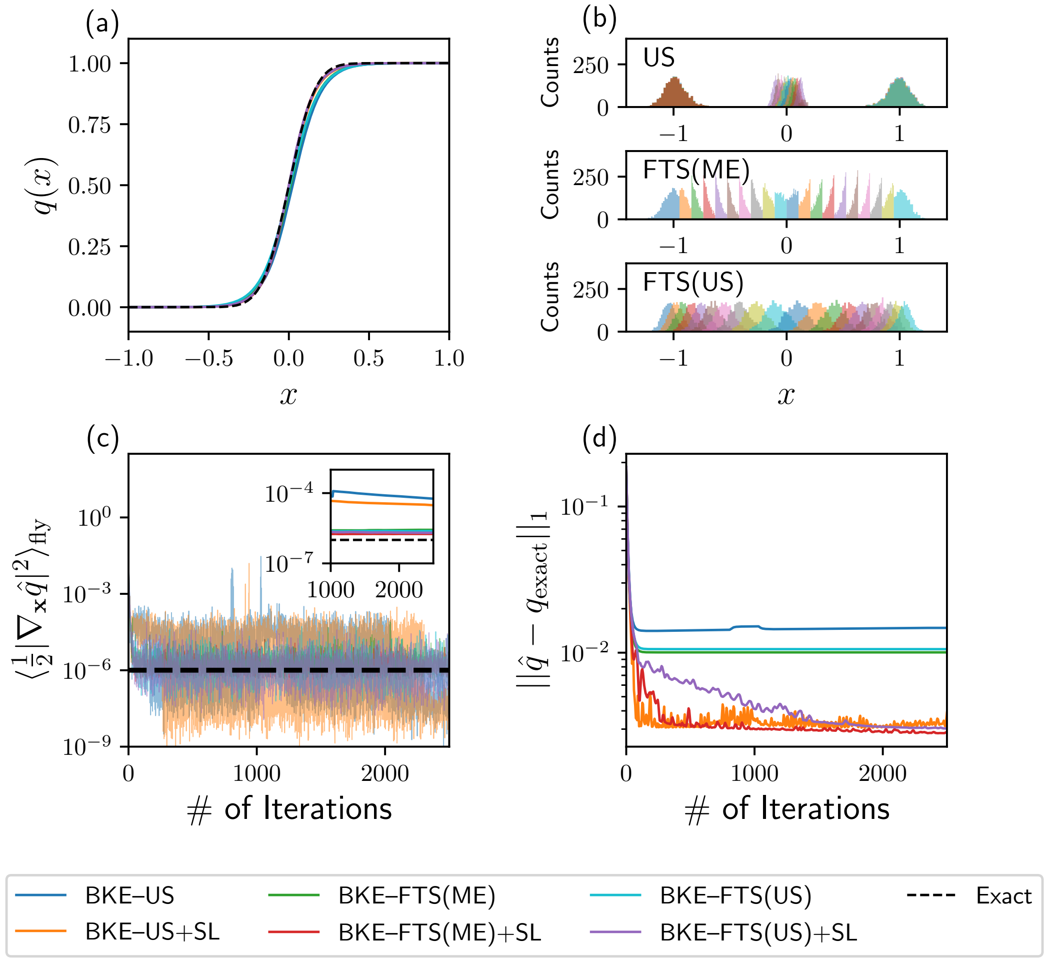


Figure 9: (a) Committor function obtained from all methods compared with the exact solution. (b) Histograms of samples obtained from the BKE–US method (top), the BKE–FTS(ME) method (middle), and the BKE–FTS(US) method (bottom). (c) On-the-fly estimates of the average BKE loss obtained at every iteration 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.

size. When these on-the-fly estimates are cumulatively averaged, as shown in the inset of Fig. 9(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. 9(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.

### 3.2 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

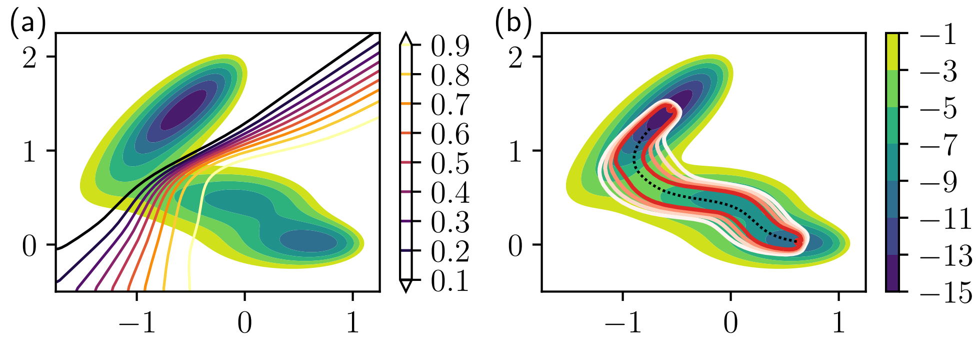


Figure 10: (a) MB potential along with isocommittor lines from the FEM solution. Note the MB contours correspond to $\beta V_{\mathrm{MB}}$. (b) MB potential along with contours indicating the lines of increasing flux $\mathbf{J}(\mathbf{x})$ from white to red along with the transition path in black, computed using the FEM solution via Eq. (2.4).

curved. To this end, we now study a particle subject to the 2D Müller-Brown (MB) potential [51], which is a Gaussian mixture potential given by

$$V_{\mathrm{MB}}(\mathbf{x}) = \sum_{k=1}^{4} A_k \exp\left(a_i\left(x-\bar{x}_i\right)^2 + b_i\left(x-\bar{x}_i\right)\left(y-\bar{y}_i\right) + c_i\left(y-\bar{y}\right)^2\right) , \tag{3.8}$$
$$A = (-200,-100,-170,15), \quad a = (-1,-1,-6.5,-0.7)\,,$$
$$b = (0,0,11,0.6), \quad c = (-10,-10,-6.5,0.7)\,,$$
$$\bar{x} = (1,0,-0.5,-1), \quad \bar{y} = (0,0.5,1.5,1)\,.$$

It has two minima at $\mathbf{x}_0^{\mathrm{A}} \approx (-0.558, 1.442)$ and $\mathbf{x}_0^{\mathrm{B}} \approx (0.623, 0.028)$. In what follows, we study this model at a temperature where $k_{\mathrm{B}}T = 10$. While an analytical form of $q(\mathbf{x})$ for the MB potential is unknown, we use FEM to numerically solve the BKE (Eq. (2.2)) via FEniCS [52, 53], and obtain a solution to the committor function $q_{\mathrm{FEM}}(\mathbf{x})$. This is done on the domain $\Omega = [-1.75, 1.25] \times [-0.5, 2.25]$, with the reactant and product states defined by $A = \{\mathbf{x} \in \Omega : |\mathbf{x} - \mathbf{x}_0^{\mathrm{A}}| < 0.025\}$ and $B = \{\mathbf{x} \in \Omega : |\mathbf{x} - \mathbf{x}_0^{\mathrm{B}}| < 0.025\}$, respectively. The FEM solution is obtained by applying Dirichlet boundary conditions as per Eq. (2.3) along with a zero-flux Neumann boundary condition on $\partial\Omega$, and a mesh of roughly $3 \cdot 10^5$ elements. Contours of the MB potential along with isocommittor lines of $q_{\mathrm{FEM}}(\mathbf{x})$ are shown in Fig. 10(a), along with contours of increasing flux and the transition path in Fig. 10(b).

The ensemble-averaged BKE loss with $q_{\mathrm{FEM}}(\mathbf{x})$ over $\Omega$ is obtained by evaluating the variational objective function in Eq. (2.7):

$$\left\langle \frac{1}{2}|\nabla_{\mathbf{x}} q_{\mathrm{FEM}}|^2 \right\rangle \approx 2.46 \cdot 10^{-4}\,. \tag{3.9}$$

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. 10(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

$$\left\langle \frac{1}{2}|\nabla_{\mathbf{x}} \hat{q}(\mathbf{x};\boldsymbol{\theta}_{\mathrm{k}})|^2 \right\rangle_{\mathrm{full}} = \int_\Omega \mathrm{d}\mathbf{x}\rho(\mathbf{x})\ell(\mathbf{x};\boldsymbol{\theta}_{\mathrm{k}}) = \langle \ell(\mathbf{x};\boldsymbol{\theta}_{\mathrm{k}}) \rangle\,. \tag{3.10}$$

Equation (3.10) provides an additional metric for evaluating accuracy; in particular, comparing Eq. (3.10) with the on-the-fly estimates allows us to evaluate the sampling error that arises from the choice of estimator, while comparing Eq. (3.10) with the FEM value (Eq. (3.9)) allows us to evaluate the error inherent to the neural network.

Figures 11(a-c) show the isocommittor lines and sampled configurations obtained from all algorithms. We see from the isocommittor lines that methods employing supervised learning elements improve the

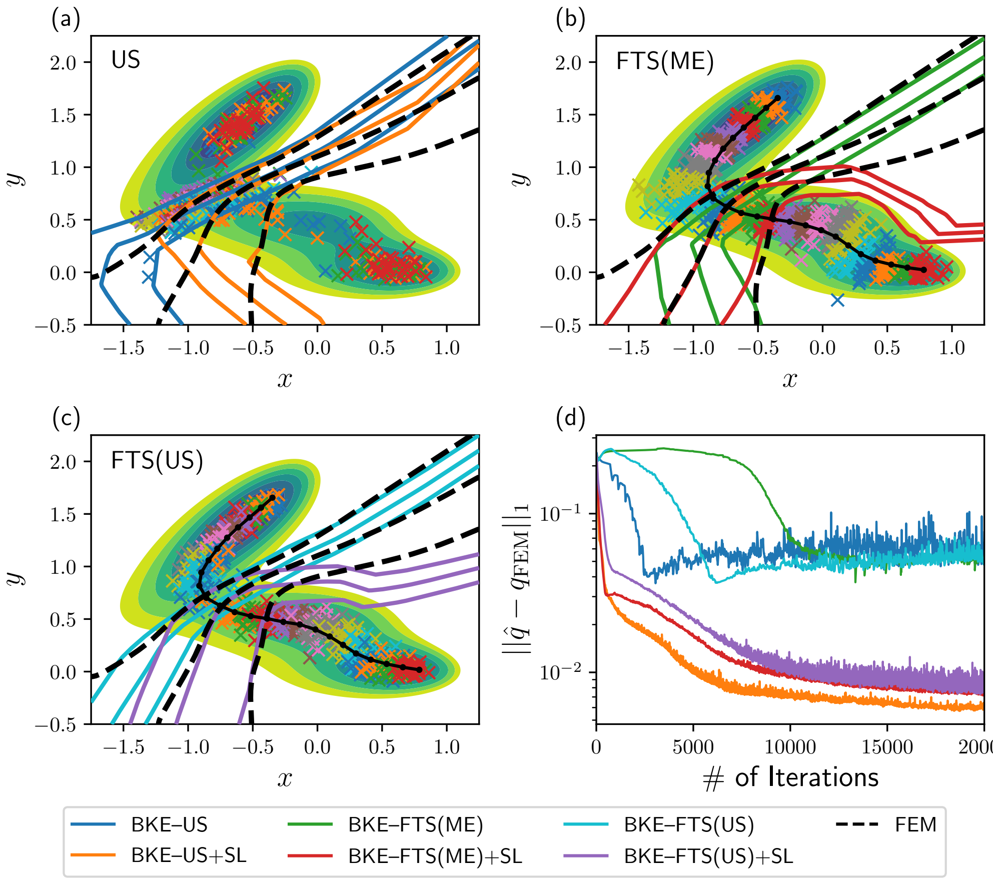


Figure 11: Isocommittor lines for $q = 0.1$, $0.5$, and $0.9$ (left to right) from (a) the BKE–US and BKE–US+SL method, (b) the BKE–FTS(ME) and BKE–FTS(ME)+SL method, and (c) the BKE–FTS(US) and BKE–FTS(US)+SL method. $\times$ markers denote representative samples obtained from algorithms without supervised 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.

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. 11(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. 11(a). As mentioned in Section 2.2, 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. 30). Conversely, all algorithms using the FTS method yield overlapping samples that homogeneously cover the transition tube and hence accurate estimates of reweighting factors (Figs. 31 and 32), indicating that reaction rate estimates may be computed with higher accuracy and lower variance.

Figure 12(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. (3.10)) shows convergence towards the FEM value (Fig. 12(c)). This shows the sampling error is still large, and larger batch sizes ($N_{\text{batch}}$) are needed to obtain accurate on-the-fly estimates. Figure 13(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

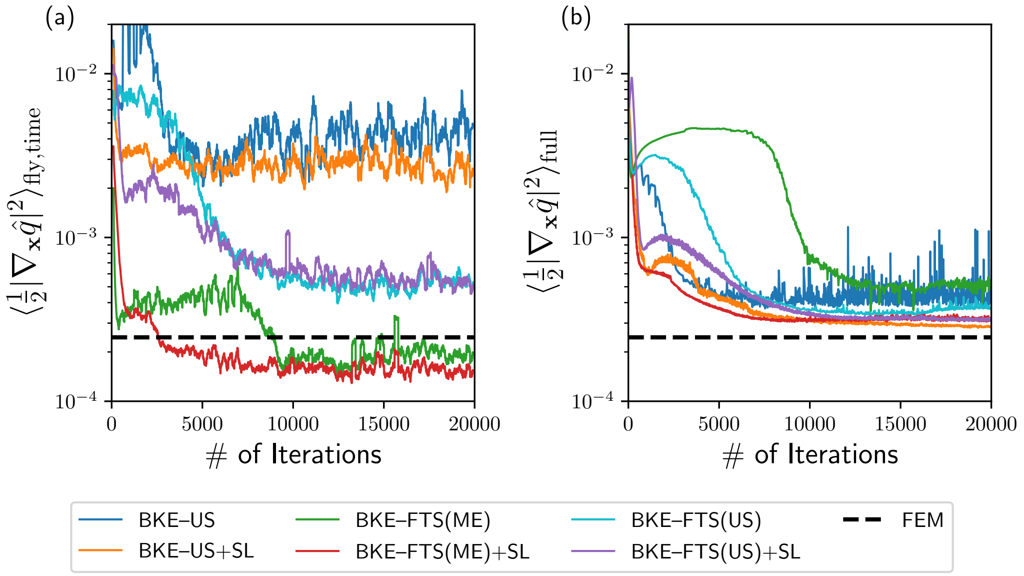


Figure 12: (a) The filtered on-the-fly estimate of the BKE loss obtained at every iteration, with the filtering window set to 200 iterations. (b) The ensemble-averaged loss per Eq. (3.10) obtained at every iteration.

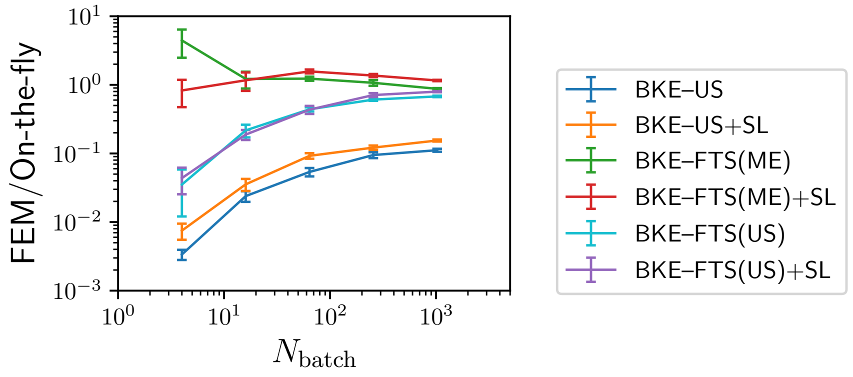


Figure 13: The ratio between the average BKE loss from FEM solution (Eq. (3.9)) and on-the-fly estimates, where the latter is cumulatively averaged over the last 3000 iterations of the neural network training.

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 2.2, 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 $z_\alpha$ (Fig. 30). 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. 21, 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.

