Contents lists available at ScienceDirect

Journal of Computational Physics

journal homepage: www.elsevier.com/locate/jcp

Estimating committor functions via deep adaptive sampling on rare transition paths

Yueyang Wang[a], Kejun Tang[b,c,*], Xili Wang[a], Xiaoliang Wan[d], Weiqing Ren[e], Chao Yang[a]

[a] School of Mathematical Sciences, Peking University, China
[b] Faculty of Computility Microelectronics, Shenzhen University of Advanced Technology, China
[c] School of Sciences, Great Bay University, China
[d] Department of Mathematics and Center for Computation and Technology, Louisiana State University, USA
[e] Department of Mathematics, National University of Singapore, Singapore

ARTICLE INFO

Keywords:
Committor function
Deep adaptive sampling
Rare event
Transition path

ABSTRACT

The committor function is central to investigating rare but critical events in molecular simulations. However, computing the committor function suffers from the curse of dimensionality. Recently, using neural networks to estimate the committor function has gained attention due to its potential for high-dimensional problems. Training neural networks to approximate the committor function requires sampling transition data from straightforward simulations of rare events, which is highly inefficient. The scarcity of transition data poses a significant challenge for accurately approximating the committor function. To address this issue, we propose an efficient framework to generate data points in the transition state region, facilitating the effective training of neural networks to approximate the committor function. We introduce a Deep Adaptive Sampling method for TRansition paths (DASTR), where deep generative models are employed to generate samples that effectively capture the transition information. Specifically, we treat a non-negative function in the integrand of the loss functional as an unnormalized probability density function and approximate it using a deep generative model. The resulting samples from the deep generative model are concentrated in the transition state region, with fewer samples in other regions. This distribution provides effective samples for approximating the committor function, significantly improving accuracy. We demonstrate the effectiveness of the proposed method through both simulations and realistic examples.

1. Introduction

Understanding transition events between metastates in a stochastic system plays a central role in chemical reactions and statistical physics [1–4]. The physical process can be formulated as the following stochastic differential equation (SDE)

$$d\boldsymbol{X}_t = -\nabla V(\boldsymbol{X}_t)dt + \sqrt{2\beta^{-1}}d\boldsymbol{W}_t, \tag{1}$$

where $\boldsymbol{X}_t \in \Omega \subset \mathbb{R}^d$ is the state of the system at time t, $V : \Omega \mapsto \mathbb{R}$ denotes a potential function, β the inverse temperature, and $\boldsymbol{W}_t$ the standard d-dimensional Wiener process. For two disjoint subsets of this stochastic system, we are interested in the transition rate,

[*] Corresponding author.
E-mail addresses: wangyueyang@stu.pku.edu.cn (Y. Wang), tangkj@gbu.edu.cn (K. Tang), xiliwang@stu.pku.edu.cn (X. Wang), xlwan@lsu.edu (X. Wan), matrw@nus.edu.sg (W. Ren), chao_yang@pku.edu.cn (C. Yang).

https://doi.org/10.1016/j.jcp.2025.114439
Received 26 January 2025; Received in revised form 7 July 2025; Accepted 7 October 2025
Available online 10 October 2025

which can be characterized by the *committor function*. For two distinct metastable regions $A, B \subset \Omega$, and $A \cap B = \emptyset$, denoting by τ_ω the first hitting time of a subset $\omega \subset \Omega$ for a trajectory, the committor function $q : \Omega \mapsto [0, 1]$ is defined as $q(\boldsymbol{x}) = \mathbb{P}\left(\tau_B < \tau_A | \boldsymbol{X}_0 = \boldsymbol{x}\right)$, where $\mathbb{P}$ denotes the probability. The committor function is a probability that a trajectory of SDE starting from $\boldsymbol{x} \in \Omega$ first reaches B rather than A. By definition, it is easy to see that $q(\boldsymbol{x}) = 0$ for $\boldsymbol{x} \in A$ and $q(\boldsymbol{x}) = 1$ for $\boldsymbol{x} \in B$. This committor function provides the information of the transition process, and it is governed by the following partial differential equation (PDE) [5,6]

$$\begin{aligned} -\beta^{-1}\Delta q(\boldsymbol{x}) + \nabla V(\boldsymbol{x}) \cdot \nabla q(\boldsymbol{x}) = 0, &\quad \boldsymbol{x} \in \Omega\backslash(A \cup B), \\ q(\boldsymbol{x}) = 0, &\quad \boldsymbol{x} \in A, \\ q(\boldsymbol{x}) = 1, &\quad \boldsymbol{x} \in B, \\ \nabla q(\boldsymbol{x}) \cdot \boldsymbol{n} = 0, &\quad \boldsymbol{x} \in \partial\Omega\backslash(A \cup B), \end{aligned} \tag{2}$$

where $\boldsymbol{n}$ is the outward unit normal vector of the boundary $\partial\Omega\backslash(A \cup B)$. Once the committor function $q(\boldsymbol{x})$ is found, we can use it to extract the statistical information of reaction trajectories [2,4] and compute transition rates.

1.1. Connections with prior work and contributions

Obtaining the committor function q needs to solve the above high-dimensional PDE, which is computationally infeasible for traditional grid-based numerical methods. In Chen et al. [7], a low-rank tensor train approach is proposed to compute the committor function, which relies on the low-rank tensor train approximation of the Boltzmann-Gibbs distribution. This approach cannot be directly applied to realistic problems if no explicit low-rank tensor train formats for the potential are given. Some efforts have been made to employ deep neural networks to approximate the committor function [6,8–12]. The key idea is that committor functions are represented by deep neural networks that can be trained by minimizing a variational loss functional [6,8] or a residual loss functional [11,12]. The training data points for discretizing the variational loss are usually sampled from the equilibrium distribution of the SDE (i.e. the Gibbs measure) [8,13,14], which requires simulating the stochastic differential equations. This sampling method is inefficient due to the scarcity of transition data, especially for realistic systems at low temperatures. Modified sampling methods are proposed in Li et al. [6], Rotskoff et al. [15], Hasyim et al. [16], Kang et al. [17], Lin and Ren [18], Singh et al. [19], Das et al. [20], Singh and Limmer [21,22] to alleviate this issue, where a new probability measure for sampling is constructed by modifying the potential function so that more samples can be obtained in the transition state region.

When the transition is rare, samples from the transition state region are difficult to obtain from simulating the SDE [15,17]. If insufficient data points are located on the transition paths, the trained neural network for approximating the committor function will have a large generalization error. To address this problem, we propose a new framework called Deep Adaptive Sampling on rare TRansition paths (DASTR) to train the deep neural network. More specifically, we generate samples in the transition state region using an iterative construction. To do this, we define a proper sampling distribution using both the current approximate committor function and the potential function, in contrast to merely modifying the potential function as in Li et al. [6], Rotskoff et al. [15], Hasyim et al. [16], Kang et al. [17], Lin and Ren [18]. The key idea is to reveal the transition information by taking into account the properties of the committor function. Unlike the methods based on local approximation of the committor and the SDE [23,24], the new sampling distribution is approximated by a deep generative model based on which new samples are generated and added to the training set. Once the training set is updated, the neural network model for approximating the committor function is further trained for refinement. This procedure is repeated to form a deep adaptive sampling approach on rare transition paths.

It is challenging to deal with high-dimensional realistic problems using deep generative models because we need to ensure two things: one is that more samples are located in the transition state region, and the other is that all samples must obey the molecular configurations. Directly approximating and sampling a high-dimensional distribution may result in a relatively large number of samples with unreasonable molecular configurations, which limits the application of DASTR. To deal with this issue, we combine the proposed DASTR method with dimension reduction techniques to automatically select the collective variables (CVs), where an autoencoder is trained to help avoid hand-craft selections of collective variables. Such a dimension reduction step helps avoid generating physically unreasonable configurations, thereby not only reducing computational complexity but also enhancing sampling efficiency. To summarize, the main contributions of this work are as follows:

- We propose a general framework, called deep adaptive sampling on rare transition paths (DASTR), for estimating high-dimensional committor functions.
- For high-dimensional realistic problems, the proposed DASTR method can be applied to the latent collective variables obtained by an autoencoder without hand-picking. One can reduce computational complexity and enhance sampling efficiency by adaptive sampling in the latent space. We demonstrate the effectiveness of the proposed method with the alanine dipeptide problem.

1.2. Related work

1.2.1. Adaptive sampling of neural network solver

The basic idea of adaptive sampling involves utilizing a non-negative error indicator, such as the residual square, to refine collocation points in the training set. Sampling approaches [25] (e.g., Markov Chain Monte Carlo) or deep generative models [26–28] are often invoked to sample from the distribution induced by the error indicator. Typically, an additional deep generative model (e.g., normalizing flow models) or a classical model (e.g., Gaussian mixture models [29,30]) for sampling is required. This work uses the variational formulation and defines a novel indicator for adaptive sampling by incorporating the traits of committor functions.

1.2.2. Autoencoder for protein systems

As a dimension reduction technique, autoencoders have shown the potential for the protein structure prediction and generation [31]. Autoencoders compress the input data into a lower-dimensional latent space and then reconstruct the input data through a decoder, enabling the learning of underlying features in the data. This approach not only helps reduce the computational resources needed for protein simulations but also significantly lowers the dimensionality and complexity of the problem. The prediction and generation of new protein structures can also be assisted by analyzing the variables in the latent space [32–34]. In our framework, the deep generative model can be used in the latent space to adaptively generate latent variables, which helps us explore the transition paths more efficiently and avoid selecting collective variables by hand-picking.

1.3. Organization

The rest of the paper is organized as follows. Details of neural network methods for computing committor functions are introduced in Section 2. Our DASTR approach is presented in Section 3. In Section 4, we demonstrate the effectiveness of our DASTR approach with numerical experiments. Finally, we conclude in Section 5.

2. Neural network solver for committor functions

The neural network approximation of partial differential equations involves minimizing a proper loss functional, e.g., the residual loss [35–37] or the variational loss [38–40]. For the committor function, we consider the variational loss [6] instead of the residual loss. The variational loss involves up to first-order derivatives of the committer function, while the residual loss requires computing the second-order derivatives. In other words, computing the residual loss is more expensive, especially for high dimensional problems (large d in (2)). Let $q_\theta(\boldsymbol{x})$ be a neural network parameterized with θ, where the input of the neural network is the state variable $\boldsymbol{x}$. One can solve the following variational problem to approximate the committor function

$$\begin{aligned}&\min_\theta \int_{\Omega\backslash(A\cup B)} |\nabla q_\theta(\boldsymbol{x})|^2 e^{-\beta V(\boldsymbol{x})} d\boldsymbol{x},\\ &\text{s.t. } q_\theta(\boldsymbol{x}) = 0, \boldsymbol{x}\in A; q_\theta(\boldsymbol{x}) = 1, \boldsymbol{x}\in B.\end{aligned} \tag{3}$$

The details of the derivation of (3) can be found in Appendix A. We then obtain the following unconstrained optimization problem by adding a penalty term

$$\min_\theta \int_{\Omega\backslash(A\cup B)} |\nabla q_\theta(\boldsymbol{x})|^2 e^{-\beta V(\boldsymbol{x})} d\boldsymbol{x} + \lambda\left(\int_A q_\theta(\boldsymbol{x})^2 p_A(\boldsymbol{x}) d\boldsymbol{x} + \int_B (1-q_\theta(\boldsymbol{x}))^2 p_B(\boldsymbol{x}) d\boldsymbol{x}\right), \tag{4}$$

where $\lambda > 0$ is a penalty parameter, p_A and p_B are two probability density functions on A and B respectively.

To optimize the above variational problem, one needs to generate some random collocation points from a proper probability distribution to estimate the integral in (3). One choice is to sample collocation points from the Gibbs measure $e^{-\beta V(\boldsymbol{x})}/Z$, where $Z = \int_{\Omega\backslash(A\cup B)} e^{-\beta V(\boldsymbol{x})} d\boldsymbol{x}$ is the normalization constant, and this can be done by simulating the SDE defined in (1). However, generating collocation points by the SDE is inefficient for approximating the committor function, especially for molecular systems with low temperatures (or high energy barriers). This is because the committor function is characterized by the behavior in the transition area while the samples generated by the Langevin dynamics (Eq. (1)) cluster around the metastable regions A and B. In other words, the samples from the SDE may not include sufficient effective samples for training q_θ. Hence, we need a strategy to seek more effective samples to approximate the committor function, which will be presented in the next section.

Now suppose that we have a set of collocation points $\mathsf{S} = \{\boldsymbol{x}_i\}_{i=1}^N$, where each $\boldsymbol{x}_i \in \Omega\backslash(A\cup B)$ is drawn from a certain probability distribution p, and two sets of collocation points $\mathsf{S}_A = \{\boldsymbol{x}_{A,i}\}_{i=1}^{N_A}$ and $\mathsf{S}_B = \{\boldsymbol{x}_{B,i}\}_{i=1}^{N_B}$, where each $\boldsymbol{x}_{A,i}$ and each $\boldsymbol{x}_{B,i}$ are drawn from p_A and p_B respectively. The optimization problem (4) can be discretized as follows

$$\min_\theta \frac{1}{N}\sum_{i=1}^N |\nabla q_\theta(\boldsymbol{x}_i)|^2 \frac{e^{-\beta V(\boldsymbol{x}_i)}}{p(\boldsymbol{x}_i)} + \lambda\left(\frac{1}{N_A}\sum_{i=1}^{N_A} q_\theta(\boldsymbol{x}_{A,i})^2 + \frac{1}{N_B}\sum_{i=1}^{N_B}(q_\theta(\boldsymbol{x}_{B,i})-1)^2\right). \tag{5}$$

The key point here is to choose an effective set S to train q_θ. In the next section, we will show how to adaptively generate effective collocation points (a high-quality dataset) on rare transition paths, based on which we can improve the accuracy of the approximate solution of (2). Considering that the main difficulties come from the transition state region, we will focus on how to choose S and assume that the integral on the boundary is well approximated by two prescribed sets S_A and S_B. For simplicity, we will ignore the penalty term when discussing our method.

3. Deep adaptive sampling on rare transition paths

3.1. Main idea

Our goal is to adaptively generate more effective data points distributed in the transition state region, which will be achieved by constructing a deep adaptive sampling method on the transition paths.

Suppose that at the k-th step, we have obtained the current approximate solution q_{θ_k} with S_k. We want to use the information of q_{θ_k} and the potential function V to detect where the transition area is, based on which we expect to generate new data points in the transition state region that can help improve the discretized loss given by S_k. We then refine S_k to get S_{k+1} for the next training step. The more effective data points in the transition area we have, the more accurate solution q_θ we can obtain. To achieve this, we define a proper probability distribution for sample generation based on the following observations: First, $|\nabla_x q|^2$ has a peak in the transition state region, implying that more data points should be introduced around the peak. Second, we may lower the energy barrier to facilitate transitions between the metastable states, which can be done by adding a biased potential V_{bias} to the original potential V [6,17].

3.2. *Sample generation*

Let $p_{V,q}$ be a probability density function (PDF) that is dependent on V and q_θ. Here, we present two choices for constructing $p_{V,q}$. One choice is to set

$$p_{V,q}(\boldsymbol{x}) = \frac{|\nabla q_\theta(\boldsymbol{x})|^2 e^{-\beta V(\boldsymbol{x})}}{C_1}, \tag{6}$$

where C_1 is the normalization constant. That is, we treat the nonnegative integrand in (3) as an unnormalized probability density function. If there exists a high energy barrier, we can use a biased potential V_{bias} to lower the energy barrier, which yields the following sampling distribution

$$p_{V,q}(\boldsymbol{x}) = \frac{|\nabla_x q_\theta(\boldsymbol{x})|^2 e^{-\beta(V(\boldsymbol{x})+V_{\text{bias}}(\boldsymbol{x}))}}{C_2}, \tag{7}$$

where C_2 is the corresponding normalization constant. The biased potential can be chosen to be an umbrella potential [41] or a potential derived from the metadynamics [42,43].