## 3.3 Error Analysis of the Average BKE Loss Estimator

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. 14), 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 [54] 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. (3.5). 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 decoupling the influence of optimization from sampling. The histograms from this numerical experiment are shown in Fig. 15, 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. (3.10)). 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. 16, 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. (3.5). To this end, we provide a more concise notation for the estimator (Eq. (3.5)) by re-writing it as

$$\left\langle \frac{1}{2}|\nabla_{\mathbf{x}}\hat{q}(\mathbf{x};\boldsymbol{\theta}_\mathrm{k})|^2 \right\rangle_{\mathrm{fly}} = \frac{\sum_{\alpha=1}^{M} z_\alpha \left( \frac{1}{|\mathcal{M}_k^\alpha|} \sum_{\mathbf{x}\in\mathcal{M}_k^\alpha} \left[ \frac{\ell(\mathbf{x};\boldsymbol{\theta}_k)}{c(\mathbf{x};\boldsymbol{\theta}_k)} \right] \right)}{\sum_{\alpha=1}^{M} z_\alpha \left( \frac{1}{|\mathcal{M}_k^\alpha|} \sum_{\mathbf{x}\in\mathcal{M}_k^\alpha} \left[ \frac{1}{c(\mathbf{x};\boldsymbol{\theta}_k)} \right] \right)} = \frac{\sum_{\alpha=1}^{M} z_\alpha \bar{\ell}_\alpha^*}{\sum_{\alpha=1}^{M} z_\alpha \bar{1}_\alpha^*}\,, \tag{3.11}$$

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 (3.11) requires computing free energies through $z_\alpha$, 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 $z_\alpha$, $\bar{\ell}_\alpha^*$, and $\bar{1}_\alpha^*$ are determined individually. In what follows, we first investigate the statistics of $z_\alpha$ as computed via FEP.

To begin, we write the free-energy difference $\Delta F_{\alpha,\alpha'} = F_\alpha - F_{\alpha'}$ per Eq. (2.18) as

$$\beta\Delta F_{\alpha,\alpha'} = -\log\left[ \frac{1}{|\mathcal{M}_k^{\alpha'}|} \sum_{\mathbf{x}\in\mathcal{M}_k^{\alpha'}} \exp(-\beta\Delta W_{\alpha,\alpha'}(\mathbf{x};\boldsymbol{\theta}_k)) \right]\,, \tag{3.12}$$

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

$$\beta\Delta F_{\alpha,\alpha'} \approx -\log\left[ \frac{1}{|\mathcal{M}_k^{\alpha'}|} \sum_{\mathbf{x}\in\mathcal{M}_k^{\alpha'}} (1-\beta\Delta W_{\alpha,\alpha'}(\mathbf{x};\boldsymbol{\theta}_k)) \right] \tag{3.13}$$

$$\approx -\log\left[ 1 - \frac{1}{|\mathcal{M}_k^{\alpha'}|} \sum_{\mathbf{x}\in\mathcal{M}_k^{\alpha'}} \beta\Delta W_{\alpha,\alpha'}(\mathbf{x};\boldsymbol{\theta}_k) \right] \tag{3.14}$$

$$\approx \frac{1}{|\mathcal{M}_k^{\alpha'}|} \sum_{\mathbf{x}\in\mathcal{M}_k^{\alpha'}} \beta\Delta W_{\alpha,\alpha'}(\mathbf{x};\boldsymbol{\theta}_k)\,. \tag{3.15}$$

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 $\Delta W_{\alpha,\alpha'}(\mathbf{x};\boldsymbol{\theta}_k)$,

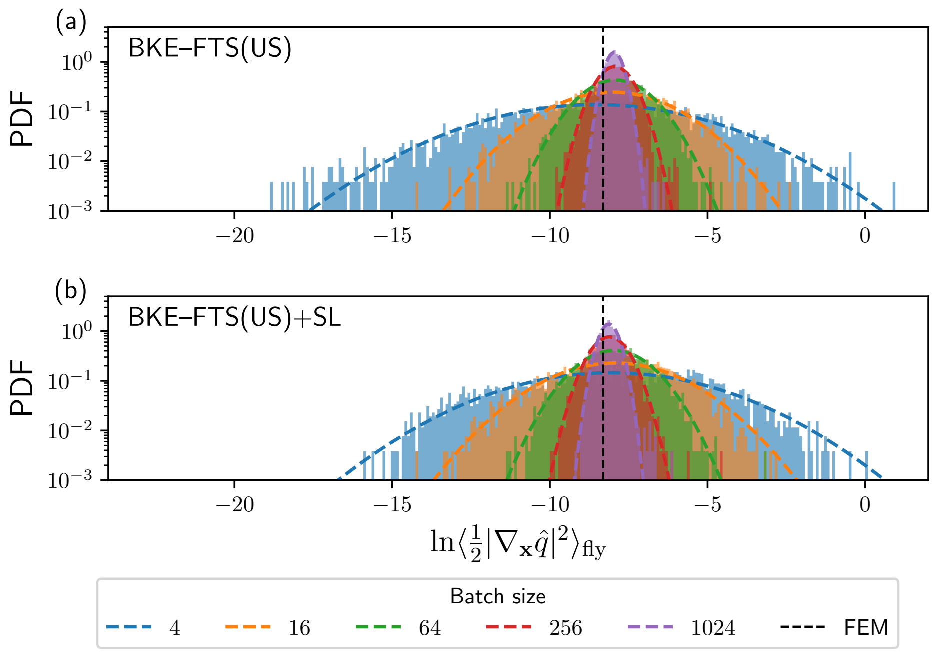


Figure 14: Histograms yielding the probability density functions for the on-the-fly estimate of the BKE loss at various batch sizes from the last 3000 iterations of training for the (a) BKE–FTS(US), and (b) BKE–FTS(US)+SL methods. Corresponding dashed lines are log-normal distributions fitted using the method of moments [54], while the dashed vertical black line corresponds to the average BKE loss from the FEM solution.

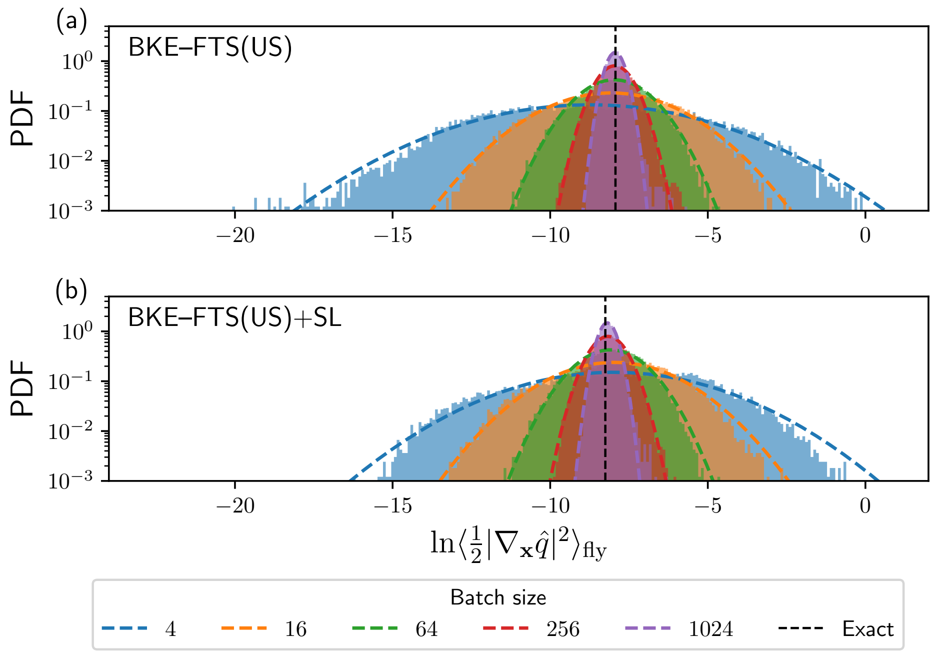


Figure 15: Histograms of the on-the-fly estimate of the average BKE loss at various batch sizes, with the neural network parameters fixed at every iteration for the (a) BKE–FTS(US), and (b) BKE–FTS(US)+SL methods. The neural network configuration corresponds to the one obtained from training at batch size 4. Corresponding dashed lines are log-normal distributions fitted using the method of moments [54], while the dashed vertical black line corresponds to the average BKE loss computed by numerical integration (Eq. (3.10)).

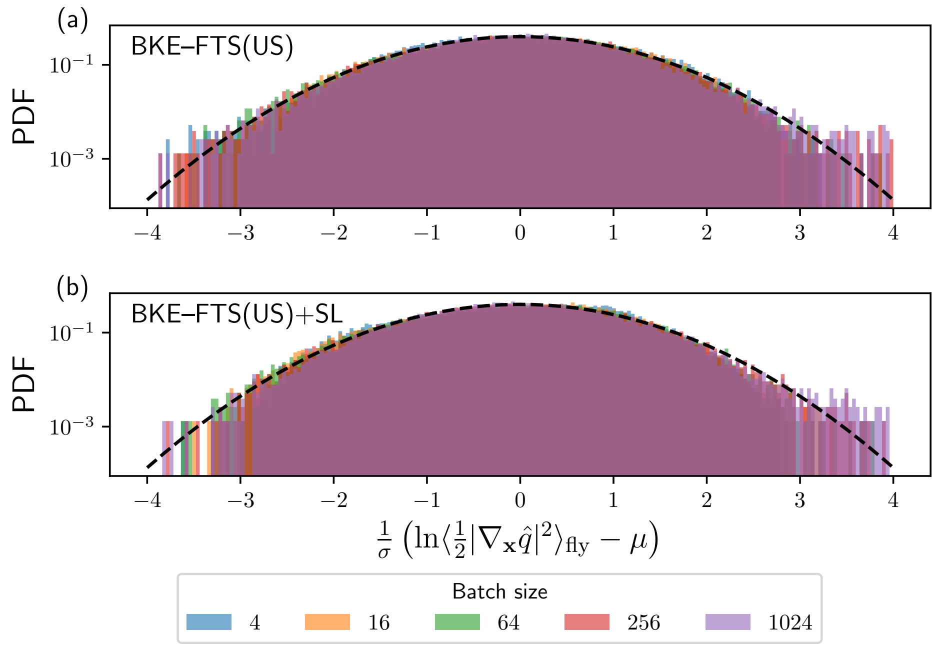


Figure 16: Histograms of the on-the-fly estimate of the average BKE loss shifted by the mean $\mu$ and normalized by the standard deviation $\sigma$ at various batch sizes, with the neural network parameters fixed at every iteration for the (a) BKE–FTS(US), and (b) BKE–FTS(US)+SL methods. The neural 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 $\sigma = 1$.

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. (2.19), for $\alpha$ not equal to the reference index $\gamma$, 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 [55]. Upon normalizing $z^{\star}_{\alpha}$ to obtain $z_{\alpha}$, we should observe approximately log-normal statistics for $z_{\alpha}$, since the normalization requires dividing $z^{\star}_{\alpha}$ with its sum, which is approximately log-normal [56–60].

The arguments we put forth for the statistics of $\beta\Delta F_{\alpha,\alpha'}$ and $z_{\alpha}$ 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. 33 and 34), which immediately imply that their exponentiation is log-normally distributed. The resulting reweighting factors $z_{\alpha}$ 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. 17 for representative replicas, and Fig. 35 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 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 $z_{\alpha}$ for these groups are shown in the second column of Table 1.

With the sampling distributions of $z_{\alpha}$ understood, we now study the sampling distributions for $\bar{\ell}^*_{\alpha}$ and $\bar{1}^*_{\alpha}$. Assuming the values of $\ell^*_{\alpha}$ and $1^*_{\alpha}$ 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. 17 for representative histograms, and Figs. 36 and 37 for all histograms. Since log-normality arises when normally-distributed random

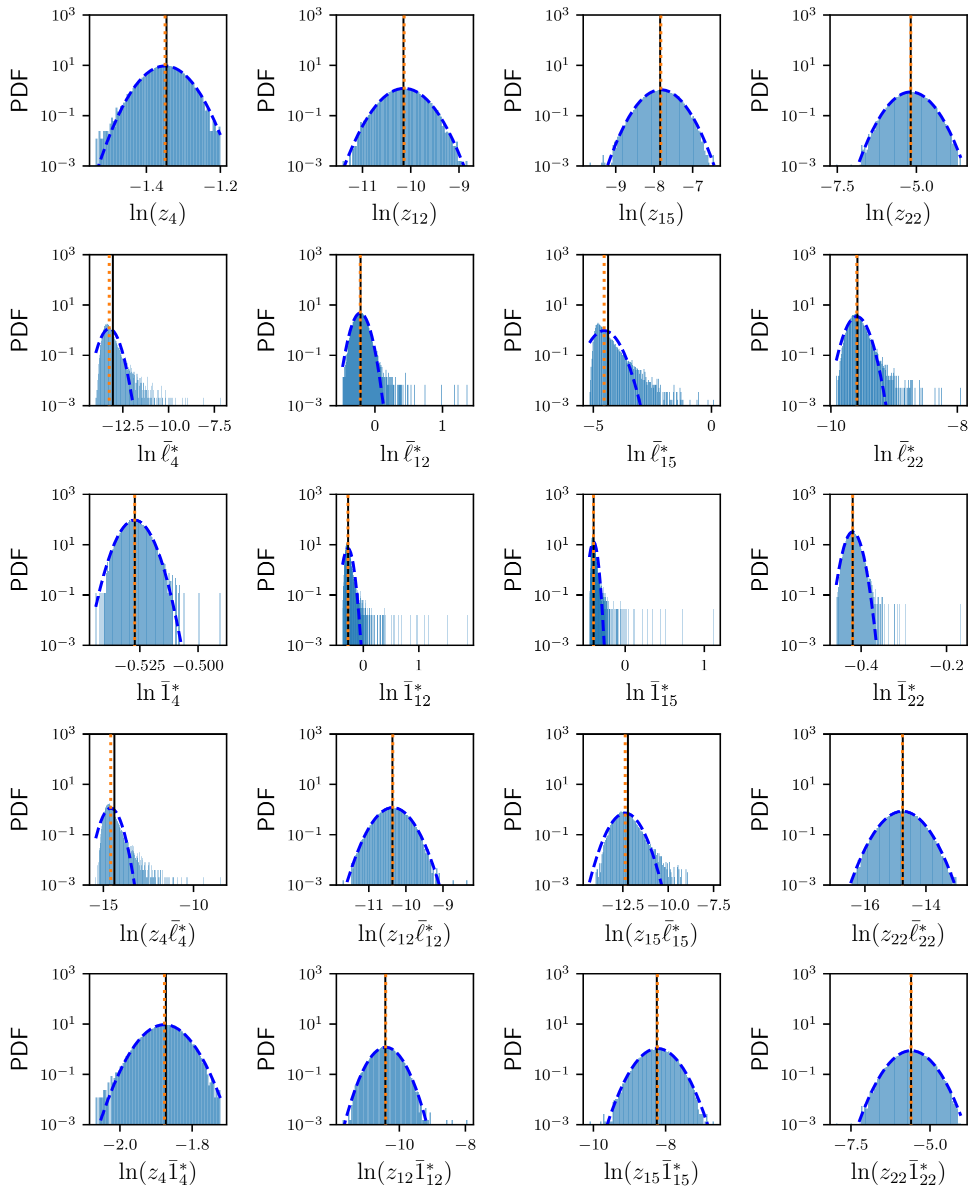


Figure 17: Probability density functions of the various quantities representative of Table 1. Data is obtained from sampling with batch size 1024, with a fixed neural network obtained from the BKE–FTS(US)+SL method at the same batch size. Dashed blue lines are log-normal distributions fitted using the method of moments [54], 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.

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 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. (3.12), are log-normal; see the fourth and fifth rows of Fig. 17 for representative

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

Table 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. 35–39.

histograms, and Figs. 38 and 39 for all histograms, as well as the fifth and sixth columns of Table 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

$$\rho_Y(y) = \int_{-\infty}^{\infty} \mathrm{d}x\ \rho_{X_1}(x)\rho_{X_2}(y-x)\,. \tag{3.16}$$

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

$$\rho_Y(y) \approx \int_{-\infty}^{\infty} \mathrm{d}x\ \rho_{X_1}(x)\delta(y-x) = \rho_{X_1}(y)\,. \tag{3.17}$$

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. (3.17) 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 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. (3.11), 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 [56–60], both the numerator and denominator should be approximately log-normal. From simulations, we find that the numerator is log-normally distributed (Fig. 18(a)) while the denominator is log-normally distributed with slight deviations in the tails (Fig. 18(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. 18(c). This is also in agreement with what is observed during training (Fig. 14), and when the neural network is fixed (Fig. 15). Although the log-normality of the denominator is only approximate, one can use the previous argument on sums of random variables, i.e., Eq. (3.17), 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 $N_{\mathrm{batch}}$. Denoting the mean and variance of the

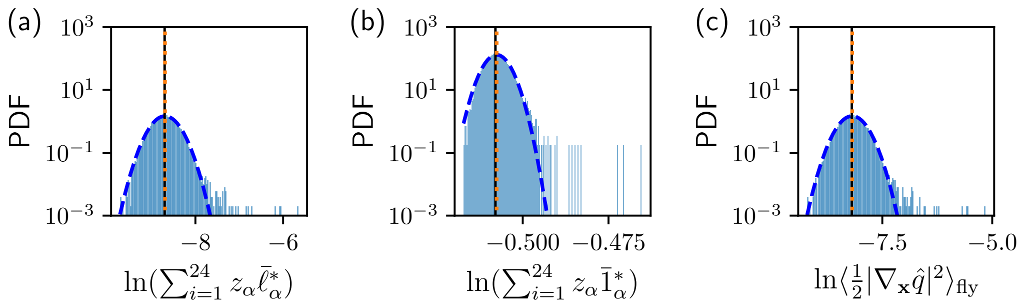


Figure 18: Probability density functions of sums of $z_\alpha \bar{\ell}_\alpha^*$, $z_\alpha \bar{1}_\alpha^*$, and the on-the-fly estimate of the BKE loss function in logarithmic space. Data is obtained from sampling with batch size 1024, with a fixed neural network obtained from the BKE–FTS(US)+SL method at the same batch size. Dashed blue lines are log-normal distributions fitted using the method of moments [54], 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.

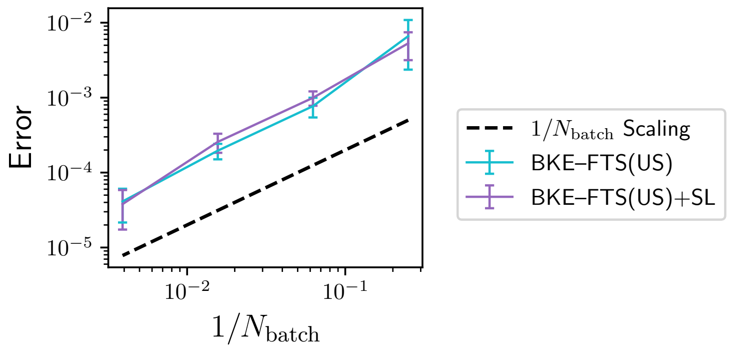


Figure 19: The absolute error in the on-the-fly BKE loss at different batch sizes, with respect to the largest batch size. All error bars are 95 confidence intervals.

log-normal distribution as $\mu$ and $\sigma^2$, respectively, we expect that the cumulative mean of the on-the-fly BKE loss over iterations is given by [55]

$$\frac{1}{K - k^\star + 1} \sum_{k=k^\star}^{K} \left\langle \frac{1}{2} |\nabla_{\mathbf{x}} \hat{q}(\mathbf{x}; \boldsymbol{\theta}_{\mathrm{k}})|^2 \right\rangle_{\text{fly}} \approx \exp\left(\mu + \frac{1}{2}\sigma^2\right), \tag{3.18}$$

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 (3.18) 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. 11(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. 12 is always less than one. Furthermore, $\sigma^2 \sim O(1/N_{\text{batch}})$, implying for large $N_{\text{batch}}$ that

$$\frac{1}{K - k^\star + 1} \sum_{k=k^\star}^{K} \left\langle \frac{1}{2} |\nabla_{\mathbf{x}} \hat{q}(\mathbf{x}; \boldsymbol{\theta}_{\mathrm{k}})|^2 \right\rangle_{\text{fly}} \sim \exp(\mu)\left(1 + O(1/N_{\text{batch}})\right), \tag{3.19}$$

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 $N_{\text{batch}}$ in Figure 13 for both the BKE–FTS(US) and BKE–FTS(US)+SL methods, where the $O(1/N_{\text{batch}})$ scaling can be observed.

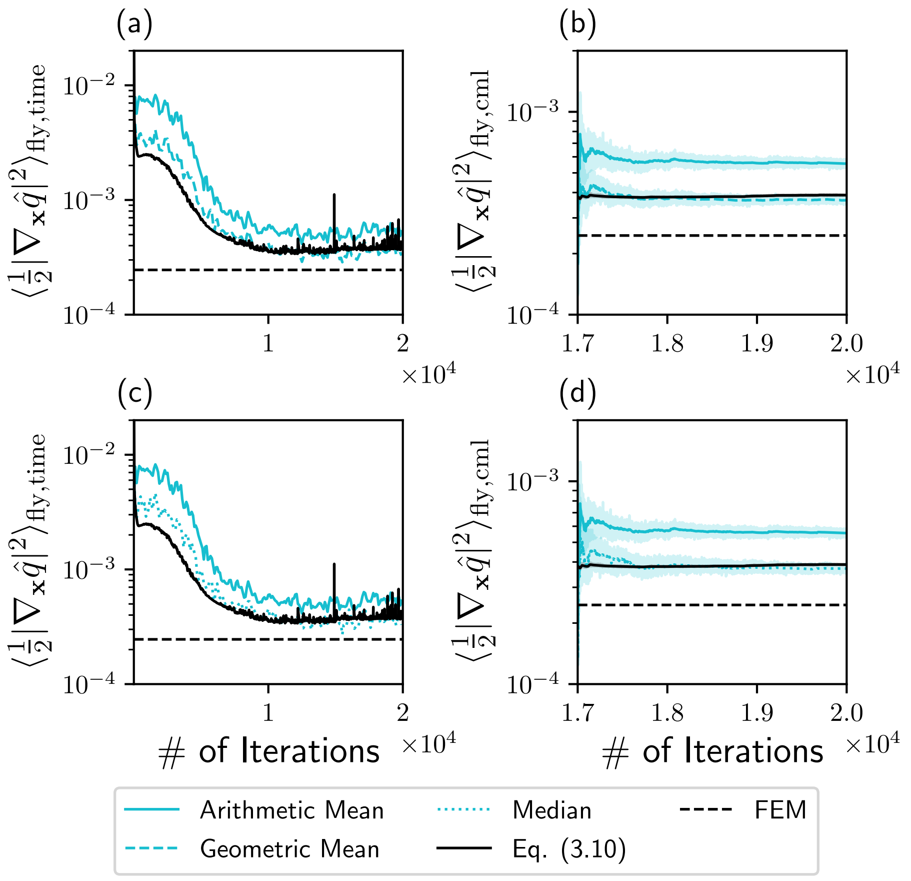


Figure 20: Using the geometric mean (a,b) and median (c,d) to remove the sampling error in the filtered on-the-fly estimates (left column) and cumulative average (right column) of the on-the-fly estimates in the BKE–FTS(US) method for batch size 64. Note that the remaining error between the FEM value and the average BKE loss computed per Eq. (3.10) 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.

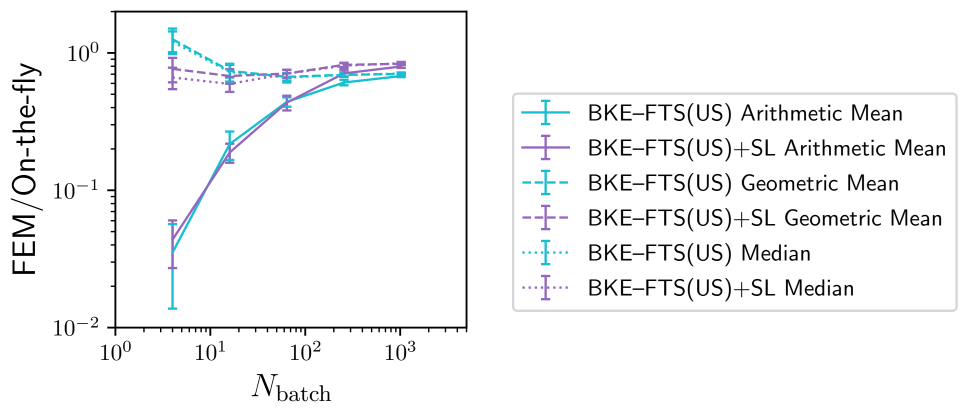


Figure 21: The ratio between FEM and on-the-fly estimates, after taking the geometric mean and median. Error bars are 95 confidence interval.

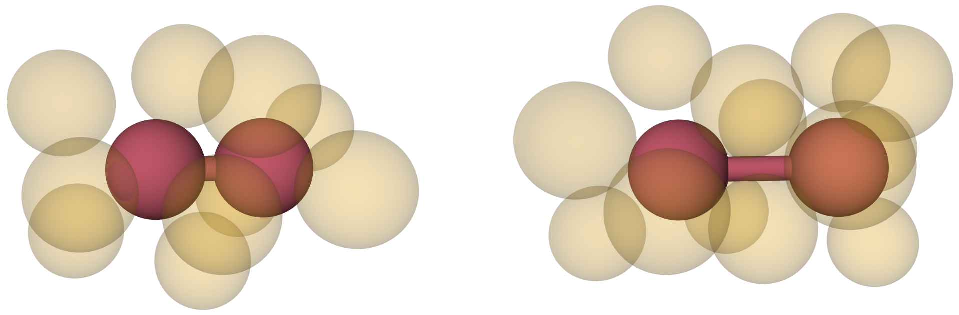


Figure 22: Dimer particles in (left) compact and (right) extended states for $r_0 = 2^{1/6}$ and $s = 0.25$ for a system with $\rho = 0.9$. Only the seven nearest neighbors of each solvent particle are visualized and made transparent. Image created using Ovito [61].

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. (3.10)). 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 [55]. We demonstrate this by applying the geometric mean (Figs. 20(a,b)) and median (Figs. 20(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. 21 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. 14(a) and 15(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.

## 4 Computational Study of a Solvated Dimer System

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 1–6 on a solvated dimer system [62], where the dimer transitions between a compact and an extended state; see Fig. 22. 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

$$V_{\text{dimer}}(r) = h\left[1 - \frac{(r - r_0 - s)^2}{s^2}\right]^2, \tag{4.1}$$

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. 22). The solvent particles interact between themselves and the dimer particles by the Weeks-Chandler-

Andersen potential [63]

$$V_{\mathrm{WCA}}\left(r\right)=\left(4\epsilon\left[\left(\frac{1}{r}\right)^{12}-\left(\frac{1}{r}\right)^{6}\right]+\epsilon\right)\Theta\left(r_{\mathrm{WCA}}-r\right)\ , \tag{4.2}$$

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

$$\hat{q}(\mathbf{x},\mathbf{z};\boldsymbol{\theta})=\sigma\left(f(\mathbf{x},\mathbf{z};\boldsymbol{\theta})\right)\ , \tag{4.3}$$

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 [64] available with PyTorch Geometric [65]. 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.3. See also Ref. [64] for more details on the general architecture of SchNet and our code repository[†] 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

$$W_{\alpha}=\frac{1}{2}\kappa_{\alpha}\left(r-r_{\alpha}\right)^{2}\ , \tag{4.4}$$

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 3. The neural network parameters are initialized randomly, and updated by minimizing Eq. (3.2) 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.3.

Figure 23 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

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

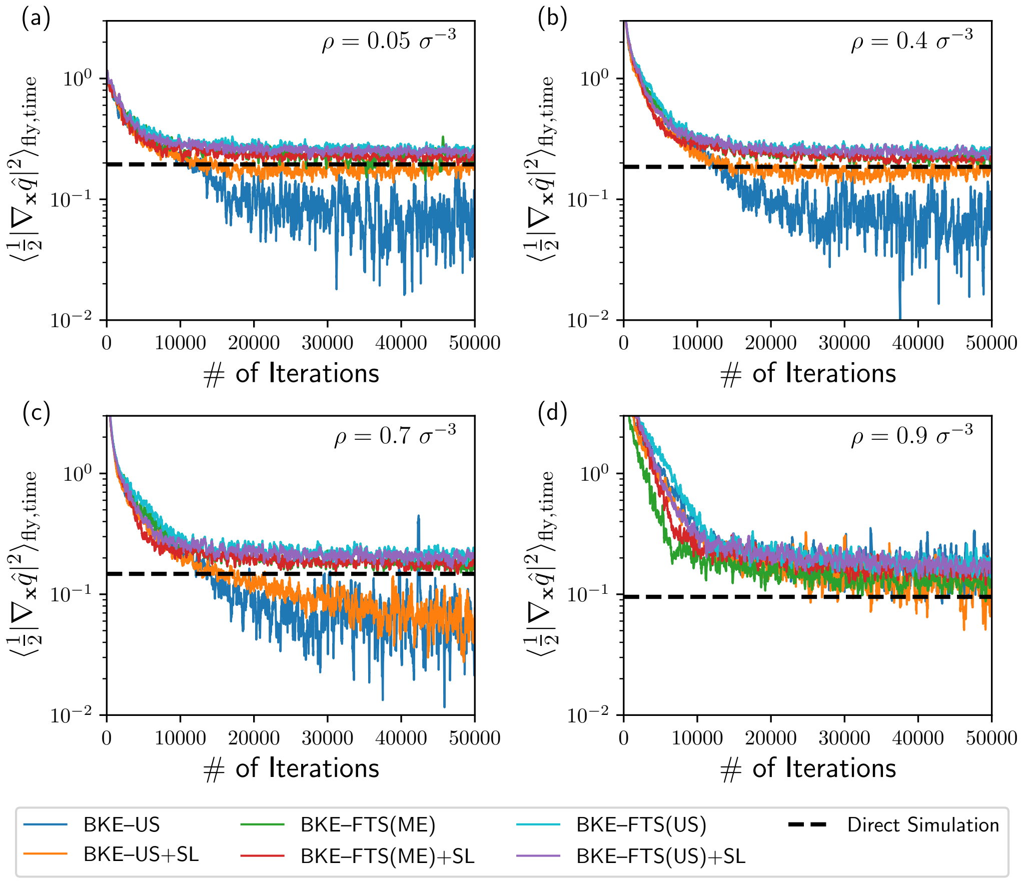


Figure 23: The filtered on-the-fly estimate of the BKE loss obtained at every iteration for the solvated dimer system, with the filtering window set to 200 iterations. A total of $10^4$ unbiased trajectories are used to compute a direct estimate of the reaction rate (dashed line) for comparison with the proposed methods.

0.7 (Fig. 23(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) 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. 24). 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 $\rho = 0.9$.

The accuracy of all methods can be assessed by comparing an empirical committor function $q_{\mathrm{emp}}(r)$ computed at a fixed value of bond length $r$ with the corresponding value $\hat{q}(r, \mathbf{z}; \boldsymbol{\theta})$ 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 $q_{\mathrm{emp}}(r)$ and $\hat{q}(r, \mathbf{z}; \boldsymbol{\theta})$ represent estimates of the mean committor at fixed $r$. Given the full empirical committor function $q_{\mathrm{emp}}(\mathbf{x})$ (Eq. (2.20)) 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

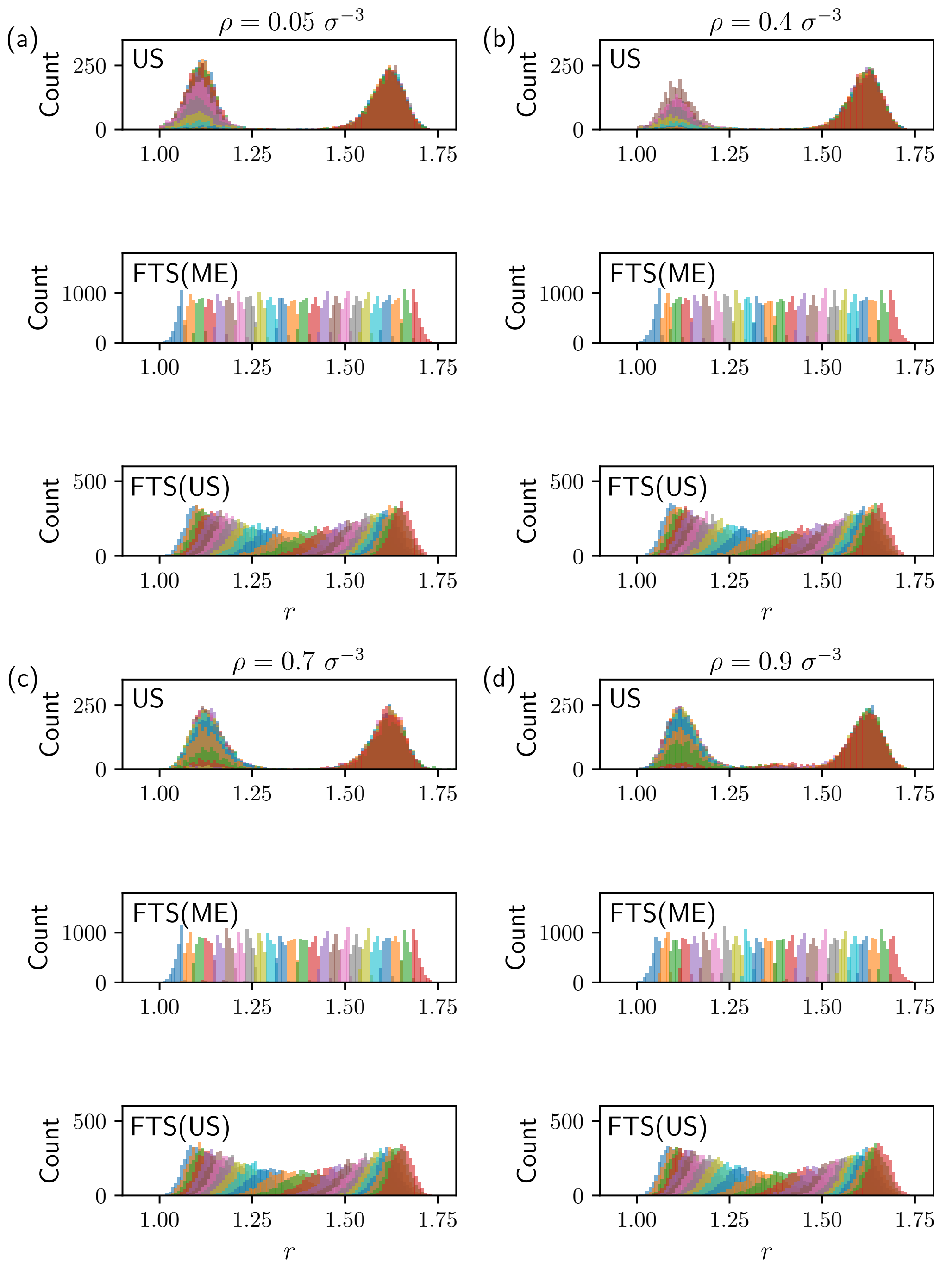


Figure 24: Histograms of dimer distances obtained from the BKE–US method (top), the BKE–FTS(ME) method (middle), and the BKE–FTS(US) method (bottom) for (a) $\rho = 0.05$, (b) $\rho = 0.4$, (c) $\rho = 0.7$, and (d) $\rho = 0.9$.