Now the question is how we can generate samples from the above sampling distribution. Here, we use KRnet, which is a type of flow-based generative models [44,45], for PDF approximation and sample generation. We note that other deep generative models with exact likelihood computation [46,47] can also be used here. Following Tang et al. [26,28,48], we use KRnet for sample generation since it can be regarded as a generalization of real NVP [48], while it does not require numerically solving ordinary differential equations during sampling, compared with continuous normalizing flows. Let $p_{\mathsf{KRnet}}(\boldsymbol{x};\Theta_f)$ be a PDF model induced by KRnet with parameters Θ_f [26,48–50]. The PDF model p_{KRnet} is induced by a bijection f_{KRnet} with parameters Θ_f:

$$p_{\mathsf{KRnet}}(\boldsymbol{x};\Theta_f) = p_Z(f_{\mathsf{KRnet}}(\boldsymbol{x}))\left|\det \nabla_x f_{\mathsf{KRnet}}\right|,$$

where p_Z is a prior PDF (e.g., the standard Gaussian distribution). The overall structure of KRnet is defined as follows

$$\boldsymbol{z} = f_{\mathsf{KRnet}}(\boldsymbol{x}) = L_N \circ f^{\mathsf{outer}}_{[K-1]} \circ \cdots \circ f^{\mathsf{outer}}_{[1]}(\boldsymbol{x}),$$

where $f^{\mathsf{outer}}_{[i]}$ is defined as

$$f^{\mathsf{outer}}_{[k]} = L_S \circ f^{\mathsf{inner}}_{[k,L]} \circ \cdots \circ f^{\mathsf{inner}}_{[k,1]} \circ L_R.$$

More specifically, $f^{\mathsf{inner}}_{[k,i]}$ is a combination of L affine coupling layers [44,45] and one scale and bias layer, and L_N, L_S and L_R denote the nonlinear layer, the squeezing layer and the rotation layer respectively, where details can be found in the literature [26,48,49,51]. We can approximate $p_{V,q}$ through solving the optimization problem

$$\Theta_f^* = \arg\min_{\Theta_f} D_{\mathsf{KL}}(p_{V,q}(\boldsymbol{x})\|p_{\mathsf{KRnet}}(\boldsymbol{x};\Theta_f)),$$

where $D_{\mathsf{KL}}(\cdot\|\cdot)$ indicates the Kullback-Leibler (KL) divergence between two distributions. Minimizing the KL divergence is equivalent to minimizing the cross entropy between $p_{V,q}$ and p_{KRnet} [52,53]:

$$H(p_{V,q}, p_{\mathsf{KRnet}}) = -\int_{\Omega\setminus(A\cup B)} p_{V,q}(\boldsymbol{x}) \log p_{\mathsf{KRnet}}(\boldsymbol{x};\Theta_f) d\boldsymbol{x}.$$

The normalization constants in (6) and (7) do not affect the optimization with respect to Θ_f. Since the samples from $p_{V,q}$ are not available, one can approximate the cross entropy using the importance sampling technique:

$$H(p_{V,q}, p_{\mathsf{KRnet}}) \approx -\frac{1}{N}\sum_{i=1}^{N} \frac{p_{V,q}(\boldsymbol{x}_i)}{p_{\text{IS}}(\boldsymbol{x}_i)} \log p_{\mathsf{KRnet}}(\boldsymbol{x}_i;\Theta_f), \tag{8}$$

where $p_{\text{IS}}(\boldsymbol{x}_i)$ is a known PDF model and $\{\boldsymbol{x}_i\}_{i=1}^N$ are the samples from $p_{\text{IS}}(\boldsymbol{x}_i)$. For example, the PDF model $p_{\text{IS}}(\boldsymbol{x}_i)$ can be chosen to be a PDF model induced by a known KRnet with parameters Θ_f', i.e.,

$$\boldsymbol{x}_i = f^{-1}_{\mathsf{KRnet}}(\boldsymbol{z}_i), \tag{9}$$

with $\boldsymbol{z}_i$ being sampled from the standard Gaussian distribution. We then minimize the discretized cross entropy (8) to obtain an approximation of Θ_f^*.

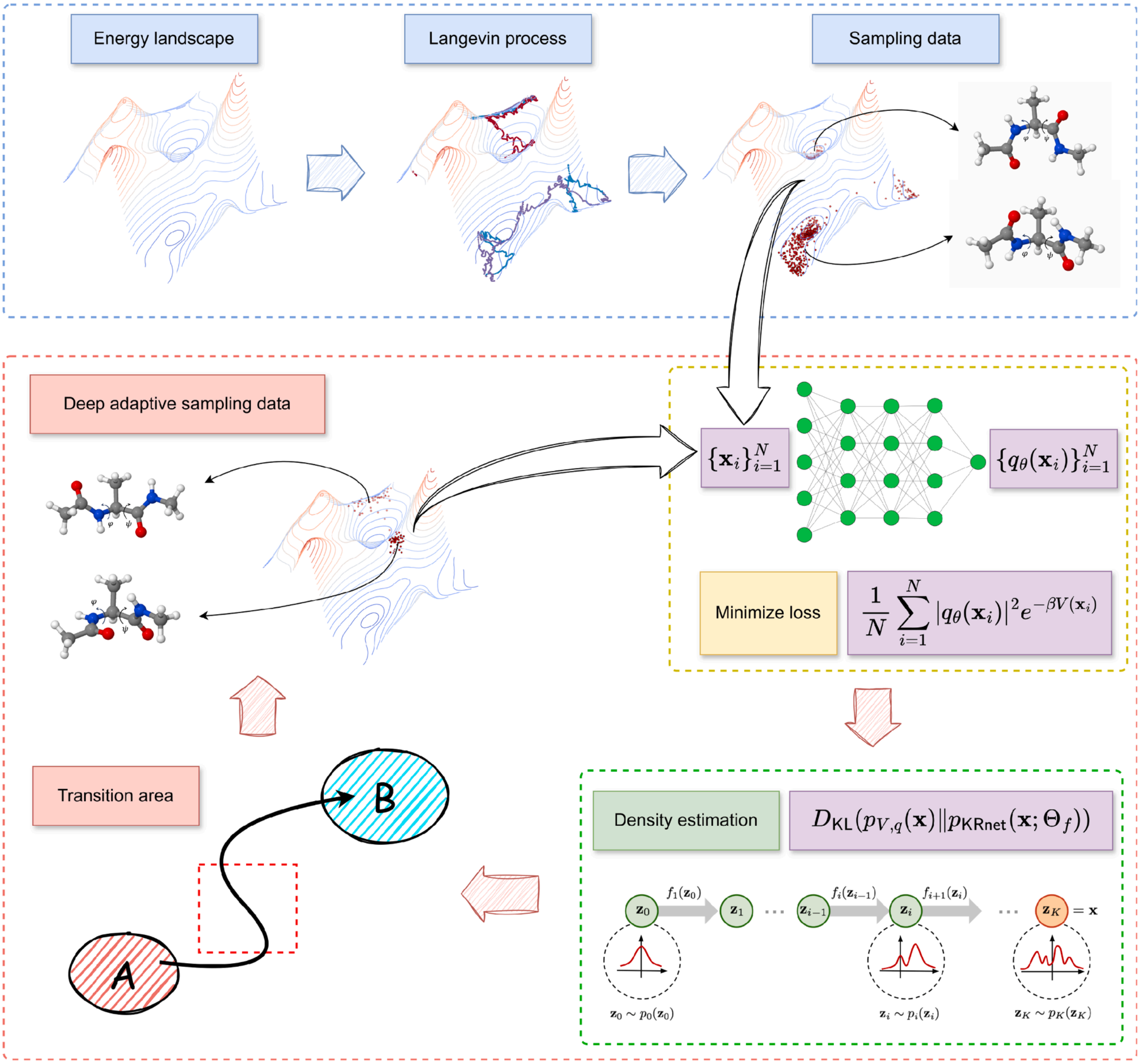

Energy landscape
Langevin process
Sampling data
Deep adaptive sampling data
$\{\mathbf{x}_i\}_{i=1}^N$
$\{q_\theta(\mathbf{x}_i)\}_{i=1}^N$
Minimize loss
$\frac{1}{N}\sum_{i=1}^{N}|q_\theta(\mathbf{x}_i)|^2 e^{-\beta V(\mathbf{x}_i)}$
Transition area
A
B
Density estimation
$D_{\mathrm{KL}}(p_{V,q}(\mathbf{x})\|p_{\mathrm{KRnet}}(\mathbf{x};\Theta_f))$
$f_1(\mathbf{z}_0)$
$f_i(\mathbf{z}_{i-1})$
$f_{i+1}(\mathbf{z}_i)$
$\mathbf{z}_0$
$\mathbf{z}_1$
$\mathbf{z}_{i-1}$
$\mathbf{z}_i$
$\mathbf{z}_K = \mathbf{x}$
$\mathbf{z}_0 \sim p_0(\mathbf{z}_0)$
$\mathbf{z}_i \sim p_i(\mathbf{z}_i)$
$\mathbf{z}_K \sim p_K(\mathbf{z}_K)$
φ
ψ

Fig. 1. The schematic of DASTR for computing the committor function. Training a deep neural network q_θ to approximate the high-dimensional committor function must use a high-quality dataset (i.e. data points from the transition area). Typically, the data points from Langevin dynamics are not in the transition state region since the transition between two metastable states is rare and difficult to sample. The proposed DASTR method can adaptively generate effective data points on the transition area according to the information of the current approximate solution. The key point is to define a sampling distribution $p_{V,q}$ dependent on the current approximate solution and the potential. Effective data points in the transition area are generated by sampling from $p_{V,q}$, which is achieved through training a deep generative model.

3.3. DASTR algorithm

Our adaptive sampling strategy is stated as follows. Let $\mathsf{S}_0 = \{\boldsymbol{x}_{0,i}\}_{i=1}^{N_0}$ be a set of collocation points that are sampled from a given distribution $p_0(\boldsymbol{x})$ in $\Omega\backslash(A \cup B)$. Using S_0, we minimize the empirical loss defined in (5) to obtain q_{θ_1}. With q_{θ_1}, we minimize the cross entropy in (8) to get $p_1 = p_{\mathsf{KRnet}}(\boldsymbol{x};\Theta_f^{*,(1)})$. A new set $\mathsf{S}_1^g = \{\boldsymbol{x}_{1,i}\}_{i=1}^{n_1}$ with $n_1 \le N_0$ is generated by $f_{\mathsf{KRnet}}^{-1}(\boldsymbol{z}_i;\Theta_f^{*,(1)})$ (see (9)) to refine the training set. To be more precise, we replace n_1 points in S_0 with S_1^g to get a new set S_1. Then we continue to update the approximate solution q_{θ_1} using S_1 as the training set. In general, at the k-stage, suppose that we have n_j samples $\mathsf{S}_j^g = \{\boldsymbol{x}_{j,i}\}_{i=1}^{n_j}$ from p_j for $j = 1, \dots, k$, where p_j is the PDF model at the j-th stage and it can be trained by letting $p_{j-1} = p_{\mathsf{KRnet}}(\boldsymbol{x}_i;\Theta'_f)$ in (8). The training

Algorithm 1 DASTR.

```
Input: Initial $q_{\theta_0}$, maximum stage number $N_{\text{adaptive}}$, maximum epoch number $N_e$, $N_e'$, batch size $m, m'$, initial training set $\mathsf{S}_0 = \{\boldsymbol{x}_{0,i}\}_{i=1}^{N_0}$.
for $k = 0 : N_{\text{adaptive}} - 1$ do
    for $i = 1 : N_e$ do
        for $l$ steps do
            Sample $m$ samples from $\mathsf{S}_k$.
            Update $q_\theta(\boldsymbol{x})$ by descending the stochastic gradient of the discrete variational loss (see (10)).
        end for
    end for
    for $i = 1 : N_e'$ do
        for $l$ steps do
            Sample $m'$ samples from the standard Gaussian distribution.
            Generate samples using (9).
            Update $p_{\text{KRnet}}(\boldsymbol{x}; \Theta_f)$ by descending the stochastic gradient of $H(p_{V,q}, p_{\text{KRnet}})$ (see (8)).
        end for
    end for
    Refine the training set: use $p_{k+1}$ to get $\mathsf{S}_{k+1}$.
end for
Output: $q_\theta$
```

set S_k at the k-th stage consists of $\boldsymbol{x}_{j,i}$. We use S_k to obtain $q_{\theta_{k+1}}$ as

$$\theta_{k+1} = \arg\min_{\theta} \sum_{j=0}^{k} \frac{1}{n_j} \sum_{i=1}^{n_j} \alpha_j |\nabla q_\theta(\boldsymbol{x}_{j,i})|^2 \frac{e^{-\beta V(\boldsymbol{x}_{j,i})}}{p_j(\boldsymbol{x}_{j,i})}, \tag{10}$$

where q_θ is initialized as q_{θ_k}, $\alpha_j = n_j / \sum_{j=0}^{k} n_j$ is a weight to balance the different distributions p_j, and n_0 is the number of points kept in S_0 at the k-th stage. Starting with $p_k = p_{\text{KRnet}}(\boldsymbol{x}; \Theta_f^{*,(k)})$, the density model $p_{\text{KRnet}}(\boldsymbol{x}; \Theta_f)$ is updated by (8) to get p_{k+1}. A new set $\mathsf{S}_{k+1}^g = \{\boldsymbol{x}_{k+1,i}\}_{i=1}^{n_{k+1}}$ of collocation points is generated by (9). We then use S_{k+1}^g to refine the training set to get S_{k+1}. We repeat the above procedure to obtain Algorithm 1 for the deep adaptive sampling on transition paths. We call this method DASTR for short. The main idea of our algorithm is also illustrated in Fig. 1.

3.4. DASTR in the latent space

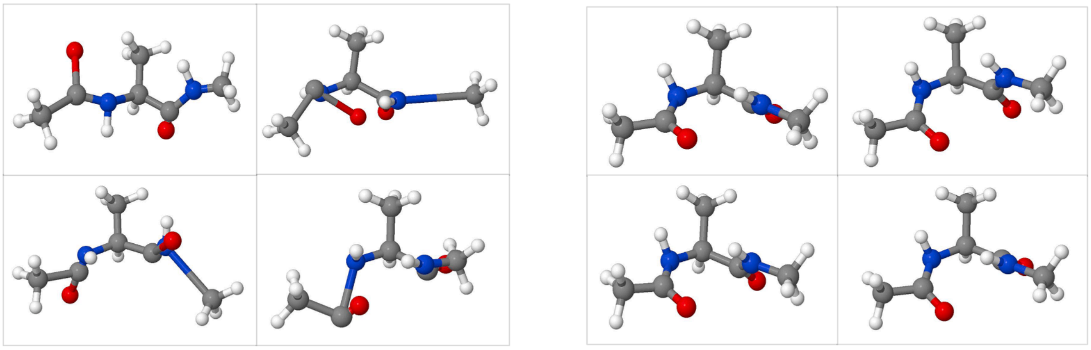
For complex systems, such as protein molecules, directly applying DASTR will result in the generation of physically unreasonable molecular configurations during the adaptive sampling procedure. The reason behind this is the strong correlation among the atomic coordinates required by physically reasonable protein structures. As a result, directly using the atomic coordinates as input to the KRnet may fail to capture the interatomic relationships effectively. This observation is demonstrated in Fig. 2. The molecular configurations in the left plot, which are almost physically unreasonable, are sampled from a trained KRnet in the original high-dimensional space, while the molecular configurations in the right plot, which are physically consistent, are sampled using latent collective variables as discussed later in Section 3.4.2.