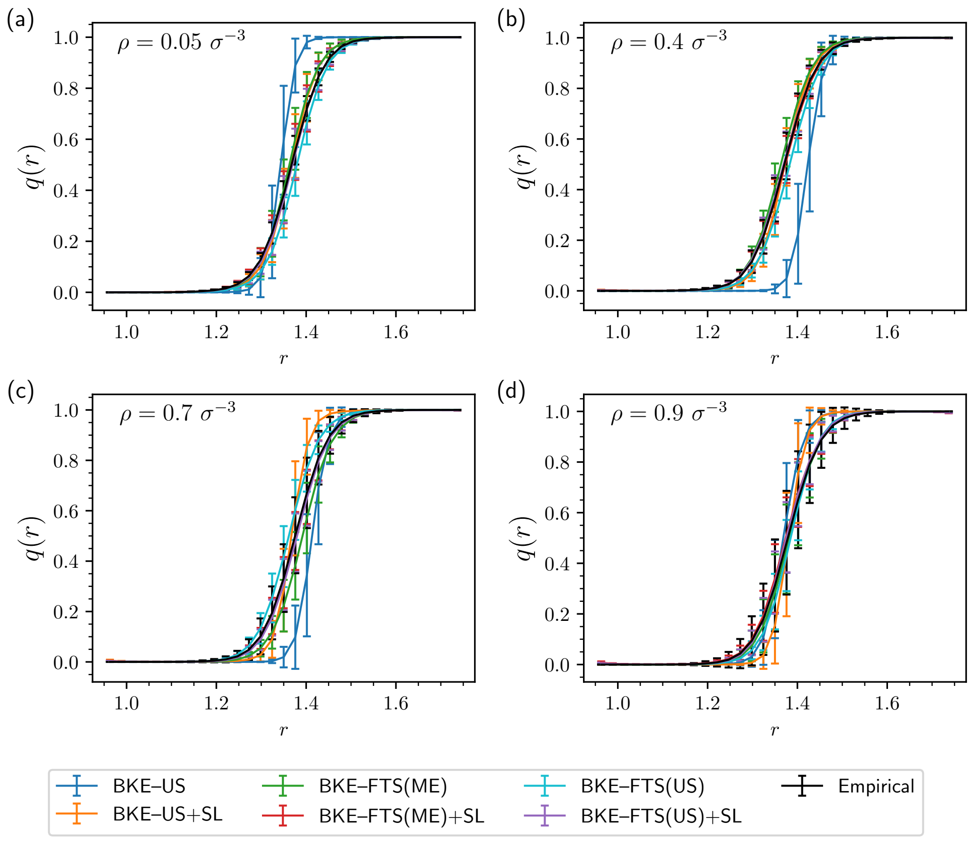


Figure 25: Average committor profiles for the methods compared to the empirical results for dimer in solvent systems. The values are binned for 31 windows between $r_{\min} = 0.95$ and $r_{\max} = 1.75$.

procedure yields the following formulas:

$$\hat{q}(r_i, \mathbf{z}; \boldsymbol{\theta}) = \frac{1}{|\mathcal{Q}_i|} \sum_{\mathbf{x} \in \mathcal{Q}_i} \hat{q}(\mathbf{x}, \mathbf{z}; \boldsymbol{\theta}) \,, \tag{4.5}$$

$$q_{\text{emp}}(r_i) = \frac{1}{|\mathcal{Q}_i|} \sum_{\mathbf{x} \in \mathcal{Q}_i} q_{\text{emp}}(\mathbf{x}) \,, \tag{4.6}$$

where every $\mathbf{x} \in \mathcal{Q}_i$ is obtained from the configurations sampled via the umbrella potential in Eq. (4.4) and $q_{\text{emp}}(\mathbf{x})$ is computed using $1250$ trajectories per configuration $\mathbf{x}$. Figure 25 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.,

$$||\hat{q}(r_i) - q_{\text{emp}}(r_i)||_1 = \frac{1}{|\mathcal{Q}_i|} \sum_{\mathbf{x} \in \mathcal{Q}_i} |\hat{q}(\mathbf{x}, \mathbf{z}; \boldsymbol{\theta}) - q_{\text{emp}}(\mathbf{x})| \,. \tag{4.7}$$

Figure 26 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 $\rho$ 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.

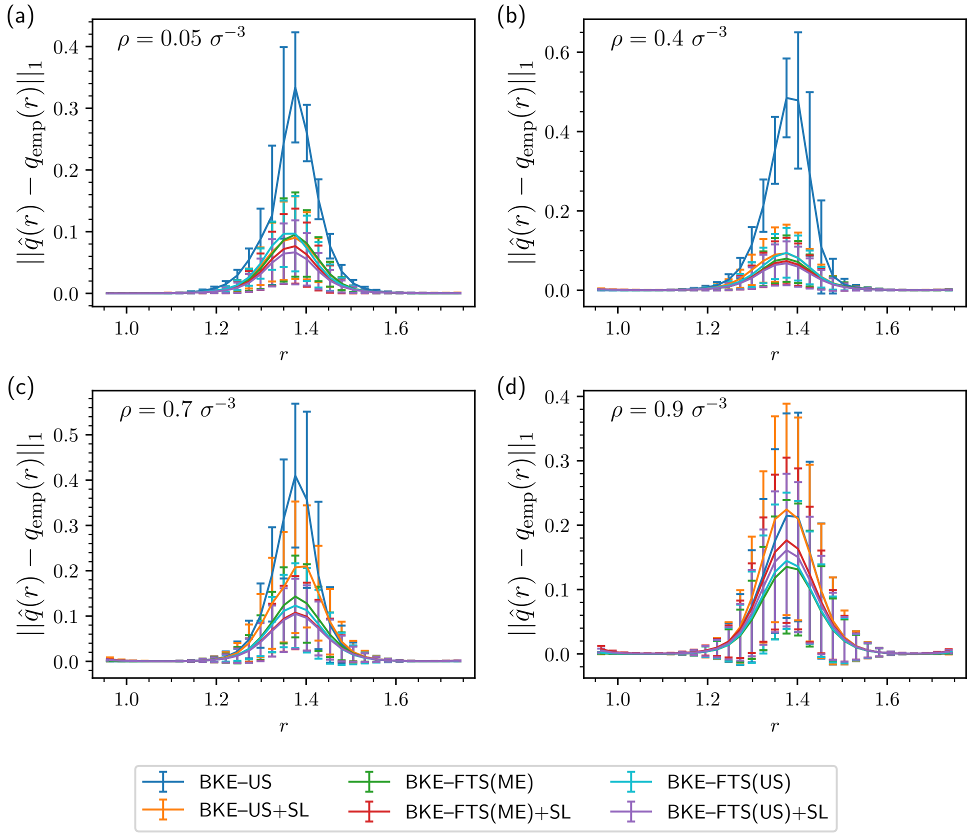


Figure 26: Mean absolute error profiles for the methods compared to the empirical results for dimer in solvent systems. The values are binned for 31 windows between $r_{\min} = 0.95$ and $r_{\max} = 1.75$.

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. (3.10)), 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. (4.4) was used to obtain $10^4$ configurations from all 32 replicas. Given this dataset, we compute the reweighting factors $z_\alpha$ using the multistate Bennett acceptance ratio (MBAR) method. Note that MBAR is used instead of FEP since it yields estimates of $z_\alpha$ with lower error than FEP, albeit at a higher computational cost [66]. Once the MBAR reweighting factors $z_\alpha^{\mathrm{MBAR}}$ are computed, the reaction rate from the neural network can be estimated from a modification of Eq. (3.5) for umbrella sampling,

$$\hat{\nu}_R = \left\langle |\nabla_{\mathbf{x}} \hat{q}(\mathbf{x};\boldsymbol{\theta})|^2 \right\rangle = \frac{\displaystyle\sum_{\alpha=1}^{32} \frac{2z_\alpha^{\mathrm{MBAR}}}{|\mathcal{M}^\alpha|} \sum_{\mathbf{x}\in\mathcal{M}^\alpha} \left[\frac{\ell(\mathbf{x};\boldsymbol{\theta})}{c(\mathbf{x};\boldsymbol{\theta})}\right]}{\displaystyle\sum_{\alpha=1}^{32} \frac{z_\alpha^{\mathrm{MBAR}}}{|\mathcal{M}^\alpha|} \sum_{\mathbf{x}\in\mathcal{M}^\alpha} \left[\frac{1}{c(\mathbf{x};\boldsymbol{\theta})}\right]} . \tag{4.8}$$

Evaluating Eq. (4.8) produces the results seen in Fig. 27(a), which are compared to the true reaction rate as estimated by direct molecular simulation. The results in Fig. 27(a) mirror the trends seen in Fig. 26.

As established by the error analysis in Section 3.3, we may avoid costly computation in Eq. (4.8) 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

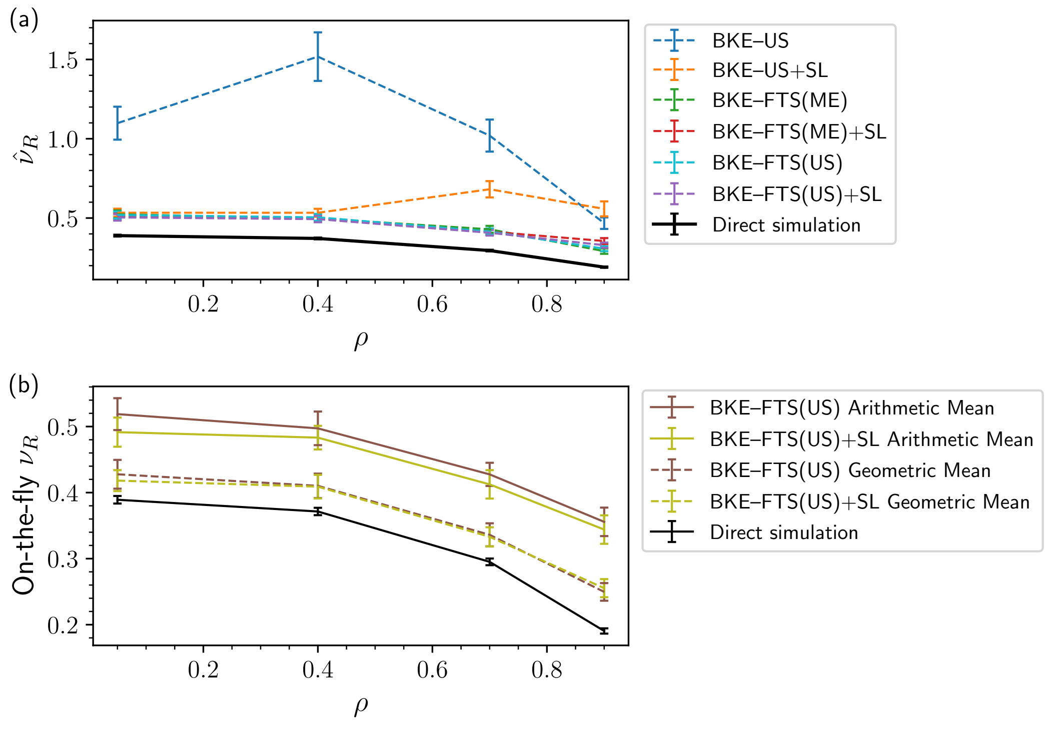


Figure 27: (a) The reaction rate $\hat{\nu}_R$ of the neural network per Eq. (4.8) as a function of density. (b) Comparison between the arithmetic mean and geometric mean applied to the last 3000 samples from training to direct simulation as a function of density. Error bars are 95 confidence interval.

the on-the-fly estimates, taken from the last portion of training, is shown in Fig. 27(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 2.4.2 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 Conclusion & Future Work

In summary, building on the work of Ref. [1], 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 [67], with neural networks satisfying equivariance throughout the hidden layers having been shown to yield increased accuracy in predictions of molecular properties over SchNet [68]. 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 $\mathrm{Na}^+$–$\mathrm{Cl}^-$ ionic pairs [69–72], to excitation events in glassy systems, where the transition state is known to have elastic signatures that are crucial for the structural relaxation [73].

## Acknowledgments

We thank Professors Benjamin Recht and Moritz Hardt for helpful discussions about machine learning methodologies, and the use of supervised learning elements in reinforcement learning. We also thank Chloe Hsu for insightful comments. This work is supported by Director, Office of Science, Office of Basic Energy Sciences, of the U.S. Department of Energy under contract No. DEAC02-05CH11231.

## Data Availability

The data that support the findings of this study are available from the corresponding author upon reasonable request

## References

[1] 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).

[2] P. J. Lu and D. A. Weitz, "Colloidal particles: Crystals, glasses, and gels", Annual Review of Condensed Matter Physics **4**, 217–233 (2013).

[3] P. Jungwirth and B. Winter, "Ions at aqueous interfaces: From water surface to hydrated proteins", Annual Review of Physical Chemistry **59**, 343–366 (2008).

[4] 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).

[5] S. Plimpton, "Fast parallel algorithms for short-range molecular dynamics", Journal of Computational Physics **117**, 1–19 (1995).

[6] 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).

[7] 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).

[8] 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).

[9] 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).

[10] W. E and E. Vanden-Eijnden, "Towards a theory of transition paths", Journal of Statistical Physics **123**, 503–523 (2006).

[11] 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).

[12] 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).

[13] 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).

[14] B. Peters, "Using the histogram test to quantify reaction coordinate error", The Journal of Chemical Physics **125**, 241101 (2006).

[15]L. Onsager, “Initial recombination of ions”, Phys. Rev. **54**, 554–557 (1938).

[16]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).

[17]B. Peters, “Reaction coordinates and mechanistic hypothesis tests”, Annual Review of Physical Chemistry **67**, 669–690 (2016).

[18]D. Frenkel and B. Smit, *Understanding Molecular Simulation: From Algorithms to Applications*, 2nd ed. (Elsevier, 2001).

[19]G. Torrie and J. Valleau, “Nonphysical sampling distributions in Monte Carlo free-energy estimation: Umbrella sampling”, Journal of Computational Physics **23**, 187–199 (1977).

[20]A. M. Berezhkovskii and A. Szabo, “Diffusion along the splitting/commitment probability reaction coordinate”, The Journal of Physical Chemistry B **117**, PMID: 23777371, 13115–13119 (2013), eprint: `https://doi.org/10.1021/jp403043a`.

[21]R. Durrett, *Stochastic Calculus: A Practical Introduction* (CRC Press, 1996).

[22]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).

[23]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).

[24]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`.

[25]P. Papadopoulos, *ME 280A: Introduction to the Finite Element Method*, `https://csml.berkeley.edu/Notes/ME280A.pdf`, 2015.

[26]J. Nocedal and S. Wright, *Numerical Optimization* (Springer-Verlag New York, 2006).

[27]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), eprint: `https://doi.org/10.1021/jp045546c`.

[28]B. Peters and B. L. Trout, “Obtaining reaction coordinates by likelihood maximization”, The Journal of Chemical Physics **125**, 054108 (2006), eprint: `https://doi.org/10.1063/1.2234477`.

[29]H. Robbins and S. Monro, “A stochastic approximation method”, Annals of Mathematical Statistics **22**, 400–407 (1951).

[30]B. Polyak, “Some methods of speeding up the convergence of iteration methods”, USSR Computational Mathematics and Mathematical Physics **4**, 1–17 (1964).

[31]D. P. Kingma and J. Ba, *Adam: A method for stochastic optimization*, 2017, arXiv:`1412.6980`.

[32]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).

[33]T. Lelièvre, M. Rousset, and G. Stoltz, *Free Energy Computations* (Imperial College Press, 2010).

[34]R. W. Zwanzig, “High-temperature equation of state by a perturbation method. I. Nonpolar gases”, The Journal of Chemical Physics **22**, 1420–1426 (1954).

[35]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).

[36]A. Pohorille, C. Jarzynski, and C. Chipot, “Good practices in free-energy calculations”, The Journal of Physical Chemistry B **114**, 10235–10253 (2010).

[37]S. S. Du, X. Zhai, B. Poczos, and A. Singh, *Gradient descent provably optimizes over-parameterized neural networks*, 2019, arXiv:`1810.02054`.

[38]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).

[39]M. Hardt and B. Recht, *Patterns, Predictions, and Actions: A story about machine learning* (`https://mlstory.org`, 2021), arXiv:`2102.05242`.

[40]S. Osher and R. Fedkiw, “Implicit Functions”, in *Level Set Methods and Dynamic Implicit Surfaces* (Springer New York, New York, NY, 2003).

[41]T. Hastie and W. Stuetzle, “Principal curves”, Journal of the American Statistical Association **84**, 502–516 (1989).

[42]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).

[43]E. Vanden-Eijnden and M. Venturoli, “Markovian milestoning with Voronoi tessellations”, The Journal of Chemical Physics **130**, 194101 (2009).

[44]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).

[45]K. Zinovjev and I. Tuñón, “Adaptive finite temperature string method in collective variables”, The Journal of Physical Chemistry A **121**, 9764–9772 (2017).

[46]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).

[47]G. Shrivastav, E. Vanden-Eijnden, and C. F. Abrams, “Mapping saddles and minima on free energy surfaces using multiple climbing strings”, The Journal of Chemical Physics **151**, 124112 (2019), eprint: `https://doi.org/10.1063/1.5120372`.

[48]C. Schütte, A. Fischer, W. Huisinga, and P. Deuflhard, “A direct approach to conformational dynamics based on hybrid monte carlo”, Journal of Computational Physics **151**, 146–168 (1999).

[49]C. Schütte, F. Noé, J. Lu, M. Sarich, and E. Vanden-Eijnden, “Markov state models based on milestoning”, The Journal of Chemical Physics **134**, 204105 (2011), eprint: `https://doi.org/10.1063/1.3590108`.

[50]G. R. Bowman, V. S. Pande, and F. Noé, *An Introduction to Markov State Models and Their Application to Long Timescale Molecular Simulation*, Vol. 797 (Springer Science & Business Media, 2013).

[51]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).

[52]A. Logg, K.-A. Mardal, and G. Wells, *Automated Solution of Differential Equations by the Finite Element Method: The FEniCS Book* (Springer, 2012).

[53]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).

[54]K. Pearson, “Method of moments and method of maximum likelihood”, Biometrika **28**, 34–59 (1936).

[55]C. Forbes, M. Evans, N. Hastings, and B. Peacock, *Statistical Distributions*, 4th ed. (John Wiley & Sons, 2011).

[56]N. A. Marlow, “A normal limit theorem for power sums of independent random variables”, Bell System Technical Journal **46**, 2081–2089 (1967).

[57]E. Barouch, G. Kaufman, and M. Glasser, “On sums of lognormal random variables”, Studies in Applied Mathematics **75**, 37–55 (1986).

[58]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).

[59]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).

[60]S. Asmussen and L. Rojas-Nandayapa, “Asymptotics of sums of lognormal random variables with gaussian copula”, Statistics and Probability Letters **78**, 2709–2714 (2008).

[61]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).

[62]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).

[63]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), eprint: `https://doi.org/10.1063/1.1674820`.

[64]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).

[65]M. Fey and J. E. Lenssen, “Fast graph representation learning with PyTorch Geometric”, CoRR **abs/1903.02428** (2019), arXiv:`1903.02428`.

[66]M. R. Shirts and J. D. Chodera, “Statistically optimal analysis of samples from multiple equilibrium states”, The Journal of Chemical Physics **129**, 124105 (2008).

[67]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 [cs.LG]`.

[68]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 [physics.comp-ph]`.

[69]A. Laio and M. Parrinello, “Escaping free-energy minima”, Proceedings of the National Academy of Sciences **99**, 12562–12566 (2002).

[70]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).

[71]P. L. Geissler, C. Dellago, D. Chandler, J. Hutter, and M. Parrinello, “Autoionization in liquid water”, Science **291**, 2121–2124 (2001).

[72]A. J. Ballard and C. Dellago, “Toward the mechanism of ionic dissociation in water”, The Journal of Physical Chemistry B **116**, 13490–13497 (2012).

[73]M. R. Hasyim and K. K. Mandadapu, “A theory of localized excitations in supercooled liquids”, J. Chem. Phys. **155**, 044504 (2021).

[74]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).

[75]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).

[76]N. Metropolis and S. Ulam, “The Monte Carlo method”, Journal of the American Statistical Association **44**, 335–341 (1949).

## A Computing Reweighting Factors in the Master-Equation Approach

Recall that the reweighting factor $z_\alpha$ in the BKE–FTS(ME) and BKE–FTS(ME)+SL methods are computed by solving the master equation Eq. (2.48). One can re-write Eq. (2.48) as a matrix equation:

$$\mathbf{K}\mathbf{z} = \mathbf{0}\,, \tag{A.1}$$

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. (A.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 $z_\alpha$, 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 $z_\alpha$ converge in the limit of large batch sizes to the $z_\alpha$ computed by numerical integration of Eq. (2.47); see Fig. 31. Note that a range of tolerance values $10^{-11}$–$10^{-8}$ have also been used with no change to the results.

## B 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 [74] for performing optimization[†].

### B.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. (2.14). 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 $\kappa^{\parallel}_\alpha = 5$, and the choice of $\kappa^{\perp}_\alpha$ 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. (2.38)-(2.39). In all algorithms, the friction coefficient $\gamma = 1$, and step size $\Delta t = 0.005$. The size of $\alpha$-th batch at every iteration is set to $|\mathcal{M}^\alpha_k| = 16$ and $|\mathcal{R}^\alpha_k| = 16$ for methods employing umbrella sampling and the FTS method, respectively. Each sample $\mathbf{x} \in \mathcal{M}^\alpha_k$ and $\mathbf{x} \in \mathcal{R}^\alpha_k$ 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 $\alpha$-th mini-batch is $|\mathcal{C}^\alpha_k| = 0.5|\mathcal{C}^\alpha|$, 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. (2.38)-(2.39), but with $R_\alpha$ 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 [30], which takes in two hyper-parameters as inputs. The first is the step size/learning rate $\eta$, while the second is the momentum coefficient $\mu$. Given any function $f(\boldsymbol{\theta})$ to minimize, the Heavy-Ball method updates model parameters $\boldsymbol{\theta}_k$ with the following equation:

$$\mathbf{m}_{k+1} = \mu \mathbf{m}_k + \nabla_{\boldsymbol{\theta}} f(\boldsymbol{\theta}_k)\,, \tag{B.1}$$
$$\boldsymbol{\theta}_{k+1} = \boldsymbol{\theta}_k - \eta \mathbf{m}_{k+1}\,, \tag{B.2}$$

[†]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.1) corresponds to, e.g., Eq. (2.15) for the BKE–US and BKE–FTS(US) method, and Eq. (2.46) 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. (2.43) with a momentum-variant called the Nesterov's method [42]. As implemented in PyTorch, which follows the simplified version in [75], the Nesterov's update can be written as

$$\mathbf{m}_{k+1} = \mu^2 \mathbf{m}_k + (1+\mu)\nabla_{\boldsymbol{\varphi}^\alpha} \hat{C}(\{\boldsymbol{\varphi}_k^\alpha\}) \,, \tag{B.3}$$

$$\boldsymbol{\varphi}_\star^\alpha = \boldsymbol{\varphi}_k^\alpha - \Delta\tau \mathbf{m}_{k+1} \,, \tag{B.4}$$

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 Second Study: Muller-Brown Potential

In the second study, umbrella sampling is performed with $M = 24$ replicas with dynamics given by Metropolis Monte Carlo [18, 76]. The particle is displaced in both directions by a random value between $-\Delta r$ and $\Delta r$ to yield a new position $\mathbf{x}'$, which is accepted with probability given by $P_{\text{acc}} = \min\left[1, \exp\left(-\beta(V_{\mathrm{MB}}(\mathbf{x}') - V_{\mathrm{MB}}(\mathbf{x}))\right)\right]$. The value of $\Delta r$ 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, $q_\alpha$ is set to be $\frac{\alpha-1}{M-1}$ and $\kappa_\alpha = 10000$ for all $\alpha$. For the path- or string-based umbrella sampling, we choose bias strength $\kappa_\alpha^{\|} = 1100$ 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. 11 and 12, the size of $\alpha$-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. 13–39, 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, $\lambda_{\mathrm{SL}}$ is increased linearly to 25000 at iteration 10000. Empirical committor values are collected at every 10 iterations of the algorithm, i.e., $\tau_{\text{emp}} = 10$. The initial and final iteration index where we start and end supervised learning is set to $k_{\text{emp},s} = 10$ and $k_{\text{emp},f} = 1000$. The number of trials for every window $H = 100$. The size of $\alpha$-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 [31], which takes in four hyper-parameters as inputs: the first is the step size/learning rate $\eta$, the second and third are momentum coefficients $\beta_1$ and $\beta_2$ that control the change in the momentum and momentum squared respectively, and the fourth parameter $\epsilon$ 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

$$\mathbf{m}_{k+1} = \beta_1 \mathbf{m}_k + (1-\beta_1)\nabla_{\boldsymbol{\theta}} f(\boldsymbol{\theta}_k) \,, \tag{B.5}$$

$$\mathbf{v}_{k+1} = \beta_2 \mathbf{v}_k + (1-\beta_2)\left[\nabla_{\boldsymbol{\theta}} f(\boldsymbol{\theta}_k) \odot \nabla_{\boldsymbol{\theta}} f(\boldsymbol{\theta}_k)\right] \,, \tag{B.6}$$

$$\hat{\mathbf{m}}_{k+1} = \frac{\mathbf{m}_k}{1-(\beta_1)^k} \,, \tag{B.7}$$

$$\hat{\mathbf{v}}_k = \frac{\mathbf{v}_k}{1-(\beta_2)^k} \,, \tag{B.8}$$

$$\mathbf{H}_{k+1} = \operatorname{diag}\left[\sqrt{\hat{\mathbf{v}}_k}\right] + \epsilon \,, \tag{B.9}$$

$$\boldsymbol{\theta}_{k+1} = \boldsymbol{\theta}_k - \eta(\mathbf{H}_{k+1})^{-1}\hat{\mathbf{m}}_{k+1} \,, \tag{B.10}$$

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.9) is applied element-wise to the vector, $\operatorname{diag}[\ldots]$ 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 $\lambda_{\mathrm{S}} = 0.1M$. 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.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. (2.14). 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. (2.38)-(2.39). In all algorithms, the friction coefficient $\gamma = 1$, and step size $\Delta t = 0.0001$. The size of $\alpha$-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, $\lambda_{\mathrm{SL}}$ 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 $\alpha$-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. 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 $\mu = 0.9$.

## C 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 [39], which utilizes the mean-squared error (MSE) loss function given by Eq. (2.22) instead of the supervised-learning loss given by Eq. (2.24). 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, except for the BKE–US+MSE method where $\lambda_{\mathrm{MSE}}$ 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. 28. 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. 29(a)) and ensemble-averaged BKE loss function (Fig. 29(b)) increase at higher iterations. This suggests that the MSE loss function is prone to overfitting [39], and we provide a sketch for why this occurs. To this end, the gradients of

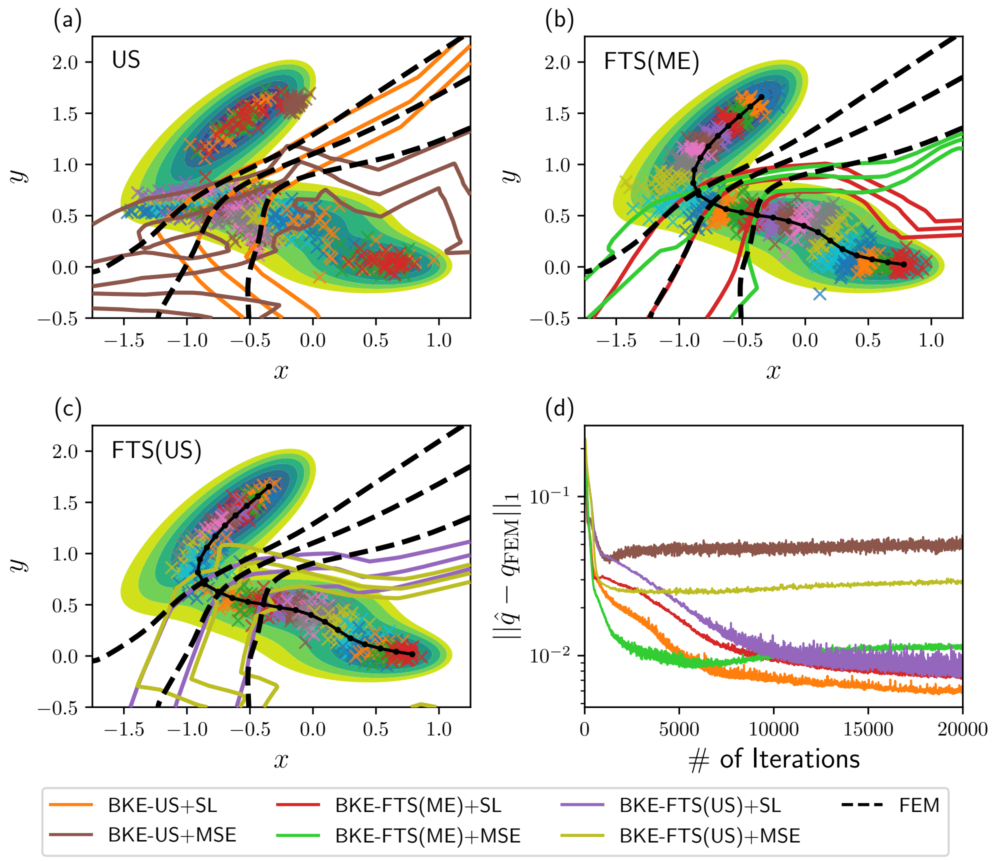


Figure 28: Isocommittor lines for $q = 0.1, 0.5,$ and $0.9$ from (a) the BKE–US+SL and BKE–US+MSE method, (b) the BKE–FTS(ME)+SL and BKE–FTS(ME)+SL method, (c) the BKE–FTS(US) and BKE–FTS(US)+MSE method. $\times$ markers denote representative samples obtained from algorithms the SL methods. (d) The $L_1$-norm error as a function of iterations.

the losses with respect to the neural network parameters are evaluated. For the MSE loss function in Eq. (2.22) this is

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

$$= \frac{\lambda_{\text{MSE}}}{M} \sum_{\alpha=1}^{M} \frac{1}{|\mathcal{C}_k^\alpha|} \sum_{(q_{\text{emp}}, \mathbf{x}) \in \mathcal{C}_k^\alpha} \left(\hat{q}(\mathbf{x}; \boldsymbol{\theta}) - q_{\text{emp}}\right) \nabla_{\boldsymbol{\theta}} \hat{q}(\mathbf{x}; \boldsymbol{\theta}) \,. \tag{C.2}$$

Note that the error $\hat{q}(\mathbf{x}; \boldsymbol{\theta}) - q_{\text{emp}}$ 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. (2.24) is

$$\nabla_{\boldsymbol{\theta}} \hat{L}_{\text{SL}}(\boldsymbol{\theta}; \{\mathcal{C}_k^\alpha\}) = \frac{\lambda_{\text{SL}}}{M} \sum_{\alpha=1}^{M} \nabla_{\boldsymbol{\theta}} \ell_{\text{ME}}(\mathcal{C}_k^\alpha; \boldsymbol{\theta}) \tag{C.3}$$

$$= \frac{\lambda_{\text{SL}}}{M} \sum_{\alpha=1}^{M} \left[ \frac{1}{|\mathcal{C}_k^\alpha|} \sum_{(q_{\text{emp}}, \mathbf{x}) \in \mathcal{C}_k^\alpha} \left(\hat{q}(\mathbf{x}; \boldsymbol{\theta}) - q_{\text{emp}}\right) \right] \left[ \frac{1}{|\mathcal{C}_k^\alpha|} \sum_{(q_{\text{emp}}, \mathbf{x}) \in \mathcal{C}_k^\alpha} \nabla_{\boldsymbol{\theta}} \hat{q}(\mathbf{x}; \boldsymbol{\theta}) \right] , \tag{C.4}$$