To resolve this issue, we resort to sampling in the latent space, where we consider two strategies: one is based on the collective variables (CVs) method [54–56] (see Section 3.4.1), and the other is based on autoencoders (see Section 3.4.2). CVs refer to variables that can capture critical information about molecular structures. For example, the dihedral angles of the backbone atoms or distance between atoms can be selected as the CVs in protein systems. CVs can help reduce the computational complexity and enhance the sampling correctness. Moreover, we propose using an autoencoder to automatically select latent CVs that help generate physically reasonable molecular configurations, even though these latent CVs typically lack explicit physical meanings.

The basic idea of the collective variables method is to replace the original coordinates with some collective variables $\boldsymbol{s}(\boldsymbol{x}) = [s_1(\boldsymbol{x}), \dots, s_m(\boldsymbol{x})]^\top$ with $m \ll d$, where d is the dimensionality of $\boldsymbol{x}$. Then we can restrict our attention to the collective variables during the adaptive sampling procedure:

$$p_{V,q}(\boldsymbol{s}(\boldsymbol{x})) = p_{V,q}(\boldsymbol{x}), \tag{11}$$

where the $p_{V,q}(\boldsymbol{x})$ corresponds to the term defined in Eqs. (6) and (7). Since the collective variables can capture the essential structural features of molecules, one can take adaptively sampling step on the collective variables $\boldsymbol{s}(\boldsymbol{x})$ as illustrated in Algorithm 1. To generate samples in the latent space, we need to train KRnet using the CVs as input to learn the probability distribution in the latent space. Similar to the discussions in Section 3.2, training KRnet can be performed by minimizing the cross entropy loss defined in the latent space. This way, the deep generative model is used to generate samples of the collective variables instead of the coordinates $\boldsymbol{x}$. After generating the collective variables, one can do some post-processing steps to obtain new samples of $\boldsymbol{x}$. This will reduce the probability of generating nonphysical samples. If there is no prior information for selecting the proper collective variables, we use an autoencoder to learn some latent variables from the data and use them as the collective variables. The overall procedure along this line is summarized in Algorithm 2.



(a) Molecules generated using coordinates of heavy atoms as the input to KRnet.

(b) Molecules generated using latent CVs as the input to KRnet.

Fig. 2. Molecular configurations of alanine dipeptide generated by two different settings in DASTR: (a) the inputs of KRnet are the coordinates of heavy atoms (b) the inputs of KRnet are the latent CVs. The hydrogen atoms are completed by the software package PyMOL [57]. This figure demonstrates that using the latent collective variables to conduct DASTR is more effective.

Algorithm 2 DASTR in the latent space.

```
Input: Initial $q_{\theta_0}$, maximum stage number $N_{\text{adaptive}}$, maximum epoch number $N_e$, $N_e'$, batch size $m, m'$, initial training set $\mathsf{S}_0 = \{\boldsymbol{x}_{0,i}\}_{i=1}^{N_0}$.
if Using autoencoder then
    Train the autoencoder using $\mathsf{S}_0$.
end if
for $k = 0 : N_{\text{adaptive}} - 1$ do
    for $i = 1 : N_e$ do
        for $l$ steps do
            Sample $m$ samples from $\mathsf{S}_k$.
            Update $q_\theta(\boldsymbol{x})$ by descending the stochastic gradient of the discrete variational loss (see (10)).
        end for
    end for
    for $i = 1 : N_e'$ do
        for $l$ steps do
            Sample $m'$ samples from the standard Gaussian distribution.
            if Using autoencoder then
                Update $p_{\mathsf{KRnet}}(\boldsymbol{s}(\boldsymbol{x}); \Theta_f)$ by descending the stochastic gradient of $H(p_{V,q}, p_{\mathsf{KRnet}})$ using (14).
            else
                Update $p_{\mathsf{KRnet}}(\boldsymbol{s}(\boldsymbol{x}); \Theta_f)$ by descending the stochastic gradient of $H(p_{V,q}, p_{\mathsf{KRnet}})$ using (12))
            end if
        end for
    end for
    Generate new samples of the latent collective variables by the trained KRnet.
    Use the pretrained decoder to get new samples of $\boldsymbol{x}$.
    Refine the training set to get $\mathsf{S}_{k+1}$.
end for
Output: $q_\theta$
```

3.4.1. *Hand-picking CVs with umbrella sampling*

We first consider that the explicit collective variables are available. For alanine dipeptide studied in this work, the dihedral angles of the backbone atoms are selected as CVs [6]. As discussed above, we need to ensure that the samples obey the molecular configurations during the adaptive sampling procedure.

It is straightforward to train a KRnet to model the distribution in terms of collective variables $\boldsymbol{s}$. The KRnet that maps the collective variables $\boldsymbol{s}$ to a standard Gaussian is obtained by minimizing the following cross entropy

$$H(p_{V,q}, p_{\mathsf{KRnet}}) \approx -\frac{1}{N}\sum_{i=1}^{N} \frac{p_{V,q}(\boldsymbol{s}(\boldsymbol{x}_i))}{p_{\mathrm{IS}}(\boldsymbol{s}(\boldsymbol{x}_i))} \log p_{\mathsf{KRnet}}(\boldsymbol{s}(\boldsymbol{x}_i); \Theta_f), \tag{12}$$

where $p_{\mathrm{IS}}(\boldsymbol{s}(\boldsymbol{x}_i)) = e^{-\beta V_{\mathrm{modified}}(\boldsymbol{x}_i)}$ and each $\boldsymbol{x}_i$ is a sample drawn from the previous step. The generation of new samples for $\boldsymbol{x}$ is achieved in two steps: we first generate samples for the collective variables $\boldsymbol{s}$ using the trained KRnet, and then sample $\boldsymbol{x}$ that satisfies $\boldsymbol{s}(\boldsymbol{x}) \approx \boldsymbol{s}$ using umbrella sampling [41] (see Appendix B.4 for more details).

The potential function V_{modified} is used to simulate the SDE to generate new samples

$$V_{\mathrm{modified}}(\boldsymbol{x}) = V(\boldsymbol{x}) + V_{\mathrm{US}}(\boldsymbol{x}),$$

where V is the original potential in (1) and $V_{\mathrm{US}}(\boldsymbol{x})$ is the umbrella potential with the following form

$$V_{\mathrm{US}}(\boldsymbol{x}) = \frac{1}{2}\sum_{i=1}^{m} k_{\mathrm{us}}(s_i(\boldsymbol{x}) - s_i(\boldsymbol{x}_0))^2. \tag{13}$$

Here, $s_i(\boldsymbol{x}_0)$ is the target CVs generated by the trained KRnet, $s_i(\boldsymbol{x})$ represents the CVs with respect to $\boldsymbol{x}$, m is the number of CVs, and k_{us} is the force constant. We perform a rapid iterative process of umbrella sampling to transfer the CVs to the target region, and finally sample near the target CVs in the modified potential. This ensures the physical validity of the molecular configurations during the adaptive sampling procedure. However, selecting proper collective variables requires additional domain-specific knowledge, which is not a trivial task. Additionally, this strategy for implementing adaptive sampling in the latent space still requires simulating the SDE, which limits its sampling efficiency.

3.4.2. *Latent CVs with autoencoder*

In this section, we propose an alternative method that employs an autoencoder to automatically select the latent variables as the collective variables. The autoencoder can be trained before the first stage in Algorithm 2 using the data from metadynamics. After training, the autoencoder is fixed during the adaptive sampling procedure.

The configurations of molecular systems are primarily determined by the positions of the heavy atoms and the positions of the hydrogen atoms can be inferred from the positions of the heavy atoms. Based on this observation, we selected the coordinates of all the heavy atoms of molecules from S_0 as the dataset for training the autoencoder. The autoencoder consists of two parts: an $\mathsf{encoder} = \boldsymbol{s}(\boldsymbol{x})$ and a $\mathsf{decoder} = \boldsymbol{S}(\boldsymbol{s}(\boldsymbol{x}))$. Both the encoder and decoder are modeled by neural networks. Training the autoencoder aims to minimize the mean squared error

$$\frac{1}{N}\sum_{i=1}^{N}(\boldsymbol{S}(\boldsymbol{s}(\boldsymbol{x}_i)) - \boldsymbol{x}_i)^2.$$

Once the autoencoder is trained, the latent CVs can be obtained by the encoder. To this end, we utilize KRnet to learn the distribution of the latent CVs by minimizing the following cross entropy with respect to the latent CVs

$$H(p_{V,q}, p_{\mathsf{KRnet}}) \approx -\frac{1}{N}\sum_{i=1}^{N}\frac{p_{V,q}(\boldsymbol{s}(\boldsymbol{x}_i))}{p_{\mathsf{KRnet}}(\boldsymbol{s}(\boldsymbol{x}_i);\Theta'_f)}\log p_{\mathsf{KRnet}}(\boldsymbol{s}(\boldsymbol{x}_i);\Theta_f), \tag{14}$$

where the parameters Θ'_f can be chosen from the previous step.

Once we have the trained KRnet in hand, we can generate samples $\boldsymbol{s}$ in the latent space. These samples are subsequently decoded using the pretrained decoder to reconstruct the positions of all the heavy atoms. The hydrogen atoms are automatically completed using the software package PyMOL [57]. Finally, we calculate the potential energy of the generated molecular configurations to exclude those samples with excessively high potential energies, thus avoiding the generation of physically unreasonable configurations. The generated molecular configurations are illustrated in Fig. 2b. The proportion of reasonable configurations generated by this method exceeds 97 % (details can be found in Section 4.3.2). The computation process is illustrated in Fig. 3.

Remark 1. The key point here is that the autoencoder helps us automatically obtain the latent collective variables that reflect the molecular configuration, which serve as the input of KRnet, without the need of hand-picking physical CVs. In Section 4.3.1, we use KRnet to learn the distribution corresponding to the physical CVs and employ umbrella sampling to generate samples of molecules based on these physical CVs. However, this process consumes significant time and computational resources because umbrella sampling is still based on the SDE simulation. In contrast, the autoencoder explores latent CVs, allowing us to break free from the reliance on physical CVs and the associated SDE-based sampling methods. Moreover, the decoder can quickly reconstruct the molecular structure, significantly improving the computational efficiency. We compare the sampling time of the two methods in Section 4.3.2.

4. Numerical study

We conduct three numerical experiments to demonstrate the effectiveness of the proposed method. The first one is a 10-dimensional rugged Mueller potential problem, the second one is a 20-dimensional standard Brownian motion problem, and the last one is the alanine dipeptide problem with the dimension $d = 66$. The performance of DASTR with the collective variables method and the autoencoder method is investigated using the alanine dipeptide problem. The detailed settings of numerical experiments are provided in Appendix B.

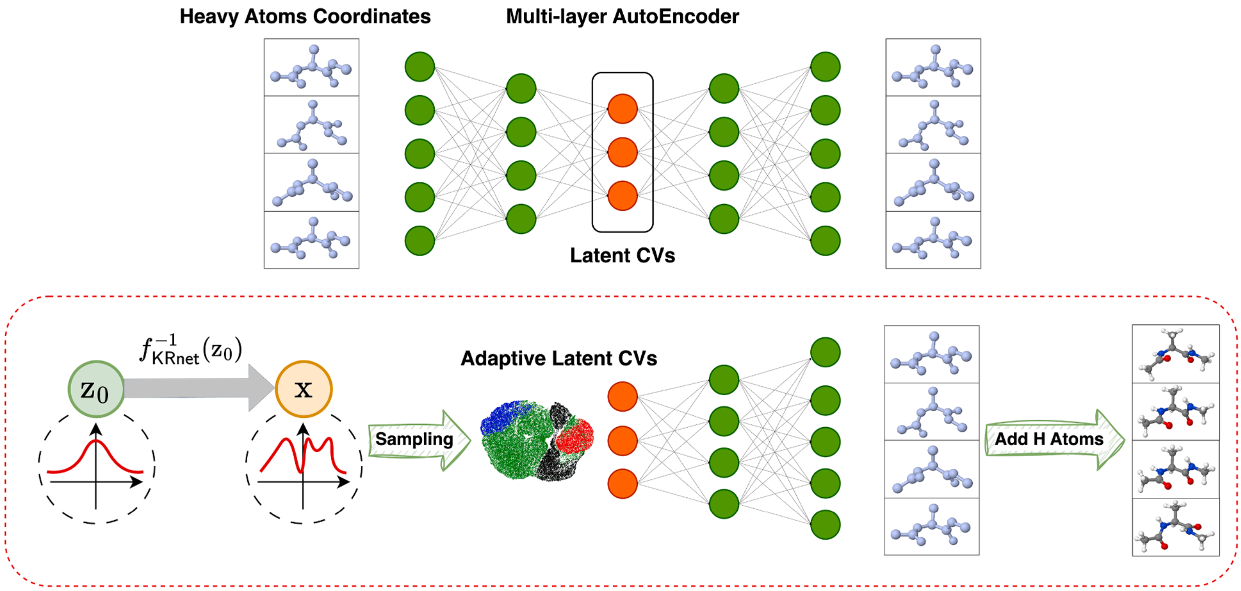


Fig. 3. The schematic of adaptive sampling in the latent space. We first train an autoencoder to obtain the latent variables as the collective variables (CVs), and then use KRnet to approximate the distribution of the CVs. After training KRnet, we use a random sample z_0 from the standard Gaussian distribution to generate a new sample of latent CVs. We can feed this new sample of latent CVs into the decoder to obtain a new sample of molecules after the post-processing step. Such a new sample of molecules is located in the transition state region. The autoencoder not only provides an effective way to automatically choose the collective variables, but also enhances the sampling efficiency of molecules in the transition state region.