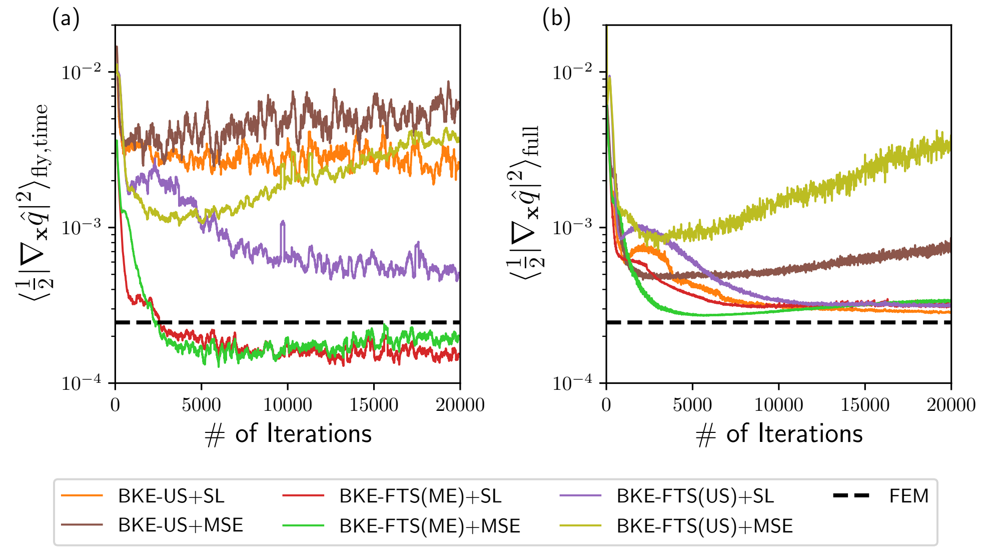


Figure 29: (a) The filtered on-the-fly estimate of the BKE loss obtained at every iteration, with the filtering window set to 200 iterations. (b) The ensemble-averaged loss per Eq. (3.10) obtained at every iteration.

where the error $\hat{q}(\mathbf{x};\boldsymbol{\theta}) - q_{\text{emp}}$ 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 $q_{\text{emp}}(\mathbf{x})$. To see this, we first write $q_{\text{emp}}(\mathbf{x})$ in terms of the exact committor function $q(\mathbf{x})$ as

$$q_{\text{emp}}(\mathbf{x}) = q(\mathbf{x}) + \epsilon(\mathbf{x})\,, \tag{C.5}$$

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 $\alpha$

$$\ell_{\text{ME}}(\mathcal{C}^\alpha;\boldsymbol{\theta}) = \frac{1}{2}\left[\frac{1}{|\mathcal{C}^\alpha|}\sum_{(q_{\text{emp}},\mathbf{x})\in\mathcal{C}^\alpha}(\hat{q}(\mathbf{x};\boldsymbol{\theta}) - q_{\text{emp}})\right]^2 \tag{C.6}$$

$$\approx \frac{1}{2}\left(\langle\hat{q}(\mathbf{x};\boldsymbol{\theta}) - q_{\text{emp}}(\mathbf{x})\rangle_\alpha\right)^2 \tag{C.7}$$

$$\approx \frac{1}{2}\left(\langle\hat{q}(\mathbf{x};\boldsymbol{\theta}) - q(\mathbf{x})\rangle_\alpha - \langle\epsilon(\mathbf{x})\rangle_\alpha\right)^2 \tag{C.8}$$

$$\approx \frac{1}{2}\left(\langle\hat{q}(\mathbf{x};\boldsymbol{\theta}) - q(\mathbf{x})\rangle_\alpha\right)^2\,, \tag{C.9}$$

where $\langle\ldots\rangle_\alpha$ is the ensemble average with respect to replica $\alpha$. 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 $q_{\text{emp}}(\mathbf{x})$.

Returning to the gradient of the supervised learning loss function given in Eq. (C.4), we have for large mini-batch sizes

$$\nabla_{\boldsymbol{\theta}}\hat{L}_{\text{SL}}(\boldsymbol{\theta};\{\mathcal{C}_k^\alpha\}) \approx \frac{\lambda_{\text{SL}}}{M}\sum_{\alpha=1}^{M}\langle\hat{q}(\mathbf{x};\boldsymbol{\theta}) - q(\mathbf{x})\rangle_\alpha\langle\nabla_{\boldsymbol{\theta}}\hat{q}(\mathbf{x};\boldsymbol{\theta})\rangle_\alpha\,, \tag{C.10}$$

in which the replica average of the gradient is coupled to a noise-reduced measure of the error. A global minimum is achieved when

$$\langle\hat{q}(\mathbf{x};\boldsymbol{\theta})\rangle_\alpha = \langle q(\mathbf{x})\rangle_\alpha \quad \forall\alpha\,. \tag{C.11}$$

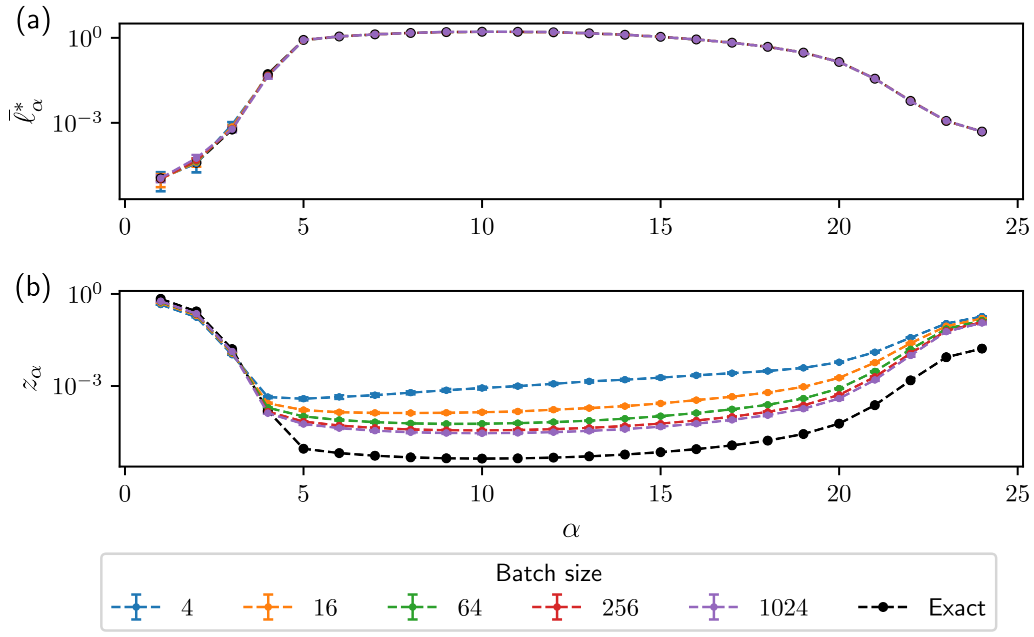


Figure 30: (a) The sample mean of the BKE loss from each replica $\bar{\ell}^*_\alpha = \frac{1}{|\mathcal{M}^\alpha_k|}\sum_{\mathbf{x}\in\mathcal{M}^\alpha_k}\frac{\ell(\mathbf{x};\boldsymbol{\theta}_k)}{c(\mathbf{x};\boldsymbol{\theta}_k)}$, and (b) the estimated reweighting factor $z_\alpha$ from each replica $\alpha$, 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.

While this condition can be satisfied for $\hat{q}(\mathbf{x};\boldsymbol{\theta}) \neq q(\mathbf{x})$ in the region sampled by replica $\alpha$, the additional loss terms in Eq. (2.12) 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.

## D Examining the Sampling Error in Reweighting Factors

In this section, we examine how sampling error in the reweighting factors $z_\alpha$ estimated from all algorithms is reduced in the limit of large batch sizes. For a given batch size, $z_\alpha$ is computed over many iterations of each algorithm while keeping the neural network fixed. Afterwards, the mean of $z_\alpha$ computed from all iterations is compared to the $z_\alpha$ computed from numerical integration of Eq. (2.16) for umbrella sampling, and Eq. (2.47) for the master-equation approach.

Figure 30(b) shows $z_\alpha$ 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 $z_\alpha$ 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. 30(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 31 shows both $z_\alpha$ and the sample mean of the BKE loss from each replica $\bar{\ell}_\alpha = \frac{1}{|\mathcal{R}^\alpha_k|}\sum_{\mathbf{x}\in\mathcal{R}^\alpha_k}\ell(\mathbf{x};\boldsymbol{\theta}_k)$, 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 $z_\alpha$, which is

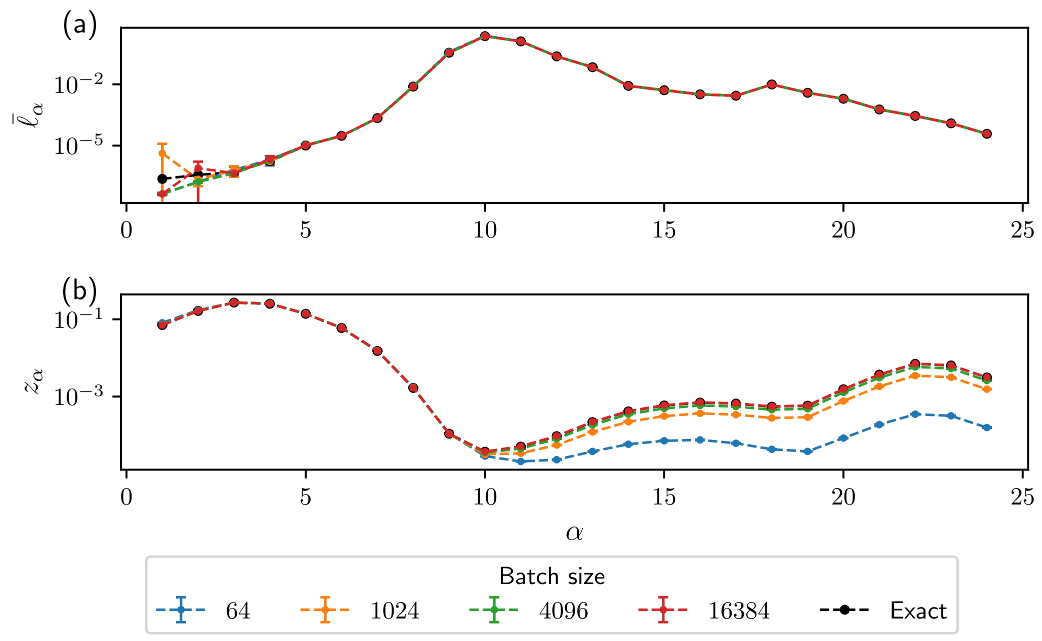


Figure 31: (a) The sample mean of the BKE loss from each replica $\bar{\ell}_\alpha = \frac{1}{|\mathcal{R}_k^\alpha|}\sum_{\mathbf{x}\in\mathcal{R}_k^\alpha} \ell(\mathbf{x};\boldsymbol{\theta}_k)$, and (b) the estimated reweighting factor $z_\alpha$ from each replica $\alpha$, in comparison to the 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.

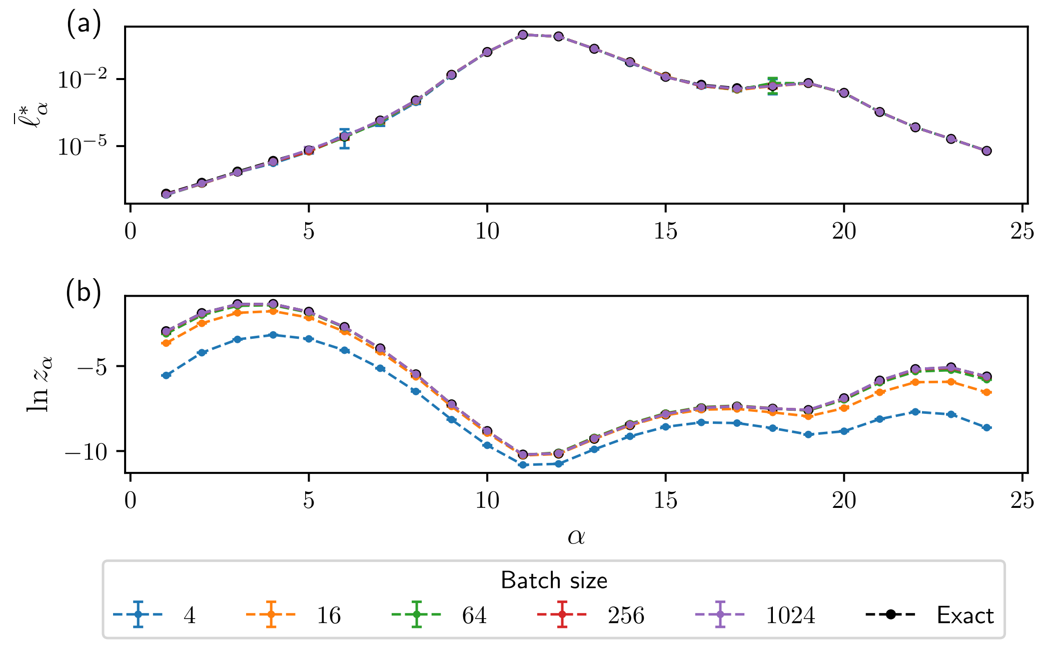


Figure 32: (a) The sample mean of the BKE loss from each replica $\bar{\ell}^*_\alpha = \frac{1}{|\mathcal{M}_k^\alpha|}\sum_{\mathbf{x}\in\mathcal{M}_k^\alpha} \frac{\ell(\mathbf{x};\boldsymbol{\theta}_k)}{c(\mathbf{x};\boldsymbol{\theta}_k)}$, and (b) the estimated reweighting factor $\ln z_\alpha$ from each replica $\alpha$, 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.

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, $z_\alpha$ computed from path-based umbrella sampling (Fig. 32(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. 32(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.

## E 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 3.3 for all replicas. The histograms for the forward free-energy differences are given in Fig. 33. The histograms for the backward free-energy differences are given in Fig. 34. The histograms for the reweighting factors are given in Fig. 35. The histograms for $\ln \bar{\ell}^*_\alpha$ and $\ln \bar{1}^*_\alpha$ are given in Figs. 36 and 37, respectively. The histograms for $\ln z_\alpha \bar{\ell}^*_\alpha$ and $\ln z_\alpha \bar{1}^*_\alpha$ are given in Figs. 38 and 39, 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 [54], 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. 40 and 41 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 to $\kappa^{\parallel}_\alpha = 2200$ and $\kappa^{\perp}_\alpha = 300$, 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.