4.1. Rugged mueller potential

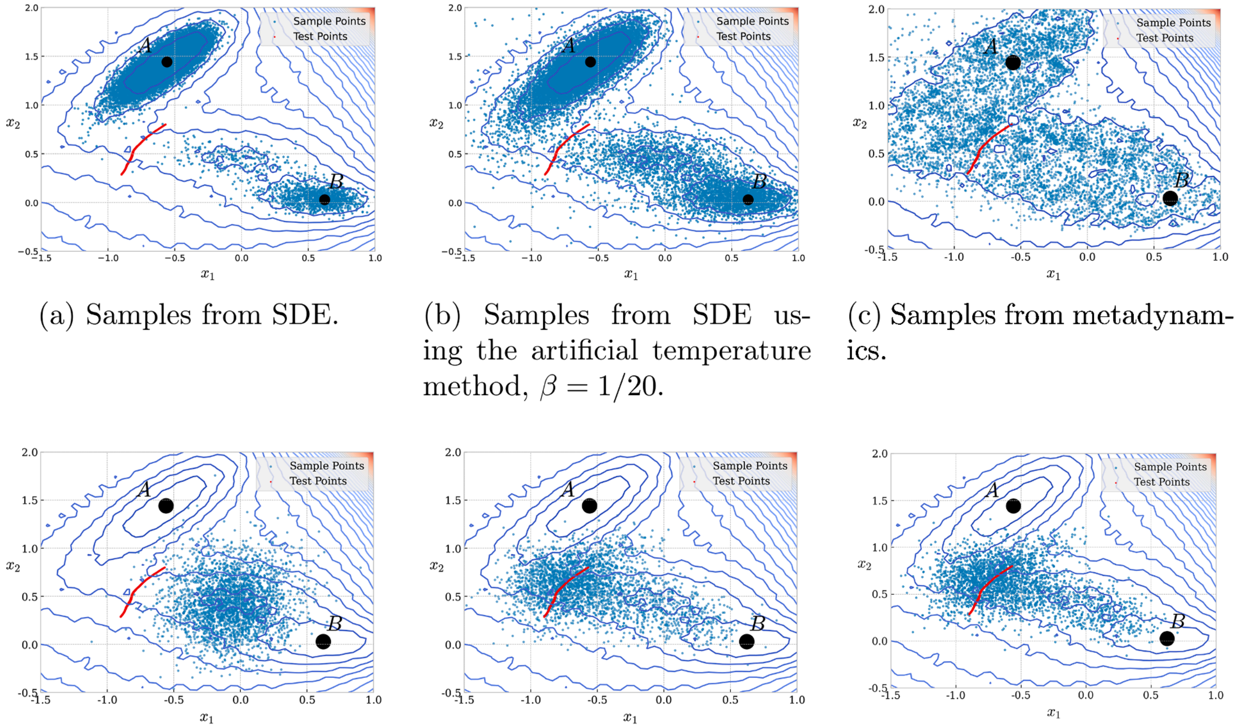
We consider the extended rugged Mueller potential embedded in the 10-dimensional space, which is a well-known test problem in computational chemical physics [6,13]. The extended rugged Mueller potential is given by $V(\boldsymbol{x}) = V_{\mathrm{rm}}(x_1, x_2) + 1/(2\sigma^2)\sum_{i=3}^{10} x_i^2$, where $\boldsymbol{x} \in \mathbb{R}^{10}$ and $V_{\mathrm{rm}}(x_1, x_2)$ is the rugged Mueller potential defined in $[-1.5, 1] \times [-0.5, 2]$

$$V_{\mathrm{rm}}(x_1, x_2) = \sum_{i=1}^{4} D_i e^{a_i(x_1-\xi_i)^2 + b_i(x_1-\xi_i)(x_2-\eta_i) + c_i(x_2-\eta_i)^2} + \gamma \sin(2k\pi x_1)\sin(2k\pi x_2).$$

We set $\sigma = 0.05$ as in Li et al. [6], and the other parameters are set to be the same as in Lai and Lu [5]. The inverse temperature is set to $\beta = 1/10$. In this test problem, the two metastable sets A and B are two cylinders with centers $[x_1, x_2] = [-0.558, 1.441]$ and $[x_1, x_2] = [0.623, 0.028]$ respectively with radius 0.1. In this setting, the solution of this 10-dimensional problem is the same as that of the two-dimensional rugged Mueller potential, i.e., $q(\boldsymbol{x}) = q_{\mathrm{rm}}(\boldsymbol{x})$ [6,13]. So, we can use the finite element method implemented in FEniCS [58,59] to obtain a reference solution to evaluate the performance. For comparison, we also implement the artificial temperature method and metadynamics [6] as the baseline model. Here we define the L^2 relative error $\|\boldsymbol{q}_\theta - \boldsymbol{q}_{\mathrm{ref}}\|_2 / \|\boldsymbol{q}_{\mathrm{ref}}\|_2$, where $\boldsymbol{q}_\theta$ and $\boldsymbol{q}$ denote two vectors whose elements are the function values of q_θ and q_{ref} at some grids respectively. We compute the relative error on some given points. For the first two-dimensional variables x_1 and x_2, we use the meshgrid generated in FEniCS to compute the relative error. We simulate the dynamics to get some samples for the rest of the variables $(x_3, \dots, x_{10})$. Finally, we concatenate these two parts to obtain the test dataset for computing the relative error. The settings of neural networks and training details can be found in Appendix B.1.

Fig. 4 shows the samples from different sampling strategies, where these samples are projected onto the $x_1 - x_2$ plane. Specifically, Fig. 4a shows the samples generated by SDE defined in (1). It can be seen that the samples from SDE are located around the two metastable states A and B, which are ineffective for approximating the committor function. Fig. 4b shows the samples from SDE with the artificial temperature method. While more samples show up in the transition state region compared with Fig. 4a, there is still insufficient data to accurately capture the committor function. Fig. 4c shows the samples from metadynamics. We choose the coordinates x_1 and x_2 (i.e., $S_1(x) = x_1$, $S_2(x) = x_2$ in (B.1)) by adding 2000 Gaussian functions with height $w = 5$ and width $\sigma_1 = \sigma_2 = 0.05$ into the potential, one for every 500 time steps. Then a set of data are sampled by simulating the Langevin dynamics using the modified potential with the time step $\Delta t = 10^{-5}$. As shown in Fig. 4d–f, our method is able to provide effective samples in the transition area. The evolution of the training set with respect to adaptivity iterations $k = 2, 15, 30$ is presented, where we randomly select 5000 samples in the training set for visualization. Compared to other approaches, many more samples are distributed in the transition state region $(\Omega \backslash (A \cup B))$, which is desired for approximating the committor function.

In Fig. 5a–d, we compare the reference solution q_{ref} obtained by the finite element method, the DASTR solution given by 4×10^5 samples and the approximate solution given by 4×10^5 samples from metadynamics and the artificial temperature method. Fig. 6a



(a) Samples from SDE.

(b) Samples from SDE using the artificial temperature method, $\beta = 1/20$.

(c) Samples from metadynamics.

(d) DASTR, samples at the 2-th stage.

(e) DASTR, samples at the 15-th stage.

(f) DASTR, samples at the 30-th stage.

Fig. 4. DASTR, samples for the 10-dimensional rugged Mueller potential problem. The red line denotes the test points from the 1/2-isosurface ($q \approx 1/2$) projected onto the $x_1 - x_2$ plane. (For interpretation of the references to colour in this figure legend, the reader is referred to the web version of this article.)

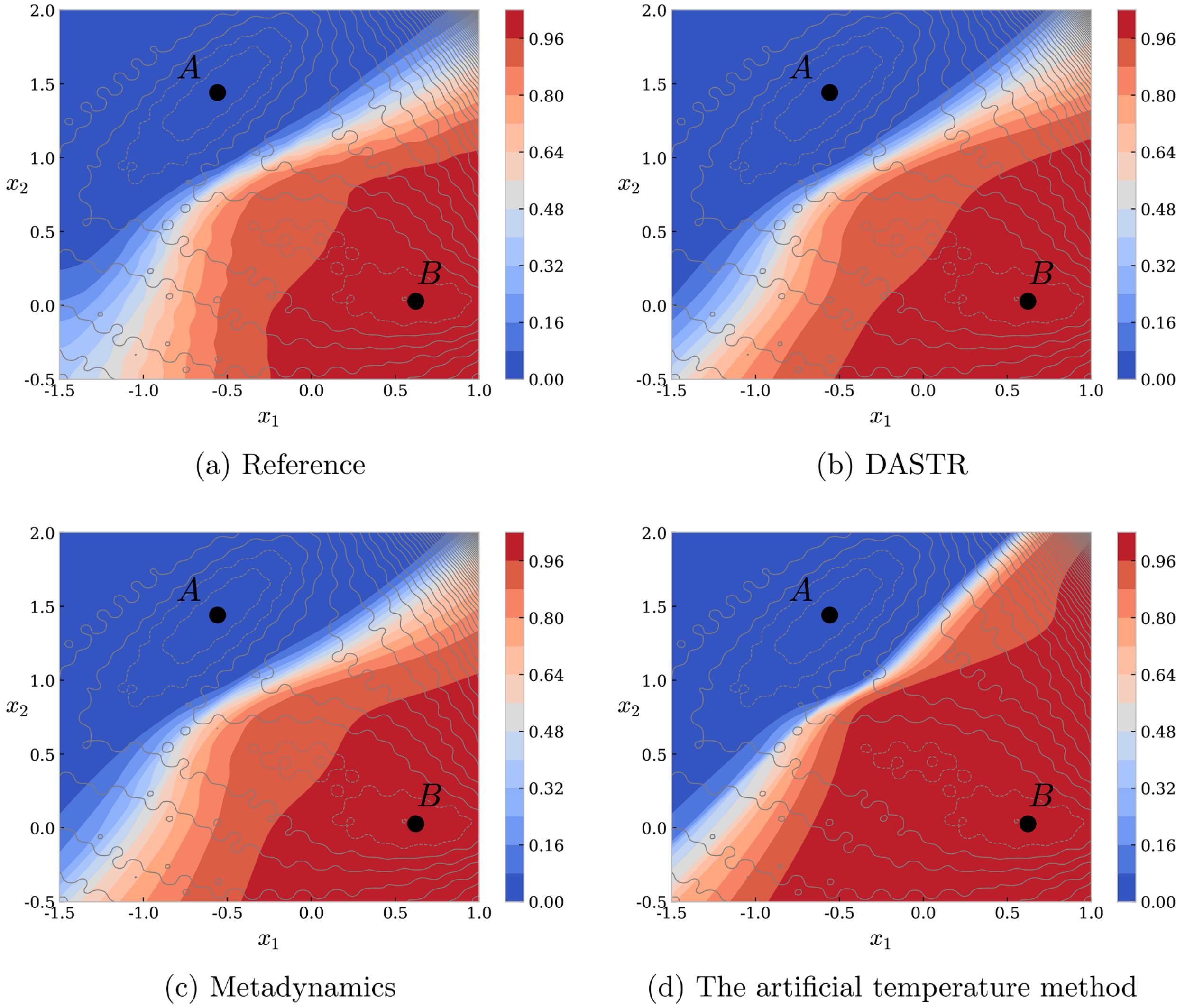
shows the error behavior of different methods. Fig. 6b shows the relative errors with respect to different sample sizes. From Fig. 6b, it is seen that the DASTR method is much more accurate than the method of sampling from dynamics. Due to the difficulty of sampling in the transition state region using SDE with the artificial temperature method, the solution obtained through the artificial temperature method fails to accurately capture the information of the committor function in the transition state region. To further investigate the performance of the proposed method, in Table 1, we show the L^2 relative errors of neural networks with varying numbers of neurons subject to different sample sizes. Here, we sample 12099 points near the 1/2-isosurface ($q(\boldsymbol{x}) \approx 0.5$) to compute the relative error. Meanwhile, we note that the boundary error is near zero (about 10^{-4}) since we choose two sufficient large sets S_A and S_B to enforce the boundary condition. The number of samples for training the neural network to approximate the boundary condition is the same as that of in $\Omega \backslash A \cup B$.

From Table 1, it is seen that our DASTR method is one order of magnitude more accurate than the artificial temperature method in all settings and has competitive performance compared with metadynamics for this rugged Mueller potential test problem.

4.2. Standard Brownian motion

In this test problem, we consider the committor function under the standard Brownian motion [60,61]. For a stochastic process $(\boldsymbol{X}_t)_{t\geq 0} \in \mathbb{R}^d$, which is a standard Brownian motion starting at $\boldsymbol{x} \in \mathbb{R}^d$, that is, $\boldsymbol{X}_t = \boldsymbol{x} + \boldsymbol{W}_t$, corresponding to $\nabla V(\boldsymbol{X}_t) = 0$ and $\beta = 1/2$ in (1). The two metastable sets A and B are defined as $A = \{\boldsymbol{x} \in \mathbb{R}^d : \|\boldsymbol{x}\|_2 < a\}$, $B = \{\boldsymbol{x} \in \mathbb{R}^d : \|\boldsymbol{x}\|_2 > b\}$ with $b > a > 0$. With these settings, for $d \geq 3$, there exists an analytical solution $q(\boldsymbol{x}) = (a^2 - \|\boldsymbol{x}\|_2^{2-d} a^2)/(a^2 - b^{2-d} a^2)$. In this test problem, we set $d = 20$ and $a = 1, b = 2$. The settings of neural networks and training details can be found in Appendix B.2. Since the solution to this test problem cannot be projected onto the low-dimensional space, we here compare different sampling methods by computing the L^2 relative error at a validation set with 5000 data points along a curve $\{(\kappa, \ldots, \kappa)^\top : \kappa \in [a/\sqrt{d}, b/\sqrt{d}]\}$ [61]. Meanwhile, we select 5000 points from the boundary $\{\boldsymbol{x} \in \mathbb{R}^d : \|\boldsymbol{x}\|_2 = a\}$ and $\{\boldsymbol{x} \in \mathbb{R}^d : \|\boldsymbol{x}\|_2 = b\}$ to compute the boundary errors.

Fig. 7 shows the results of the 20-dimensional standard Brownian motion test problem. Specifically, Fig. 7a shows the solutions obtained by different sampling methods, where it can be seen that the DASTR solution is more accurate than those of other sampling strategies. Fig. 7b shows the behavior of relative errors during training, where DASTR performs better than the uniform sampling strategy and SDE. Fig. 7c shows the relative errors for the uniform sampling method, SDE, and DASTR, where different numbers of



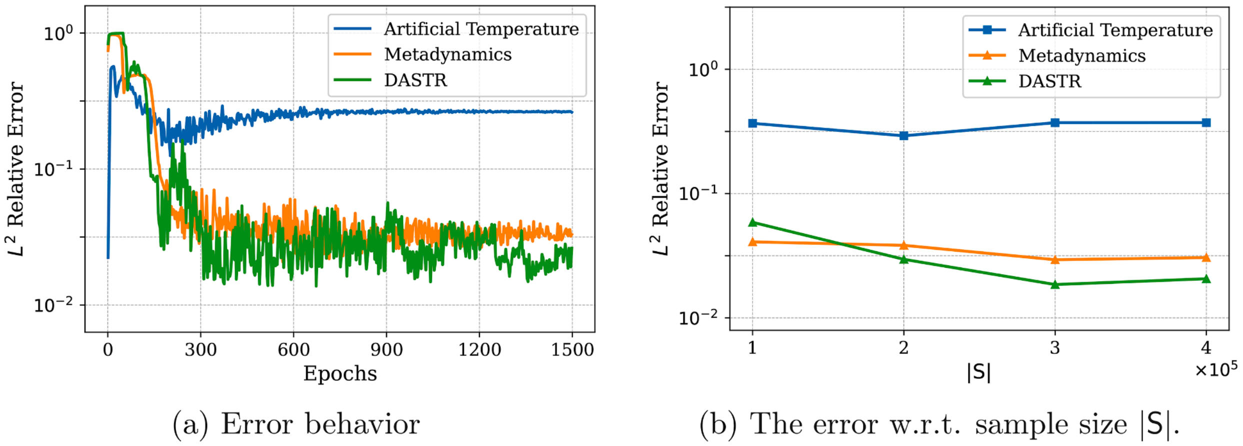
(a) Reference

(b) DASTR

(c) Metadynamics

(d) The artificial temperature method

Fig. 5. Solutions, 10-dimensional rugged Mueller potential test problem..



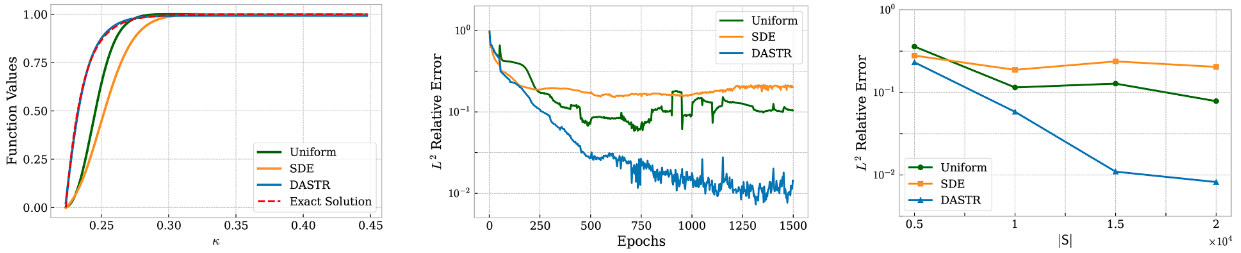
(a) Error behavior

(b) The error w.r.t. sample size |S|.

Fig. 6. Error behavior, 10-dimensional rugged Mueller potential test problem..

Table 1
10-dimensional rugged Mueller potential test problem: errors for different settings of neural networks and sampling strategies. We take 4 independent runs to compute the error statistics (relative error: mean ± standard deviation, log boundary error: mean).

		Number of Neurons in Hidden Layer			Log Boundary Error	
Sampling Method	|S|	20	50	100	A	B
SDE with the artificial temperature method	1×10^5	0.5446 ± 0.0724	0.4693 ± 0.0627	0.4023 ± 0.0819	−2.0917	−2.0276
	2×10^5	0.3183 ± 0.0592	0.2677 ± 0.0708	0.3063 ± 0.0477	−2.0940	−2.0654
	3×10^5	0.2717 ± 0.0487	0.2780 ± 0.0584	0.3955 ± 0.0311	−2.0784	−2.0238
	4×10^5	0.3822 ± 0.0555	0.3019 ± 0.0649	0.3822 ± 0.1213	−2.0890	−1.9449
Metaynamics	1×10^5	0.0535 ± 0.0022	0.0426 ± 0.0033	0.0409 ± 0.0028	−3.9793	−2.3869
	2×10^5	0.0413 ± 0.0025	0.0451 ± 0.0073	0.0384± 0.0048	−3.2065	−2.3682
	3×10^5	0.0419 ± 0.0023	0.0352 ± 0.0075	0.0294 ± 0.0033	−2.3967	−2.3791
	4×10^5	0.0440 ± 0.0042	0.0300 ± 0.0041	0.0306 ± 0.0021	−2.3983	−2.3771
DASTR (this work)	1×10^5	0.0620 ± 0.0070	0.0602 ± 0.0113	0.0615 ± 0.0071	−3.8727	−2.4399
	2×10^5	0.0498 ± 0.0102	0.0443 ± 0.0049	0.0310 ± 0.0024	−3.2961	−2.4276
	3×10^5	0.0386 ± 0.0089	0.0412 ± 0.0091	0.0172 ± 0.0028	−2.7152	−2.3933
	4×10^5	0.0371 ± 0.0056	0.0343 ± 0.0065	0.0206 ± 0.0052	−2.4379	−2.4139



(a) Solutions, $|\mathsf{S}| = 2\times10^4$.

(b) The error evolution, $|\mathsf{S}| = 2\times10^4$.

(c) The error w.r.t. sample size.

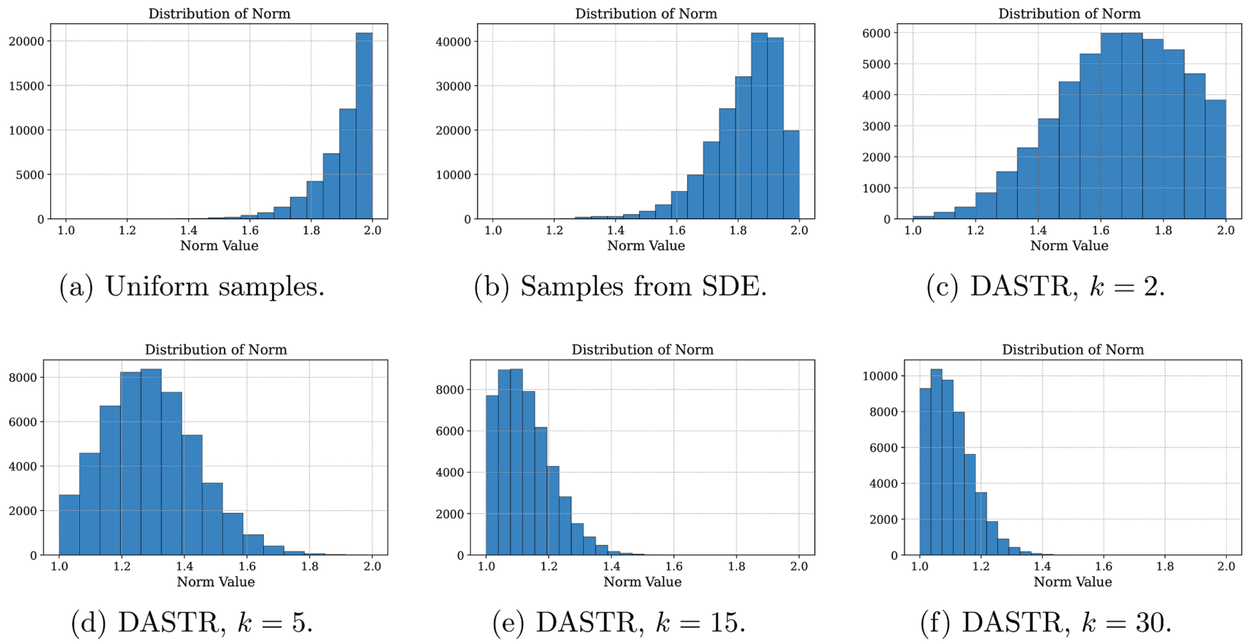
Fig. 7. Solutions evaluated along a curve and the behavior of relative errors, 20-dimensional standard Brownian motion test problem. The relative error is computed at the points along the curve $\{(\kappa,\ldots,\kappa)^\top : \kappa \in [a/\sqrt{d}, b/\sqrt{d}]\}$.

Table 2
20-dimensional standard Brownian motion test problem: errors for different settings of neural networks and sampling strategies. We take 4 independent runs to compute the statistics of the error (relative error: mean ± standard deviation, log boundary error: mean).

		Number of Neurons in Hidden Layer			Log Boundary Error	
Sampling Method	|S|	20	50	100	A	B
Uniform	5×10^3	0.1767 ± 0.0240	0.1906 ± 0.0214	0.4555 ± 0.0557	−0.3062	−0.3102
	1×10^4	0.1861 ± 0.0319	0.1760 ± 0.0492	0.1310 ± 0.0197	−1.9812	−2.8424
	1.5×10^4	0.2125 ± 0.0220	0.2003 ± 0.0295	0.1454 ± 0.0609	−2.0001	−3.0748
	2×10^4	0.1963 ± 0.0866	0.1611 ± 0.0227	0.1402 ± 0.0515	−2.4052	−3.4057
SDE	5×10^3	0.2127 ± 0.0802	0.2641 ± 0.0416	0.3696 ± 0.0633	−0.4601	−0.4928
	1×10^4	0.2846 ± 0.0523	0.2606 ± 0.0343	0.1586 ± 0.0179	−1.7785	−2.5162
	1.5×10^4	0.2861 ± 0.0177	0.1865 ± 0.0220	0.1706 ± 0.0434	−2.3749	−3.1361
	2×10^4	0.2321 ± 0.0278	0.1864 ± 0.0254	0.1342 ± 0.0434	−2.5961	−3.4535
DASTR (this work)	5×10^3	0.0996 ± 0.0374	0.1073 ± 0.0128	0.1266 ± 0.0277	−1.8125	−1.8270
	1×10^4	0.0835 ± 0.0215	0.0415 ± 0.0167	0.0410 ± 0.0106	−1.8741	−2.0758
	1.5×10^4	0.0824 ± 0.0412	0.0197 ± 0.0045	0.0141 ± 0.0053	−2.0812	−2.1624
	2×10^4	0.0227 ± 0.0051	0.0209 ± 0.0096	0.0114 ± 0.0021	−2.0991	−2.0811

samples are tested. From Fig. 7c, it is clear that, as the number of samples increases, the relative error of DASTR decreases more quickly than those of SDE and the uniform sampling strategy.

To see why DASTR works well, let us visualize the L^2-norm of samples from different sampling strategies. Fig. 8 shows the histogram of the norm of samples for different sampling strategies. From Fig. 8a and b, we can see that most of the samples fall into the interval where the norm of samples is near 2. This means that it is difficult to generate samples in the transition state region using the uniform sampling strategy or SDE. Indeed, in high-dimensional spaces, most of the volume of an object concentrates around its surface [62,63]. Hence, using uniform samples or samples generated by SDE is inefficient for estimating the committor function. Fig. 8c–f show the histogram of the norm of samples from DASTR. These histograms imply that the samples from DASTR capture the



(a) Uniform samples.

(b) Samples from SDE.

(c) DASTR, $k = 2$.

(d) DASTR, $k = 5$.

(e) DASTR, $k = 15$.

(f) DASTR, $k = 30$.

Fig. 8. Histogram of the norm of samples, 20-dimensional test problem.

Table 3
Time comparison of DASTR with the explicit collective variables and umbrella sampling and DASTR with the learned latent variables for different numbers of samples (the unit is seconds).

	Number of Samples				
Sampling Method	1×10^4	2×10^4	5×10^4	1×10^5	2×10^5
DASTR with umbrella sampling	234.01 s	476.19 s	1213.17 s	2406.86 s	4771.42 s
DASTR with learned latent variables	10.26 s	18.10 s	46.33 s	92.94 s	175.98 s

information of transitions, which improves the accuracy of estimating the committor function. In Table 2, we again present the L^2 relative errors of neural networks with varying numbers of neurons subject to different sample sizes and the boundary errors with the neurons of the neural network set to 100. Our DASTR method is one order of magnitude more accurate than the baseline methods in most settings and the boundary errors are close.

4.3. Alanine dipeptide

In this test problem, the isomerization process of the alanine dipeptide in vacuum at $T = 300K$ is studied, which is a widely used benchmark in the literature [6,17]. Two approaches are considered. In Section 4.3.1, we assume that the collective variables are known. Then, the proposed DASTR approach is applied to the collective variables, which will improve the robustness of DASTR in approximating the committor function. In Section 4.3.2, the collective variables are not explicitly given, which is a more realistic setting. We use an autoencoder to find some latent variables to serve as the collective variables.

The molecule we consider here consists of 22 atoms, each of which has three coordinates. This means that the dimension of the state variable is $d = 66$ in (2). There are two important dihedrals related to their configurations: ϕ (C-N-CA-C) and ψ (N-CA-C-N). The two metastable conformers of the molecule are C_{7eq} and C_{ax} located around $(\phi, \psi) = (-85^\circ, 75^\circ)$ and $(72^\circ, -75^\circ)$ respectively. More specifically, the two metastable sets A and B are defined as Li et al. [6]:

$$A = \left\{ \boldsymbol{x} : \left\| (\phi(\boldsymbol{x}), \psi(\boldsymbol{x})) - C_{7eq} \right\|_2 < 10^\circ \right\},$$
$$B = \left\{ \boldsymbol{x} : \left\| (\phi(\boldsymbol{x}), \psi(\boldsymbol{x})) - C_{ax} \right\|_2 < 10^\circ \right\}.$$

In Fig. 9, the molecule structures of two metastable states and two transition states are displayed.

The goal is to compute the committor function under the CHARMM force field [64–66]. Due to the high energy barrier between the two metastable states A and B, it is almost impossible for the molecule to cross this barrier from A to B. Consequently, sampling in the transition state region with SDE is extremely challenging.

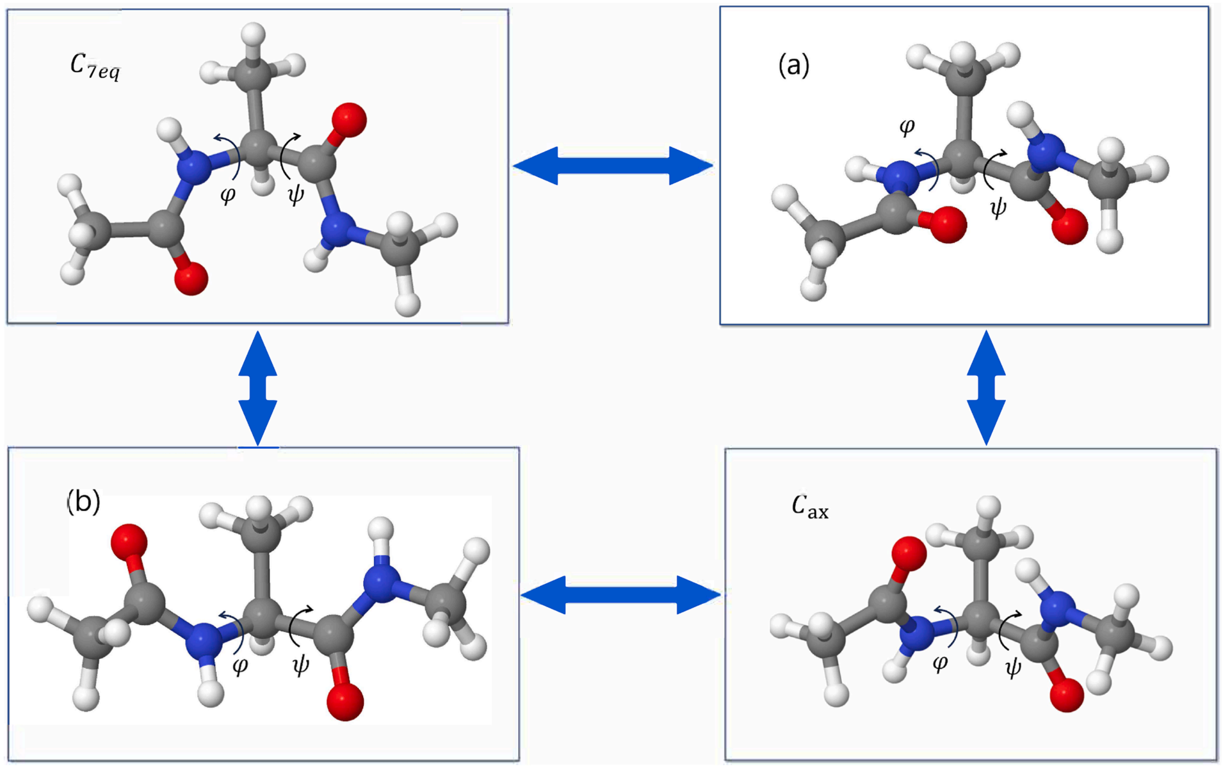
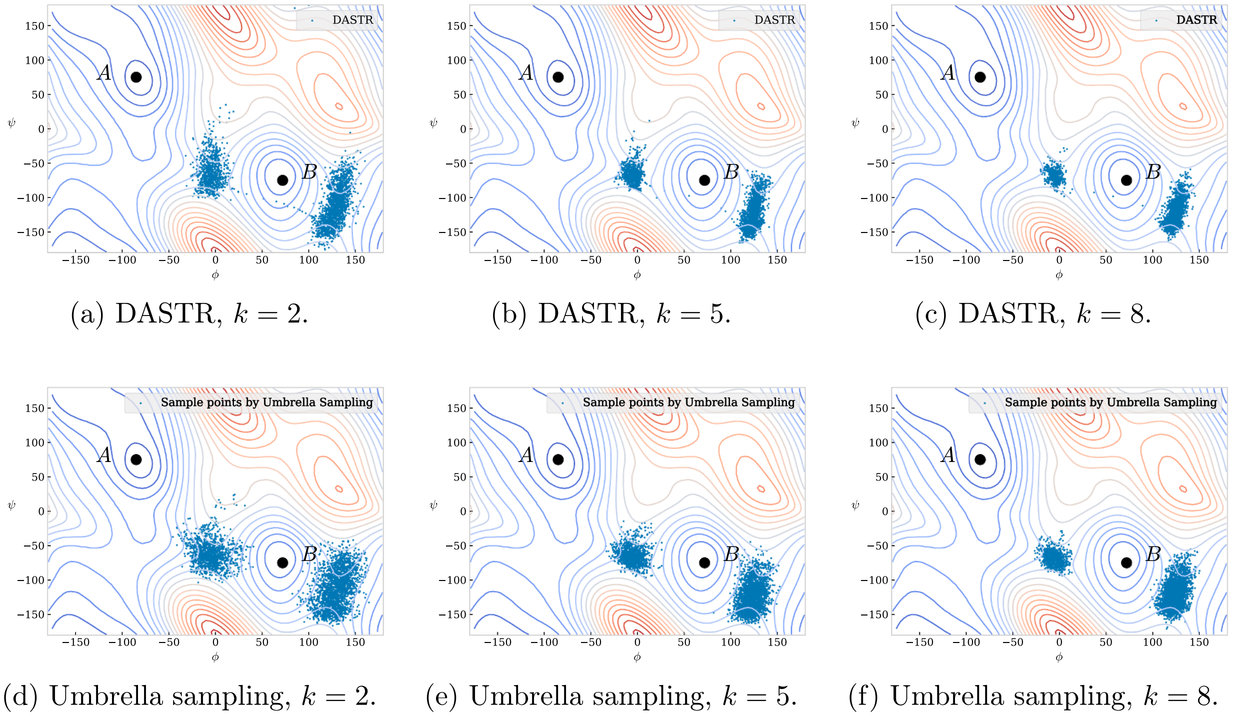


Fig. 9. The two metastable states and two transition states of the alanine dipeptide. C_{7eq} : $(\phi,\psi) \approx (-85°, 75°)$ and C_{ax} : $(\phi,\psi) \approx (72°, -75°)$ are two metastable states, (a) : $(\phi,\psi) \approx (0°, -65°)$ and (b) : $(\phi,\psi) \approx (130°, -125°)$ are two transition states.