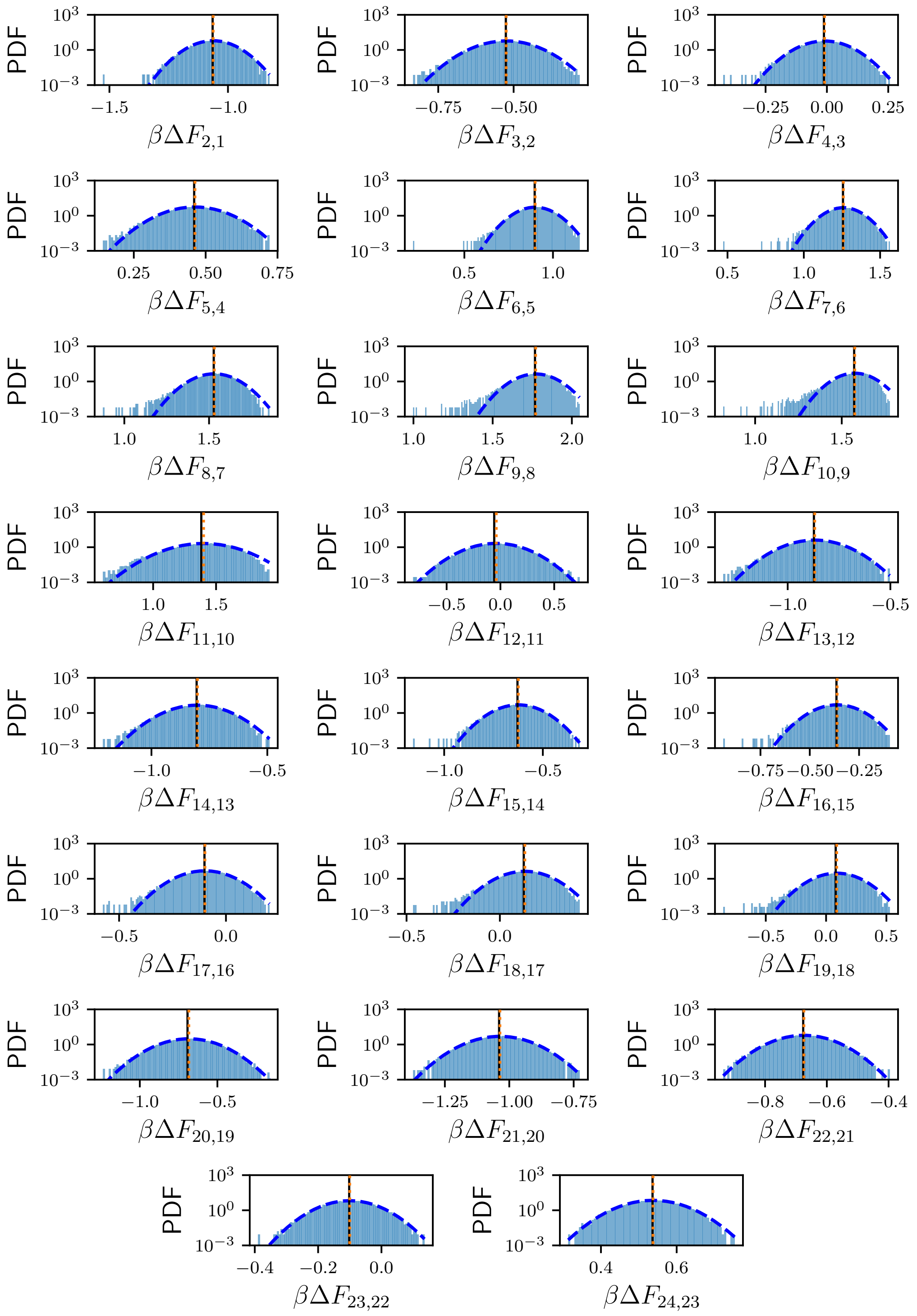


Figure 33: Probability density functions of the forward free-energy differences $\beta\Delta F_{(\alpha+1),\alpha}$.

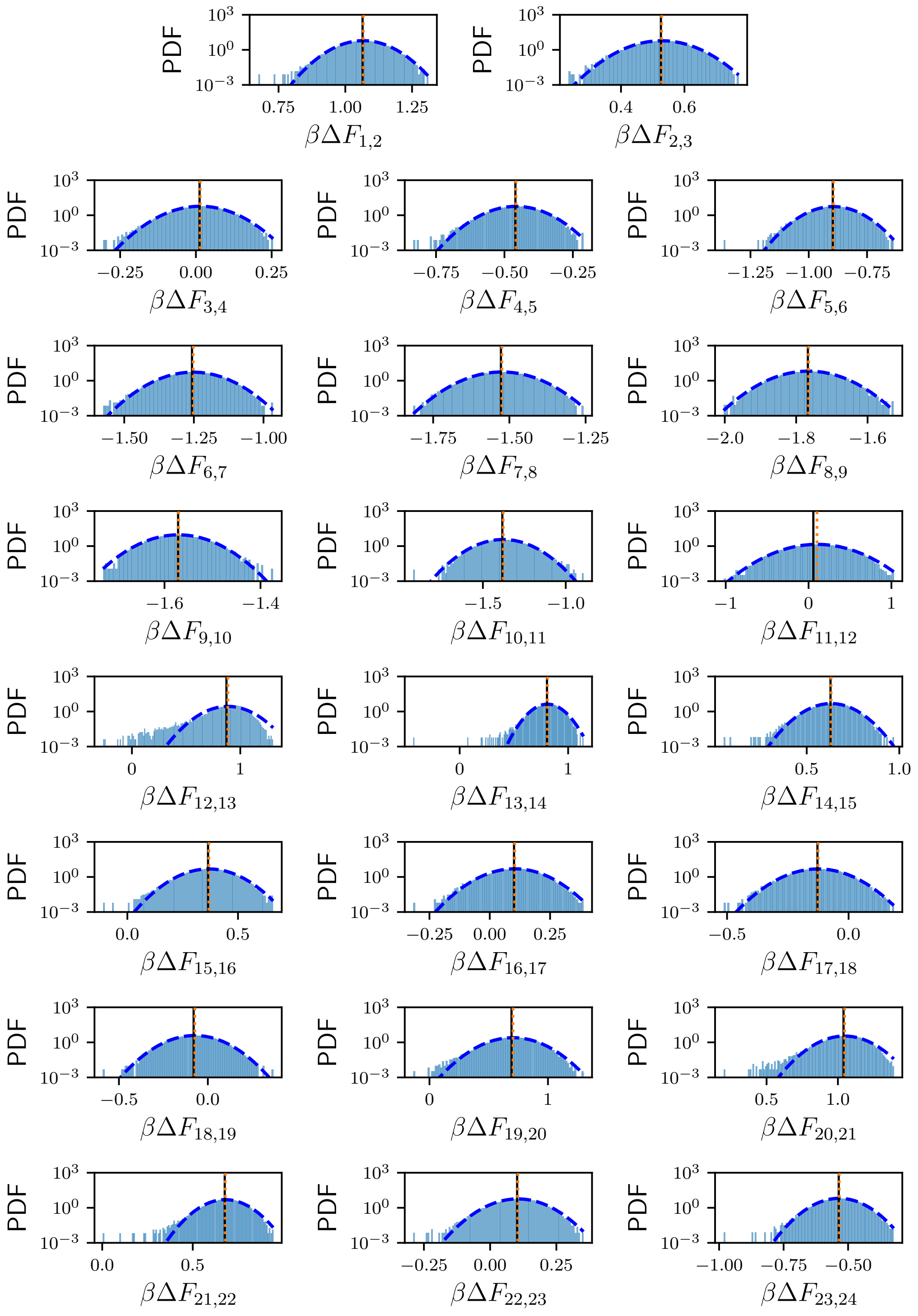


Figure 34: Probability density functions of the backward free-energy differences $\beta\Delta F_{(\alpha-1),\alpha}$.

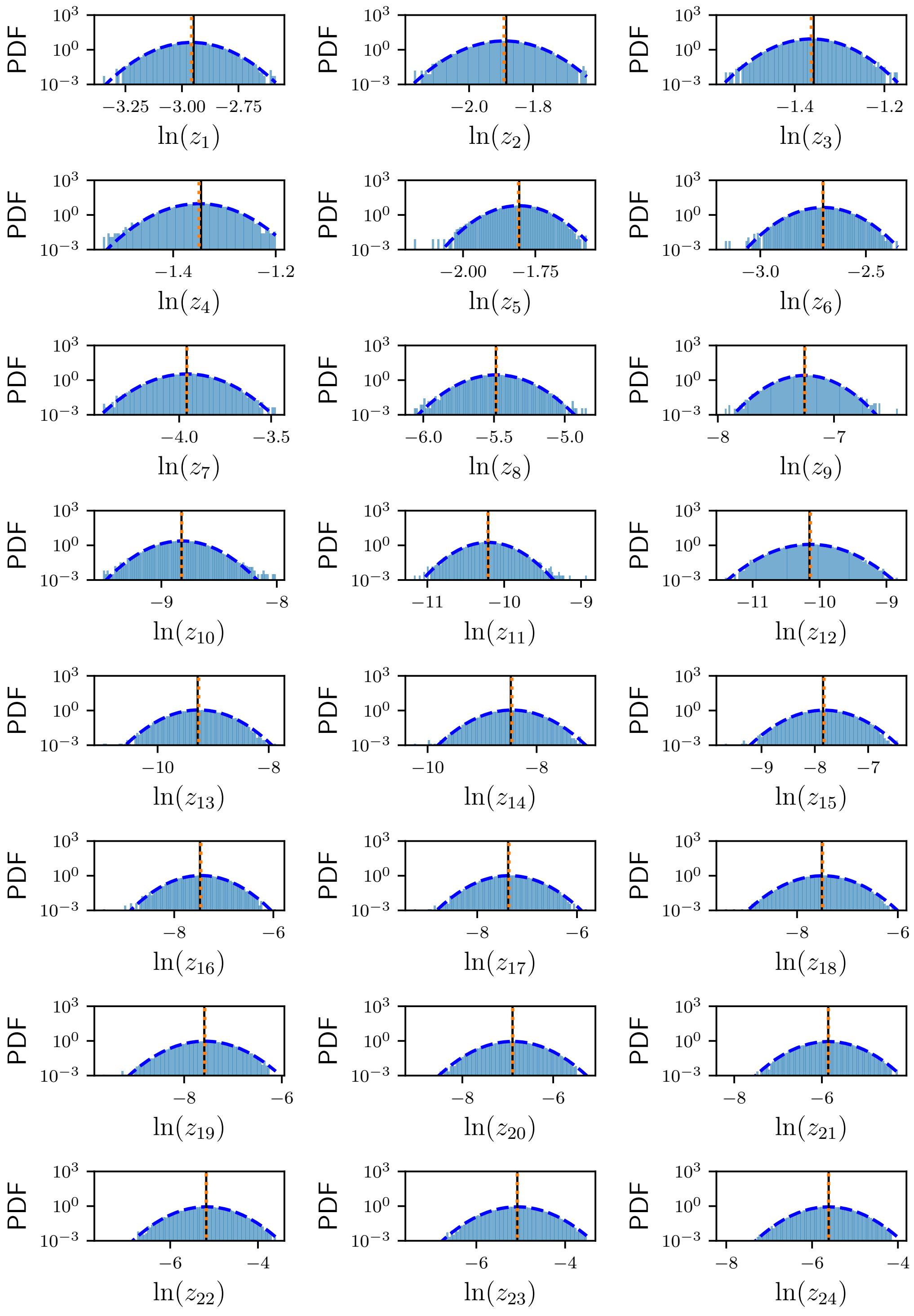


Figure 35: Probability density functions of the log of reweighting factors $\ln z_\alpha$.

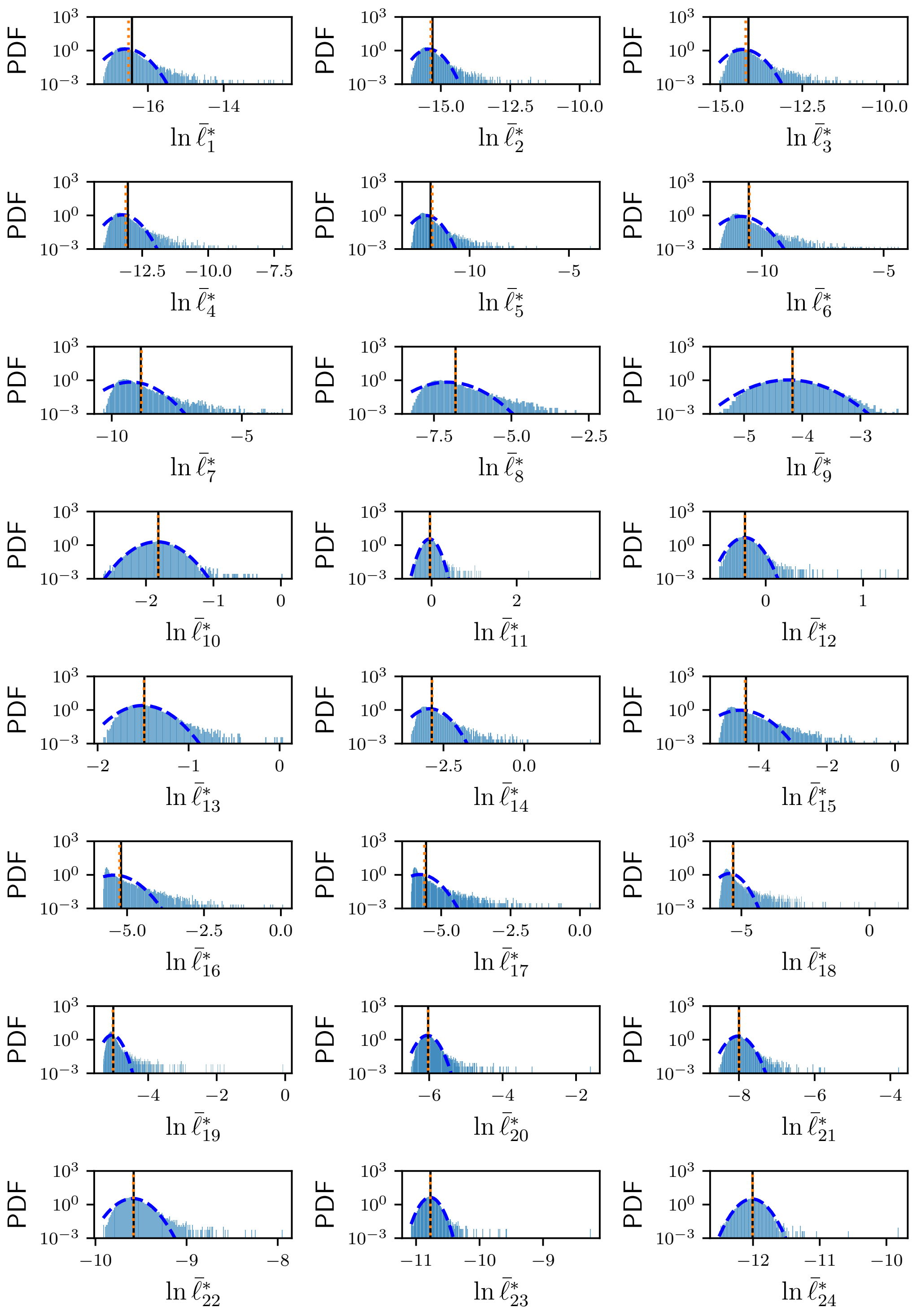


Figure 36: Probability density functions of $\ln \bar{\ell}^*_\alpha$.

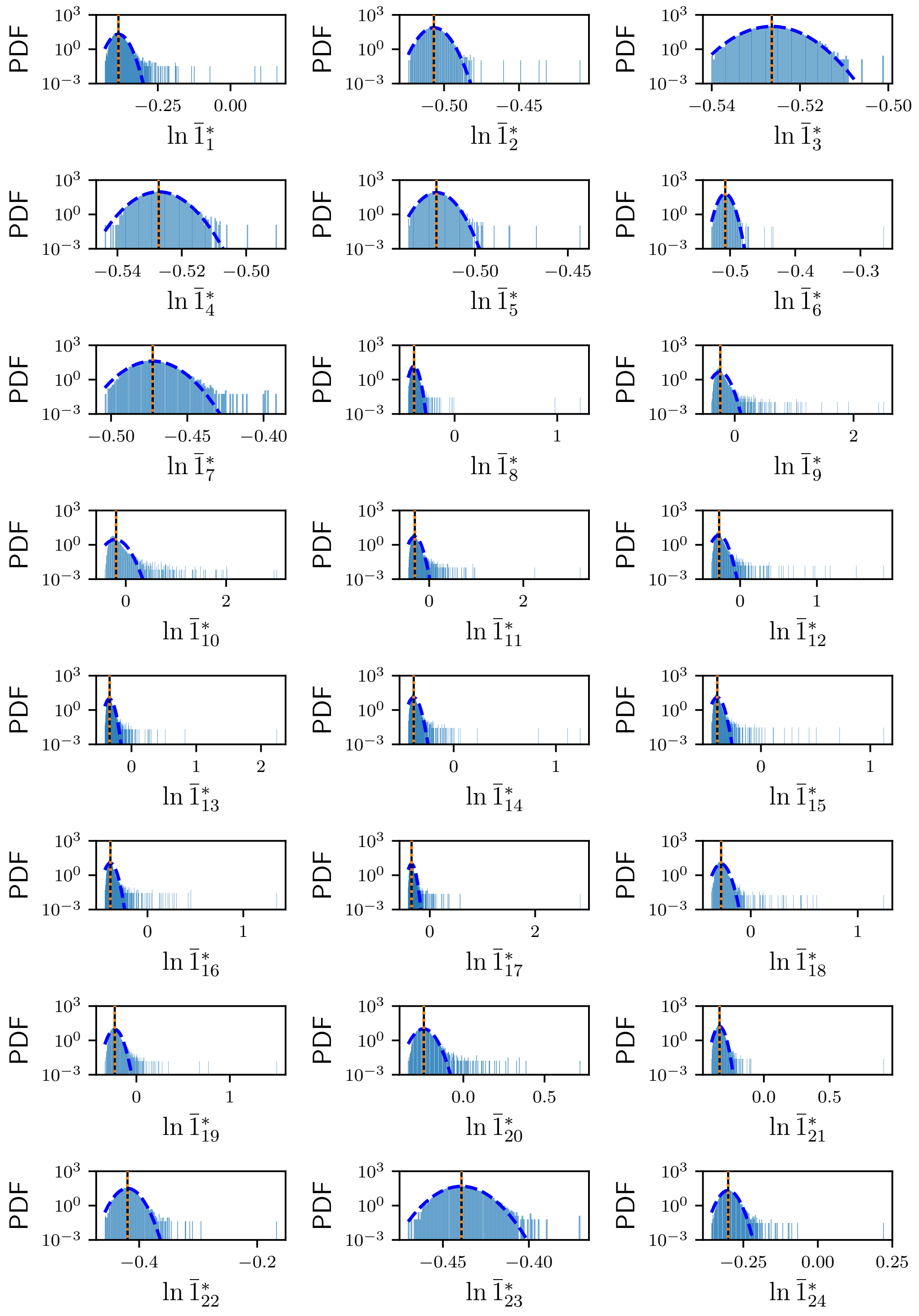


Figure 37: Probability density functions of $\ln \bar{1}^*_\alpha$.