(a) DASTR, $k = 2$.

(b) DASTR, $k = 5$.

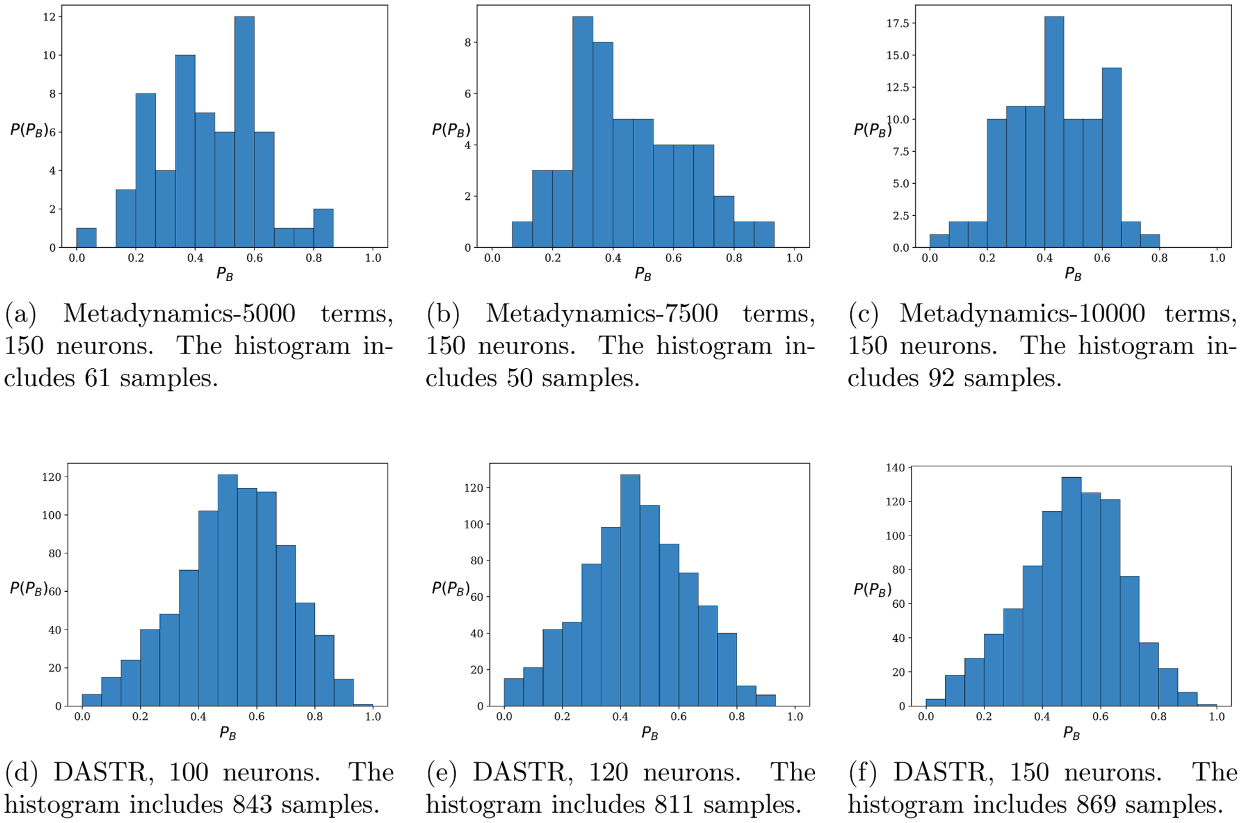
(c) DASTR, $k = 8$.

(d) Umbrella sampling, $k = 2$.

(e) Umbrella sampling, $k = 5$.

(f) Umbrella sampling, $k = 8$.

Fig. 10. Samples during training for the alanine dipeptide test problem. We use DASTR to generate target CVs in the transition state region; the umbrella sampling method is employed to generate samples around the target CVs to refine the training set. The figures are shown that the samples (scatter plot) distributed on the energy landscape with respect to ϕ and ψ.



(a) Metadynamics-5000 terms, 150 neurons. The histogram includes 61 samples.

(b) Metadynamics-7500 terms, 150 neurons. The histogram includes 50 samples.

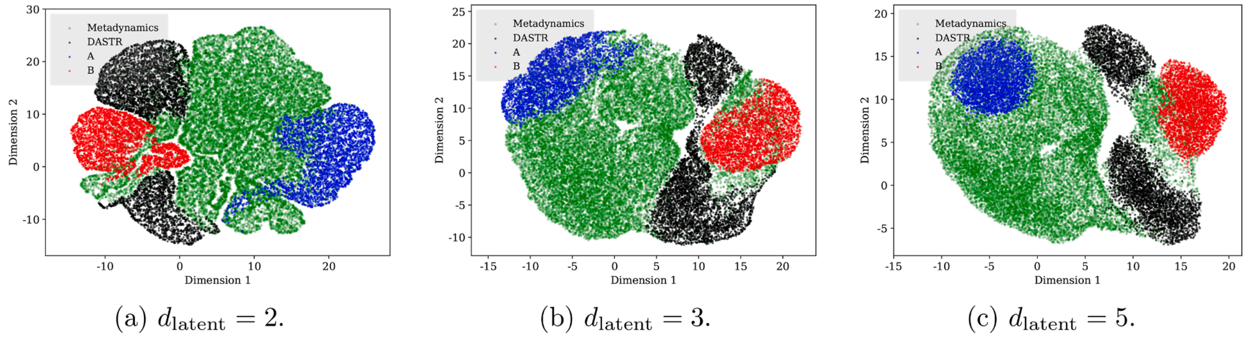
(c) Metadynamics-10000 terms, 150 neurons. The histogram includes 92 samples.

(d) DASTR, 100 neurons. The histogram includes 843 samples.

(e) DASTR, 120 neurons. The histogram includes 811 samples.

(f) DASTR, 150 neurons. The histogram includes 869 samples.

Fig. 11. The alanine dipeptide test problem: the histograms of the committor function values on the 1/2-th isosurface of q_θ with different numbers of neurons. q_θ is a five-layer fully connected neural network. The training details can be found in Appendix B.3.



(a) $d_{\text{latent}} = 2.$

(b) $d_{\text{latent}} = 3.$

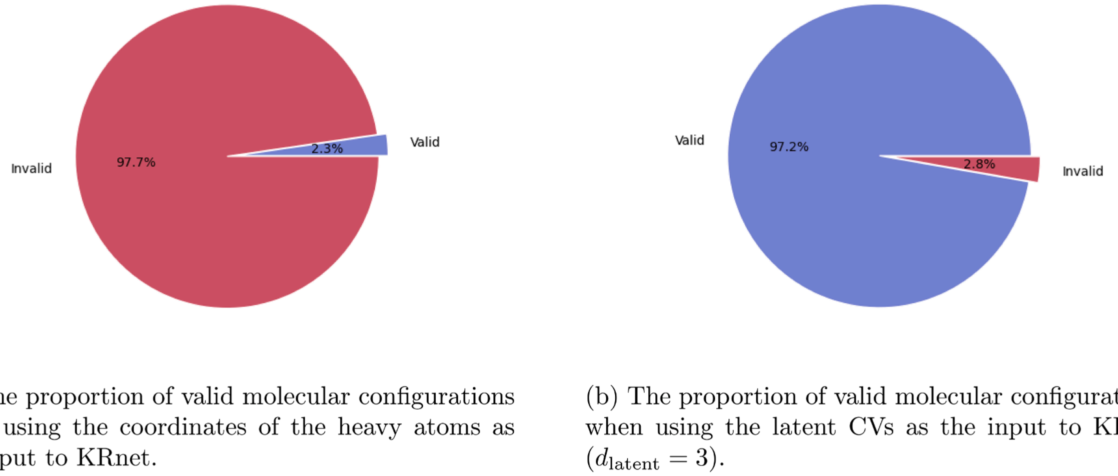
(c) $d_{\text{latent}} = 5.$

Fig. 12. Visualization of the latent collective variables, the two metastable states A and B, and samples from DASTR at the final stage in the latent space. The data points are projected onto a two-dimensional plane by UMAP [67] for visualization.

4.3.1. DASTR with explicit collective variables

In this section, we study the performance of DASTR with explicit collective variables. The collective variables is set to the two dihedrals ϕ (C-N-CA-C) and ψ (N-CA-C-N). For this realistic problem, we need to ensure that the samples from deep generative models conform to physically valid molecular configurations, making the problem far more challenging. To handle this difficulty, we combine our DASTR method with the umbrella sampling method [41] and the collective variables method. Simply speaking, we use the proposed DASTR method to generate the target collective variables in the umbrella potential. The details of the overall procedure can be found in Appendices B.3 and B.4.

For this problem, it is intractable to obtain the reference solution with grid-based numerical methods. To assess the performance of our method, we again consider those samples from the 1/2-isosurface. More specifically, we first use umbrella sampling (see Appendix B.4) to sample 1×10^7 points. After that, we use the trained model to compute q_θ at these sample points and filter to keep points on the set $\Gamma := \{\boldsymbol{x} : |q_\theta(\boldsymbol{x}) - 0.5|\} \leq 5 \times 10^{-5}$. We conduct 200 simulations of SDE for each point in Γ to obtain the corresponding



(a) The proportion of valid molecular configurations when using the coordinates of the heavy atoms as the input to KRnet.

(b) The proportion of valid molecular configurations when using the latent CVs as the input to KRnet ($d_{\text{latent}} = 3$).

Fig. 13. The proportions of valid molecular configurations for two different settings in DASTR are shown. This figure demonstrates the advantage of performing DASTR in the latent space.

Table 4
The Wasserstein distance between $\mathcal{N}\left(0.5, (4N_t)^{-1}\right)$ and the empirical distribution obtained by DASTR and metadynamics. We take 10 independent runs to compute the distance (mean ± standard deviation).

Method	Settings	Mean	Wasserstein Distance
	Gaussian Terms		
Metadynamics	5000	0.4411	0.1192 ± 0.0027
	7500	0.4432	0.1426 ± 0.0029
	10,000	0.4319	0.1131 ± 0.0025
	Latent CVs		
DASTR with Latent CVs	2	0.4894	0.0853 ± 0.0009
	3	0.4702	0.0866 ± 0.0006
	5	0.4738	0.1021 ± 0.0007

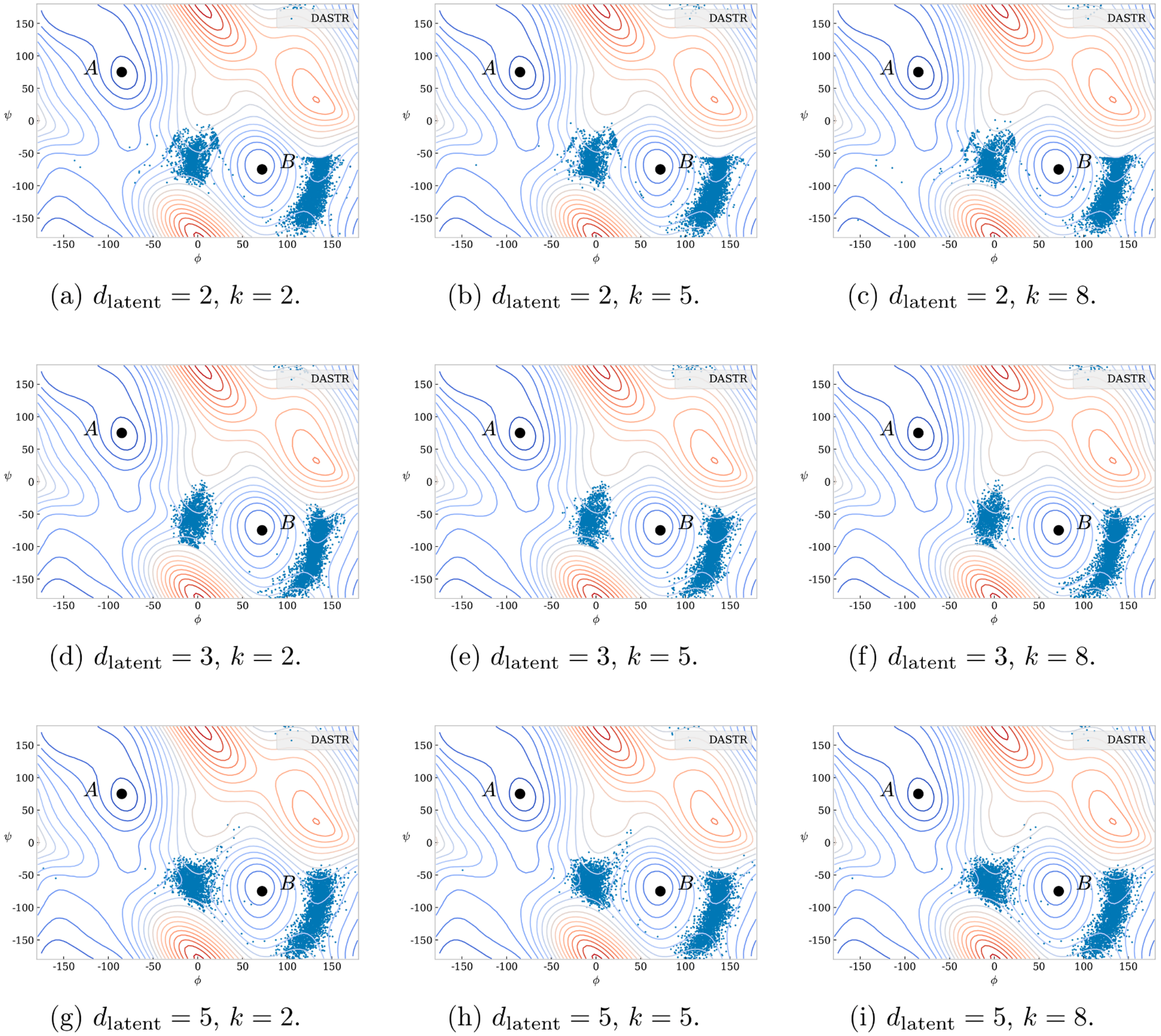
trajectories. Specifically, for each sample in Γ, we generate N_t trajectories by simulating the Langevin dynamics and use n to denote the number of trajectories ending up in region B before A. By counting the number of times of these points first reaching B before A, we can estimate q for such points by the definition of committor functions. If the trained model q_θ is indeed a good approximation of the committor function, the probability distribution of n/N_t should be close to a normal distribution with mean 0.5 and variance $(4N_t)^{-1}$ [7].

The results are shown in Figs. 10 and 11. In Fig. 10a–c, we show the candidate samples generated by DASTR. It is clear that these samples are located in the transition state region. To ensure that the samples obey the molecular configuration, we use the umbrella sampling method to refine them as shown in Fig. 10d and e. From Fig. 11a–c, it is seen that the probability distribution is not consistent with a normal distribution with mean 0.5, which means that the trained model using data from metadynamics fails to approximate the committor function near $q \approx 0.5$. Also, the number of points in Γ is much smaller than that of DASTR. This is due to the lack of sufficient samples in the transition state region, leading to the large generalization error in this area. In contrast, from Fig. 11d and e, it is seen that the approximate committor function values cluster around $1/2$, which indicates that our DASTR method performs significantly better and provides a good approximation on the $1/2$-isosurface.

4.3.2. DASTR with latent collective variables

In the previous experiment, the collective variables ϕ and ψ are given. We use KRnet to learn the features of ϕ and ψ in the transition state region. Such learned features are used for umbrella sampling to refine the training set. However, this still cannot avoid the need of SDE simulations after training deep generative models. In this section, we use autoencoders to learn the latent collective variables (CVs) that can help avoid repeated umbrella sampling simulations during sample generation.

As discussed in Section 3, the input of the autoencoder is the coordinates of the 10 heavy atoms of the alanine dipeptide. We perform self-supervised learning to train the autoencoder to learn the latent CVs. The KRnet is used to learn the distribution of the latent CVs in the transition state region, which is similar to the approach adopted in Section 4.3.1 except for the choice of the latent CVs. The settings of neural networks and training details can be found in Appendix B.3.

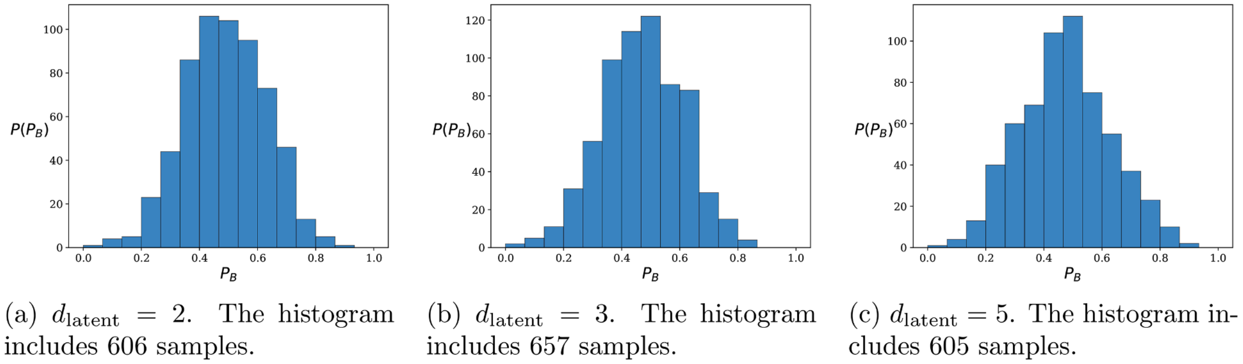


(a) $d_{\text{latent}} = 2,\ k = 2.$ (b) $d_{\text{latent}} = 2,\ k = 5.$ (c) $d_{\text{latent}} = 2,\ k = 8.$

(d) $d_{\text{latent}} = 3,\ k = 2.$ (e) $d_{\text{latent}} = 3,\ k = 5.$ (f) $d_{\text{latent}} = 3,\ k = 8.$

(g) $d_{\text{latent}} = 5,\ k = 2.$ (h) $d_{\text{latent}} = 5,\ k = 5.$ (i) $d_{\text{latent}} = 5,\ k = 8.$

Fig. 14. Samples during training for different latent dimensions, the alanine dipeptide test problem. The figures are shown that the samples (scatter plot) distributed on the energy landscape with respect to ϕ and ψ.



(a) $d_{\text{latent}} = 2$. The histogram includes 606 samples. (b) $d_{\text{latent}} = 3$. The histogram includes 657 samples. (c) $d_{\text{latent}} = 5$. The histogram includes 605 samples.

Fig. 15. Conducting DASTR in the latent space for the alanine dipeptide test problem: the histograms of the committor function values on the 1/2-th isosurface of q_θ for different latent dimensions.

In this experiment, we test three different latent dimensions $d_{\text{latent}} = 2, 3, 5$. In Fig. 12, we use UMAP [67] to project the data points onto a two-dimensional plane for visualization, where the points include the two metastable states A and B, samples from metadynamics, and the latent variables from DASTR at the final stage.

During the adaptive sampling procedure, we need to filter out those samples with excessively high potential energies. This will help avoid generating unreasonable molecular configurations. To this end, we set an energy threshold at 150 kJ/mol in this experiment. This means that any molecules with potential energies exceeding this threshold are discarded when generating new molecules during the adaptive sampling procedure. As a reference, we employ the umbrella sampling method in Section 4.3.1 to sample 1×10^5 points in the transition state region, yielding a maximum energy of approximately 115.5 kJ/mol. We generate 1×10^5 samples in the latent space and use the decoder to reconstruct the coordinates of the heavy atoms. The configuration can be completed after adding the hydrogen atoms by PyMOL [57]. For different latent dimensions $d_{\text{latent}} = 2, 3, 5$, the proportions of the samples with energies of less than 150 kJ/mol are approximately $97.52\,\%, 97.20\,\%$, and $97.49\,\%$ respectively. For comparison, we also train KRnet using the coordinates of the heavy atoms as the input, and then added hydrogen atoms using PyMOL. In this setting, about $2.3\,\%$ of the samples have energies of less than 3000 kJ/mol—most of the samples do not have physically reasonable configurations! Fig. 13 shows the comparison of proportions of valid molecular configurations between the vanilla DASTR and the DASTR in the latent space. It is clear that the sampling efficiency is improved significantly when applying DASTR in the latent space.

The decoding step requires almost no time when using the autoencoder to generate new molecules. The main time cost for this step is from the hydrogen atom completion in PyMOL, which is also negligible. In Table 3, we compare the time cost of conducting DASTR in the latent space and DASTR with umbrella sampling for different numbers of samples. One can observe that the time required to generate the molecules using the latent CVs is less than $4\,\%$ of that of the strategy in Section 4.3.1. With the autoencoder, one can apply the proposed DASTR method to the latent space. This technique eliminates the need for simulating SDE to obtain samples in the transition state region and significantly reduces the computational cost, as demonstrated in Table 3. As shown in Fig. 14, the generated samples are mainly located in the transition state region across the different latent dimensions studied. From Figs. 11 and 15, it is evident that the latter has a smaller variance and thus has a better approximation of the committor function on the $1/2$-isosurface.

To measure the quality of the trained model, we use the SciPy package to compute the Wasserstein distance between $\mathcal{N}\left(0.5, (4N_t)^{-1}\right)$ and the empirical distribution obtained by DASTR or metadynamics. Table 4 shows the Wasserstein distance for different methods with different settings. We observe that, when using metadynamics, the number of samples in Γ is significantly smaller than that of DASTR, which is primarily because there are far fewer training samples in the transition region, making the neural network model q_θ difficult to capture the transition information. For this alanine dipeptide test problem, our DASTR method outperforms metadynamics.

5. Conclusion

We have developed a novel deep adaptive sampling approach on rare transition paths (DASTR) for estimating the high-dimensional committor function. With DASTR, the scarcity of effective data points can be addressed, and the performance of neural network approximation for the high-dimensional committor function is improved significantly.

For high-dimensional realistic molecular systems, to address the issue that deep generative models alone may fail to generate physically reasonable molecular configurations, we apply DASTR to the latent space, where two options for selecting the latent variables are provided. The first option is to combine physically explicit collective variables with umbrella sampling, and the second is to train an autoencoder to find the latent collective variables. Compared to the samples from the directly approximated high-dimensional distribution, the two latent-space-based approaches take into account the physics either through domain-specific knowledge or data. Numerical experiments show that the second choice does not require domain-specific knowledge, except for data used to select the collective variables, potentially providing a generic strategy to deal with larger, more realistic molecular systems. Many questions remain open, especially regarding the correlation between representation learning and physically consistent sample generation. These questions will be left for future study.

CRediT authorship contribution statement

Yueyang Wang: Writing – original draft, Visualization, Software, Methodology, Investigation; **Kejun Tang:** Writing – original draft, Software, Project administration, Methodology, Investigation, Conceptualization; **Xili Wang:** Visualization, Software, Methodology, Investigation; **Xiaoliang Wan:** Writing – review & editing, Methodology, Investigation, Conceptualization; **Weiqing Ren:** Writing – review & editing, Conceptualization; **Chao Yang:** Resources.

Data availability

Data will be made available on request.

Declaration of competing interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

Acknowledgements

K. Tang is supported by NSFC grant 12501598 and the Natural Science Foundation of Hunan Province (2024JJ6003). X. Wan has been supported by NSF grant DMS-2513234. C. Yang has been supported by NSFC grant 12131002.

Appendix A. Derivation of variational formulation

Let $u = q + \gamma\eta$ be the result of a perturbation $\gamma\eta$ of q, where γ is small and η is a differentiable function. Since q is the minimizer of (3), for any η, we have

$$\begin{aligned}
0 &= \frac{1}{2}\frac{\partial}{\partial\gamma}|_{\gamma=0}\int_{\Omega\backslash(A\cup B)} |\nabla u(\boldsymbol{x})|^2 e^{-\beta V(\boldsymbol{x})} d\boldsymbol{x} \\
&= \int_{\Omega\backslash(A\cup B)} \nabla q(\boldsymbol{x})\cdot\nabla\eta(\boldsymbol{x}) e^{-\beta V(\boldsymbol{x})} d\boldsymbol{x} \\
&= \int_{\Omega\backslash(A\cup B)} \nabla\cdot\left(\nabla q(\boldsymbol{x})\eta(\boldsymbol{x})e^{-\beta V(\boldsymbol{x})}\right)d\boldsymbol{x} - \int_{\Omega\backslash(A\cup B)} \eta(\boldsymbol{x})\nabla\cdot\left(\nabla q(\boldsymbol{x})e^{-\beta V(\boldsymbol{x})}\right)d\boldsymbol{x} \\
&= -\int_{\Omega\backslash(A\cup B)} \eta(\boldsymbol{x})\nabla\cdot\left(\nabla q(\boldsymbol{x})e^{-\beta V(\boldsymbol{x})}\right)d\boldsymbol{x} \\
&= -\int_{\Omega\backslash(A\cup B)} \eta(\boldsymbol{x})e^{-\beta V(\boldsymbol{x})}(\Delta q(\boldsymbol{x}) - \beta\nabla V(\boldsymbol{x})\cdot\nabla q(\boldsymbol{x}))d\boldsymbol{x},
\end{aligned} \tag{A.1}$$

where the fourth equality follows from the integration by parts and the Neumann condition in (2). Because (A.1) holds for any η, we have $\Delta q(\boldsymbol{x}) - \beta\nabla V(\boldsymbol{x})\cdot\nabla q(\boldsymbol{x}) = 0$, which is the desired PDE form of the committor function.

Appendix B. Implementation details

B.1. Rugged Mueller potential

We choose a four-layer fully connected neural network q_θ with 100 neurons to approximate the solution. The activation function is chosen to be the hyperbolic tangent function for the hidden layers and the sigmoid function for the output layer. For KRnet, we take five blocks and eight affine coupling layers in each block. A two-layer fully connected neural network with 120 neurons is employed in each affine coupling layer. The activation function of KRnet is the rectified linear unit (ReLU) function. To generate points in $\Omega\backslash(A\cup B)$, we use the KRnet to learn the sampling distribution $p_{V,q}(\boldsymbol{x}) = |\nabla q_\theta(\boldsymbol{x})|^2 e^{-\beta V(\boldsymbol{x})}$ in the box $[-1.5, 1]\times[-0.5, 2]\times[-1, 1]^{d-2}$, and then remove points within the region A and B. This can be done by adding a logistic transformation layer [26] or a new coupling layer proposed in Zeng et al. [68]. We set $\lambda = 10$ in (4). The learning rate for the ADAM optimizer is set to 0.0001, with a decay rate 0.8 applied every 200 epochs for training q_θ and no decay for training KRnet, and the batch size is set to $m = m' = 5000$. The numbers of adaptivity iterations is set to $N_{\text{adaptive}} = 30$ when $N_e = N_e' = 50$ in Algorithm 1. In this test problem, we replace all the data points in the current training set with new samples.

It is difficult to sample in the transition state region when simulating the SDE. We implement the artificial temperature method as the baseline. More specifically, we increase the temperature by setting $\beta' = 1/20$ to obtain the modified SDE. This modified Langevin equation is solved by the Euler-Maruyama scheme with the time step $\Delta t = 10^{-5}$. With this setting, the data points are sampled from the trajectory of the modified Langevin equation. In this example, we compare the results obtained from DASTR with those from the artificial temperature method.

B.2. Standard Brownian motion

We choose a four-layer fully connected neural network q_θ with 100 neurons to approximate the solution, and the activation function of q_θ is set to the square of the hyperbolic tangent function. For KRnet, we take five blocks and eight affine coupling layers in each block. A two-layer fully connected neural network with 120 neurons is employed in each affine coupling layer. The activation function of KRnet is the rectified linear unit (ReLU) function. The learning rate for the ADAM optimizer is set to 0.001, with a decay rate 0.8 applied every 200 epochs for training q_θ and no decay for training KRnet. We set the number of adaptivity iterations to $N_{\text{adaptive}} = 30$, with $N_e = N_e' = 50$ training epochs per stage. The batch size for training q_θ is set to $m = 1000$ and for training the PDF model is set to $m' = 5000$. In the first stage, we generate N_0 uniform samples from $\Omega\backslash(A\cup B)$ and $N_0/2$ points each from ∂A and ∂B. For the remaining stages, we select $N_0/2$ points from the uniform samples and $N_0/2$ points from the deep generative model. We set $\lambda = 1000$ in (4).

We use the deep generative model to approximate $p_{V,q}(\boldsymbol{x}) = |\nabla q_\theta(\boldsymbol{x})|^2 e^{-\beta V(\boldsymbol{x})}$, where the probability density function induced by the deep generative model is defined in the box $[-2, 2]^d$. To ensure points in $\Omega\backslash(A\cup B)$, we just remove points within the region A and B generated by the deep generative model. For comparison, we also use the SDE to generate data points to train q_θ, where the Euler-Maruyama scheme with the time step $\Delta t = 10^{-6}$ is applied to get the trajectory.