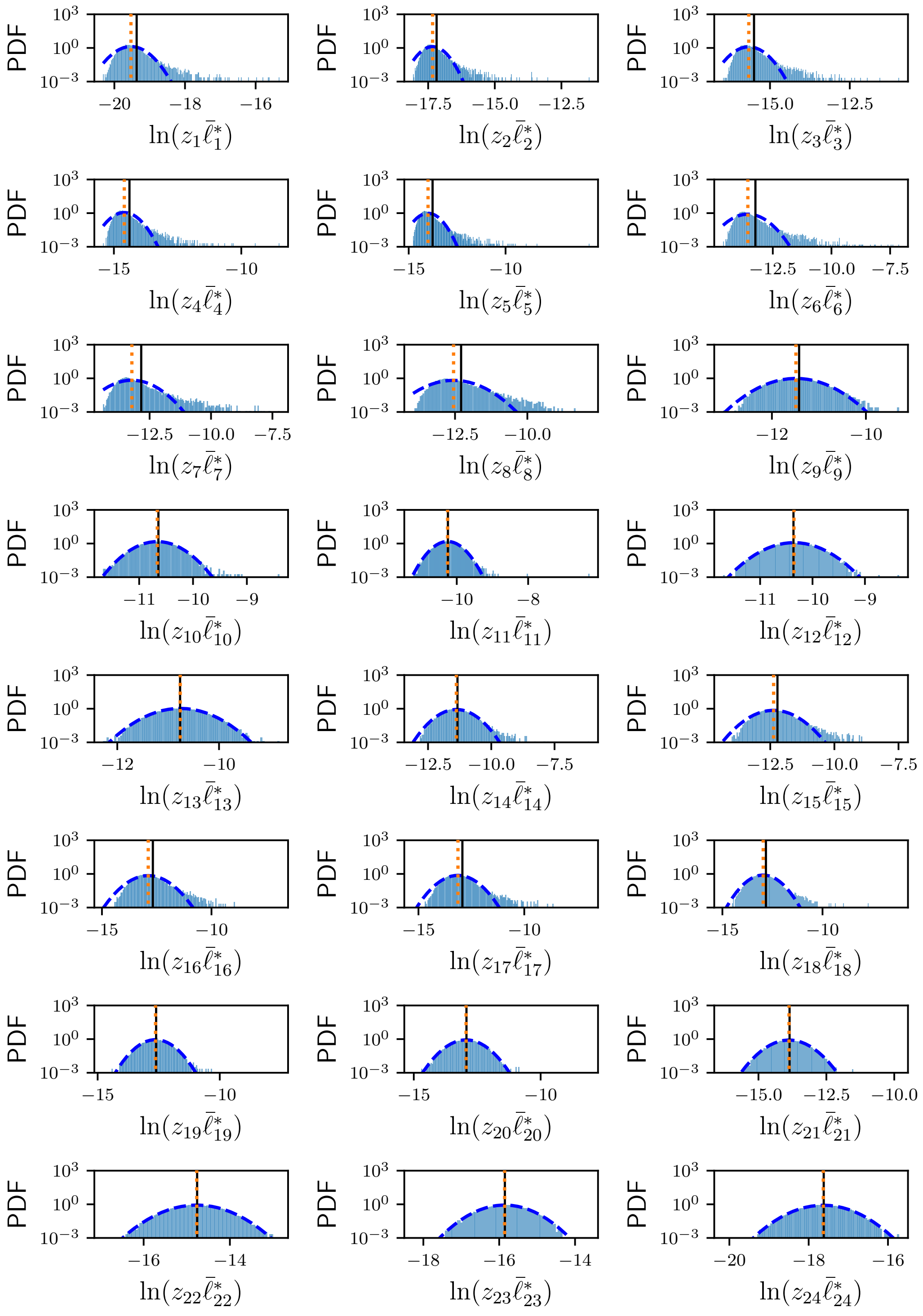


Figure 38: Probability density functions of $\ln z_\alpha \bar{\ell}_\alpha^*$.

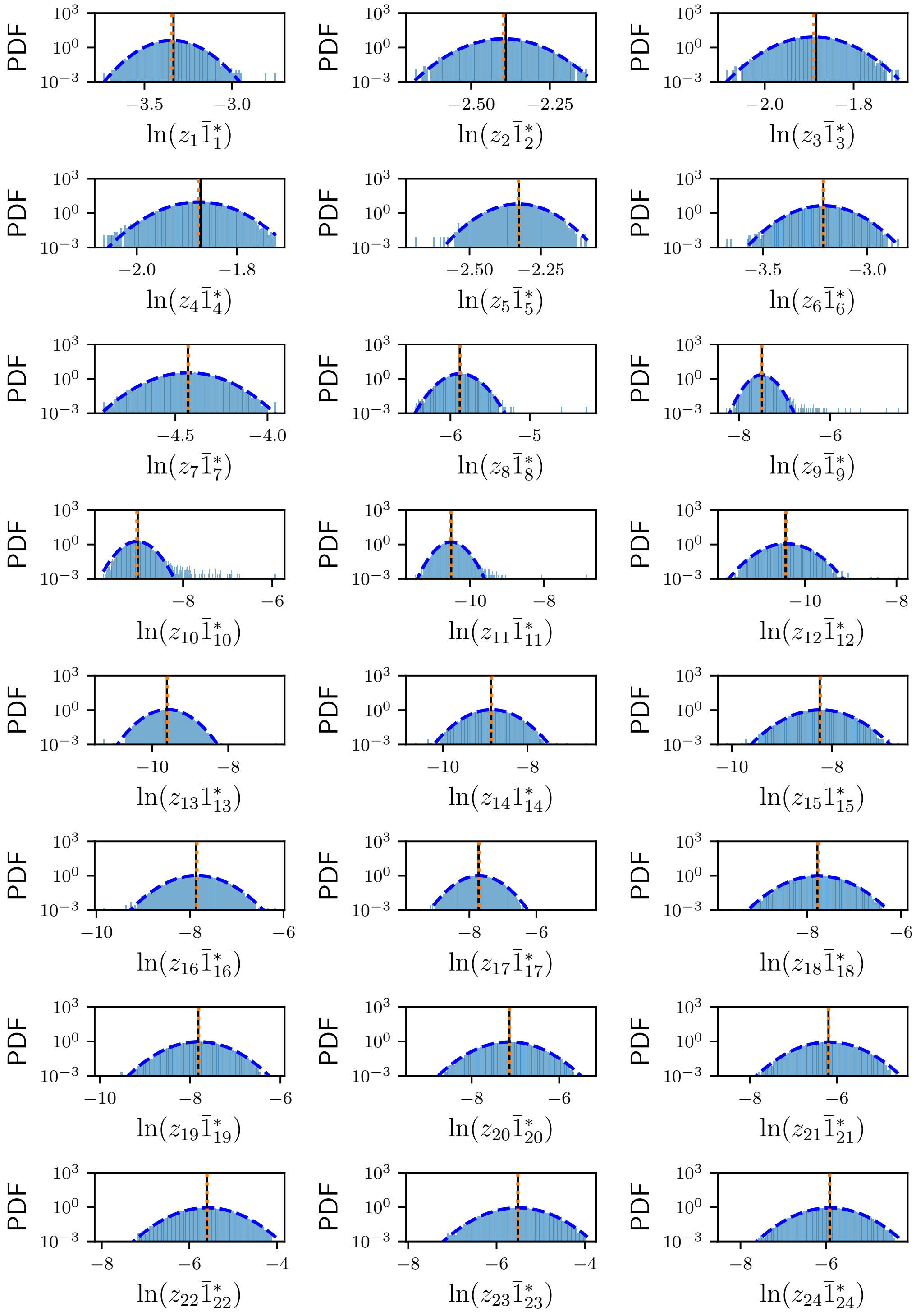


Figure 39: Probability density functions of $\ln z_\alpha \bar{1}^*_\alpha$.

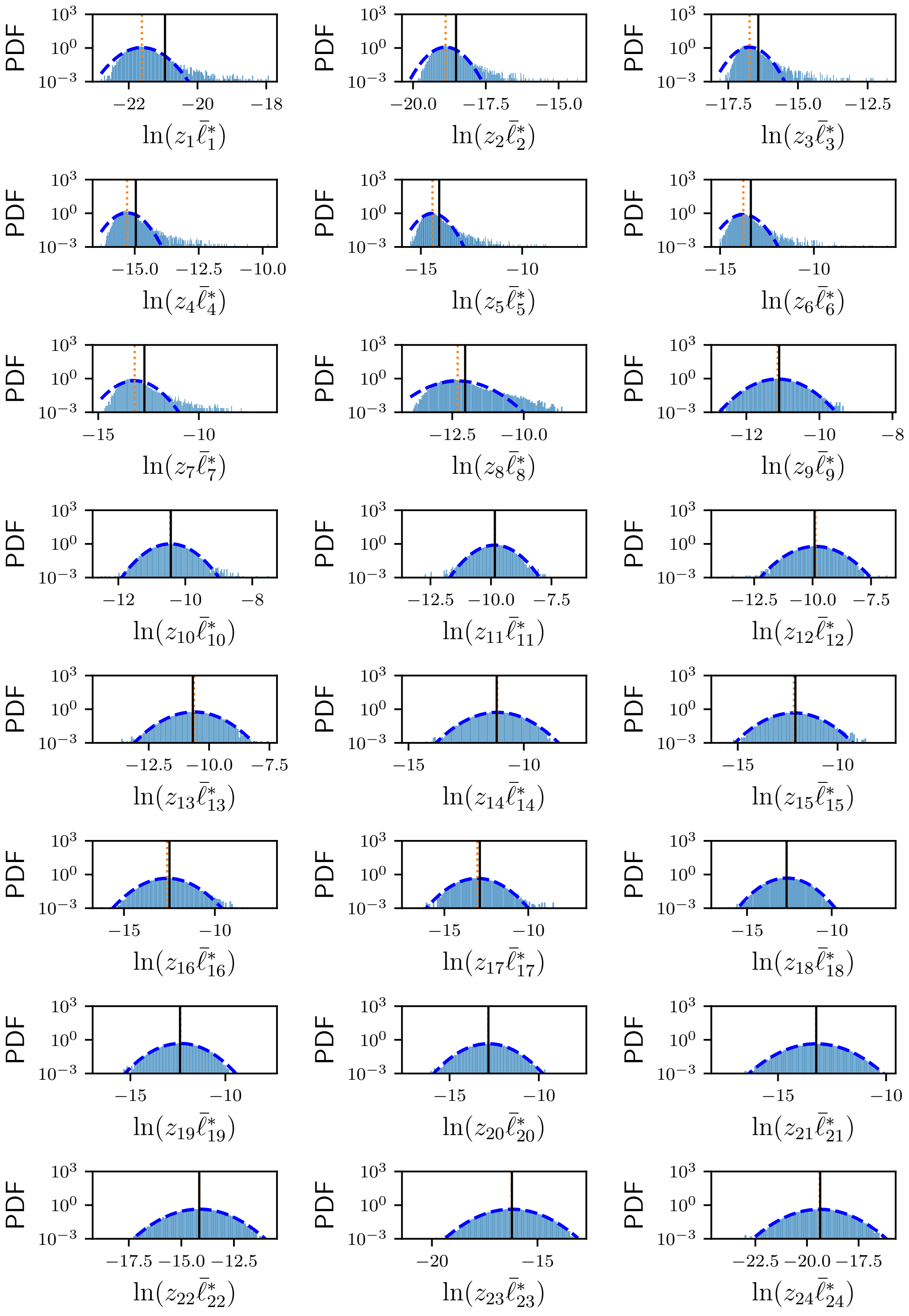


Figure 40: Probability density functions of $\ln z_\alpha \bar{\ell}^*_\alpha$, where data is obtained from umbrella sampling with bias strengths $\kappa^{\parallel}_\alpha = 2200$ and $\kappa^{\perp}_\alpha = 300$.

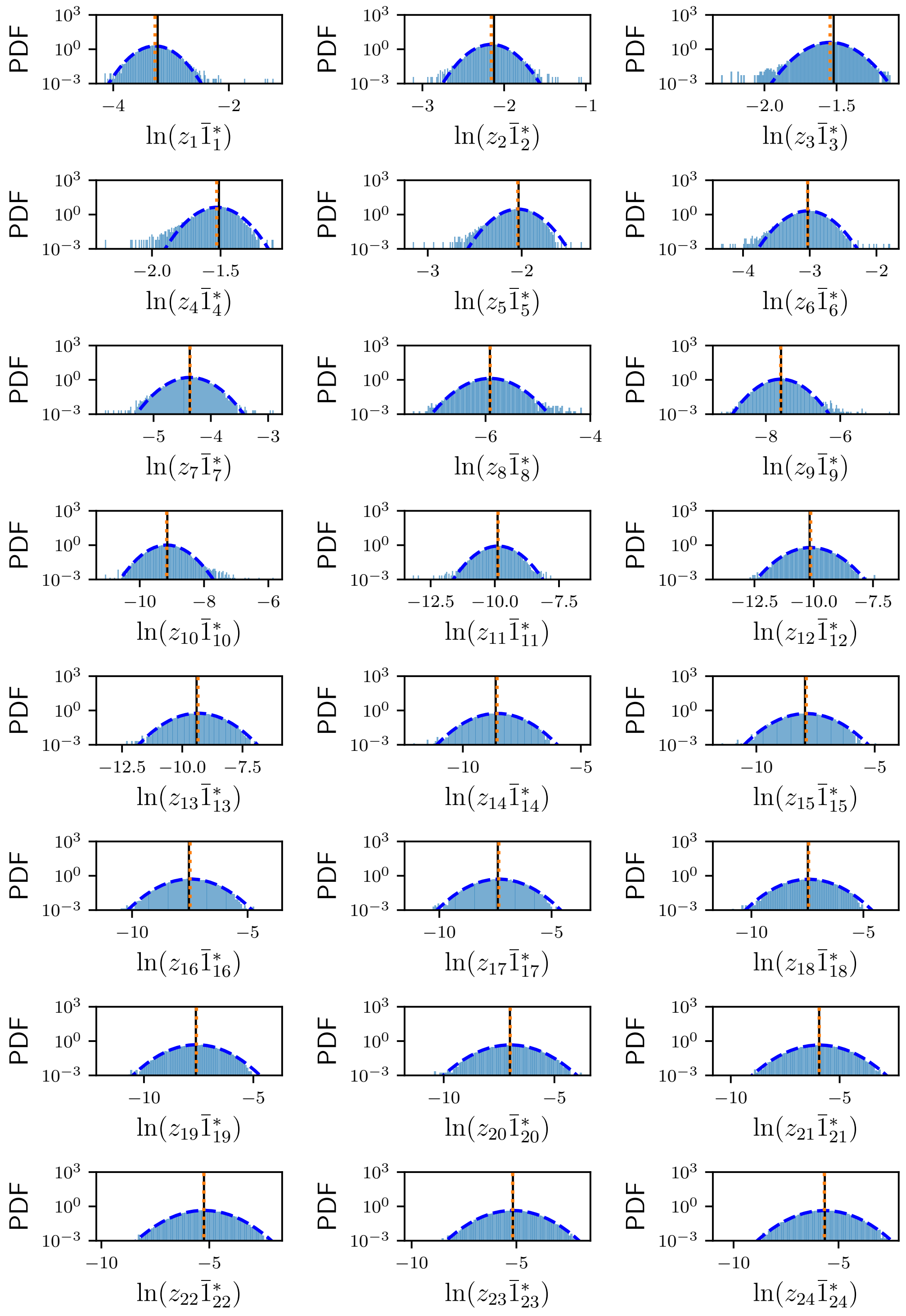


Figure 41: Probability density functions of $\ln z_\alpha \bar{1}^*_\alpha$, where data is obtained from umbrella sampling with bias strengths $\kappa^{\parallel}_\alpha = 2200$ and $\kappa^{\perp}_\alpha = 300$.