B.3. Alanine dipeptide

DASTR with Explicit Collective Variables. In this test problem, we choose the dihedrals ϕ (with respect to C-N-CA-C), ψ (with respect to N-CA-C-N) as the collective variables (CVs). For this realistic example, it is not suitable to use the uniform samples as the initial training set, since uniform samples are not effective for solving this high-dimensional ($d = 66$) problem and also do not adhere to the molecular configuration. We use metadynamics to generate samples as the initial training set.

Metadynamics is an enhanced sampling technique to explore free energy landscapes of complex systems. The idea of metadynamics is to add a history-dependent biased potential to the system to discourage it from revisiting previously sampled states [42,43]. This is done by periodically depositing Gaussian potentials along the trajectory of the CVs. Mathematically, the Gaussian potential can be expressed as:

$$V_{G,t}(\boldsymbol{x}) = \sum_{t'=0,\tau,2\tau,\ldots}^{t'<t} w \exp\left(- \sum_{i=1}^{m} \frac{(s_i(\boldsymbol{x}) - s_i(\boldsymbol{x}_{t'}))^2}{2\sigma_i^2} \right), \tag{B.1}$$

where w is the height of the Gaussian potential, σ is the width of the Gaussian potential, m is number of CVs, and $s_i(\boldsymbol{x}_t)$ denotes the collective variables at time t. After adding the above Gaussian potential, we generate samples using the modified potential:

$$V_{\text{modified}}(\boldsymbol{x}) = V(\boldsymbol{x}) + V_{G,t}(\boldsymbol{x}),$$

where $V(\boldsymbol{x})$ is the original potential. That is, the biased potential in (7) is the Gaussian potential function $V_{G,t}$. During the simulation, the Gaussian potential lowers the energy barrier, allowing the system to explore more configurations of molecules. So, we can generate effective data points as the initial training set by metadynamics for this alanine dipeptide problem.

We simulate the Langevin dynamics with the time step $\Delta t = 0.1$ fs and a damping coefficient $1\,\text{ps}^{-1}$. One term of the Gaussian potential is added every 1000 steps, with parameters $w = 1.0\,\text{kJ/mol}$, $\sigma = 0.1$ rad. We finally get a total of 5000 terms in (B.1). Then we conduct the metadynamics with 7500 and 10,000 terms for comparison. Fig. B.16 shows that the more terms we add, the more thoroughly the free energy surface is explored, and the more samples we obtain in the transition state region. Samples are selected outside the regions A and B, and system configurations are saved to conduct the importance sampling step in (10). The simulation is conducted in OpenMM [69], a molecular dynamics simulation toolkit with high-performance implementation. Fig. B.16 shows the samples from the original dynamics and metadynamics. From this figure, it is clear that using metadynamics to generate initial data points is better since more samples are located in the transition state region.

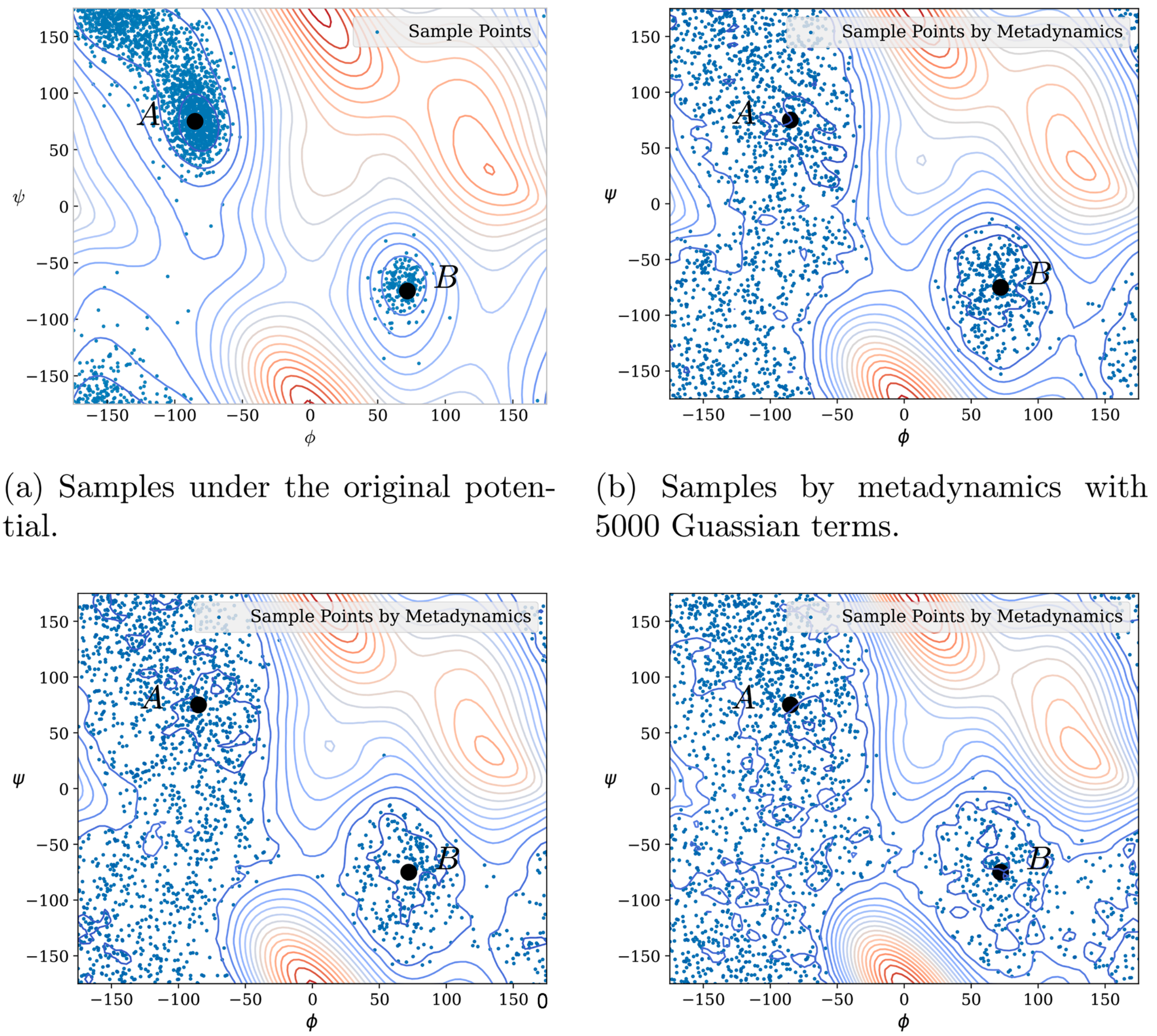
We choose a five-layer fully connected neural network q_θ (with 100, 120, 150 neurons) to approximate the solution, and the activation function for the hidden layers is set to the hyperbolic tangent function. The activation function for the output layer is the sigmoid function. Here, we only use the deep generative model to model the sampling distribution in terms of the collective variables ϕ and ψ. The trained KRnet is used to generate $s(\boldsymbol{x}_0) = [\phi, \psi]^\top$ in (13) (see B.4). For KRnet, we take one block and six affine coupling layers in each block. A two-layer fully connected neural network with 64 neurons is employed in each affine coupling layer. The activation function of KRnet is the rectified linear unit (ReLU) function. The learning rate for the ADAM optimizer is set to 0.0001, with a decay factor of 0.5 applied every 200 epochs for training q_θ and no decay for training KRnet. We set the batch size $m = 5000, m' = 10000$ and $N_e = 300, N_e' = 1000$. The numbers of adaptivity iterations is set to $N_{\text{adaptive}} = 10$. We sample 1.5×10^4 points in A and B respectively to enforce the boundary condition in the training process for all stages. We set $\lambda = 10$ in (4).

We employ KRnet to learn the sampling distribution in (7). In the first stage, we train the neural network q_θ using 2×10^5 points sampled by metadynamics. Then we use these points to train the PDF model induced by KRnet with support $[-180°, 180°]^2$, with the bias potential V_{bias} in (7) being the Gaussian potential $V_{G,t}$ defined in (B.1). In the rest of the stages, we train the neural network q_θ with 5×10^4 points sampled by umbrella sampling with the bias potential V_{US} (see B.4). We train the KRnet using the same sample points as those of training q_θ.

During the training procedure, we increase k_{us} in (13) from 200 kJ/mol to 400 kJ/mol. We sample 100 points for each target CVs in the umbrella sampling procedure. For comparison, we use the solution obtained by training a neural network q_θ with 150 neurons with 2×10^5 points sampled via metadynamics for 3000 epochs.

DASTR with Latent Collective Variables. In this experiment, both the encoder and decoder are implemented using fully connected neural networks. The encoder architecture is set as $[30, 100, 50, 50, 30, d_{\text{latent}}]$, while the decoder is set as $[d_{\text{latent}}, 30, 50, 50, 100, 30]$, with the Swish activation function. For training the autoencoder, we use 2×10^5 samples generated by metadynamics (with 10,000 terms in (B.1)) as the training set. The batch size is set to 1000. The model is trained with 5000 epochs.

The committor function is approximated by a five-layer fully connected neural network q_θ with 150 neurons, where the activation function for the hidden layers is set to the hyperbolic tangent function, and the activation function for the output layer is the sigmoid function. In this experiment, we use the deep generative model to model the probability distribution in terms of the latent CVs obtained from the autoencoder. The learning rate for the ADAM optimizer is set to 0.0001, with a decay factor of 0.5 applied every 200 epochs for training q_θ and no decay for training KRnet. The batch size is set to $m = 5000, m' = 10000$ and $N_e = 200, N_e' = 500$. In the first stage, we use 2×10^5 points sampled from metadynamics (10000 terms in (B.1)) as the initial dataset to train q_θ. In the rest stages, we use 1×10^5 points sampled from metadynamics and 1×10^5 points from KRnet and the pretrained autoencoder. Other settings are the same as those in Section 4.3.1.



(a) Samples under the original potential.

(b) Samples by metadynamics with 5000 Guassian terms.

(c) Samples by metadynamics with 7500 Guassian terms.

(d) Samples by metadynamics with 10000 Guassian terms.

Fig. B.16. Samples from the original dynamics and metadynamics.

B.4. Umbrella sampling

The umbrella sampling method is also an enhanced sampling technique. It introduces external biased potentials to pull the system out of local minima, thereby enabling a more uniform exploration of the entire free energy surface. This method is particularly effective in calculating free energy differences and studying reaction pathways in complex molecular processes. The umbrella sampling method employs a series of biased simulations, dividing the reaction space of collective variables into multiple overlapping windows, where each biased potential is applied in its corresponding window [41]. The umbrella potential is usually defined as:

$$V_{\mathrm{US}}(\boldsymbol{x}) = \frac{1}{2}\sum_{i=1}^{m} k_{\mathrm{us}}(s_i(\boldsymbol{x}) - s_i(\boldsymbol{x}_0))^2, \tag{B.2}$$

where $s_i(\boldsymbol{x})$ represents the CVs with respect to $\boldsymbol{x}$, m is the number of CVs, and k_{us} is the force constant. In this work, we focus on sampling in the final window, helping us effectively sample the desired regions of CVs. Therefore, we perform a rapid iterative process of umbrella sampling to transfer the CVs to the target region, and finally sample near the target CVs in the modified potential:

$$V_{\mathrm{modified}}(\boldsymbol{x}) = V(\boldsymbol{x}) + V_{\mathrm{US}}(\boldsymbol{x}),$$

where V is the original potential, and $s_i(\boldsymbol{x}_0)$ in (B.2) is the target CVs generated by the trained deep generative model.

References

[1] N. Okuyama-Yoshida, M. Nagaoka, T. Yamabe, Transition-state optimization on free energy surface: toward solution chemical reaction ergodography, Int. J. Quant.Chem. 70 (1) (1998) 95–103.

[2] W. E, E. Vanden-Eijnden, Towards a theory of transition paths, J. Stat. Phys. 123 (3) (2006) 503–523.

[3] A. Berteotti, A. Cavalli, D. Branduardi, F.L. Gervasio, M. Recanatini, M. Parrinello, Protein conformational transitions: the closure mechanism of a kinase explored by atomistic simulations, J. Am. Chem. Soc. 131 (1) (2009) 244–250.

[4] W. E, E. Vanden-Eijnden, Transition-path theory and path-finding algorithms for the study of rare events, Annu. Rev. Phys. Chem. 61 (2010) 391–420.

[5] R. Lai, J. Lu, Point cloud discretization of Fokker–Planck operators for committor functions, Multisc. Model. Simulat. 16 (2) (2018) 710–726.

[6] Q. Li, B. Lin, W. Ren, Computing committor functions for the study of rare events using deep learning, J. Chem. Phys. 151 (5) (2019) 054112.

[7] Y. Chen, J. Hoskins, Y. Khoo, M. Lindsey, Committor functions via tensor networks, J. Comput. Phys. 472 (2023) 111646.

[8] Y. Khoo, J. Lu, L. Ying, Solving for high-dimensional committor functions using artificial neural networks, Res. Math. Sci. 6 (1) (2019) 1–13.

[9] A. Megías, S.C. Arredondo, C.G. Chen, C. Tang, B. Roux, C. Chipot, Iterative variational learning of committor-consistent transition pathways using artificial neural networks, arXiv preprint arXiv:2412.01947. (2024).

[10] H. Chen, B. Roux, C. Chipot, Discovering reaction pathways, slow variables, and committor probabilities with machine learning, J. Chem. Theory Comput. 19 (14) (2023) 4414–4426.

[11] J. Strahan, J. Finkel, A.R. Dinner, J. Weare, Predicting rare events using neural networks and short-trajectory data, J. Comput. Phys. 488 (2023) 112152.

[12] J. Strahan, S.C. Guo, C. Lorpaiboon, A.R. Dinner, J. Weare, Inexact iterative numerical linear algebra for neural network-based spectral estimation and rare-event prediction, J. Chem. Phys. 159 (1) (2023).

[13] H. Li, Y. Khoo, Y. Ren, L. Ying, A semigroup method for high dimensional committor functions based on neural network, in: Mathematical and Scientific Machine Learning, PMLR, 2022, pp. 598–618.

[14] H. Li, Y. Khoo, Y. Ren, L. Ying, Solving for high dimensional committor functions using neural network with online approximation to derivatives, arXiv preprint arXiv:2012.06727. (2020).

[15] G.M. Rotskoff, A.R. Mitchell, E. Vanden-Eijnden, Active importance sampling for variational objectives dominated by rare events: consequences for optimization and generalization, in: Mathematical and Scientific Machine Learning, PMLR, 2022, pp. 757–780.

[16] M.R. Hasyim, C.H. Batton, K.K. Mandadapu, Supervised learning and the finite-temperature string method for computing committor functions and reaction rates, J. Chem. Phys. 157 (18) (2022).

[17] P. Kang, E. Trizio, M. Parrinello, Computing the committor with the committor to study the transition state ensemble, Nat. Comput. Sci. (2024) 1–10.

[18] B. Lin, W. Ren, Deep Learning Method for Computing Committor Functions with Adaptive Sampling, arXiv preprint arXiv:2404.06206. (2024).

[19] A.N. Singh, A. Das, D.T. Limmer, Variational path sampling of rare dynamical events, Annu. Rev. Phys. Chem. 76 (2025).

[20] A. Das, B. Kuznets-Speck, D.T. Limmer, Direct evaluation of rare events in active matter from variational path sampling, Phys. Rev. Lett. 128 (2) (2022) 028005.

[21] A.N. Singh, D.T. Limmer, Splitting probabilities as optimal controllers of rare reactive events, J. Chem. Phys. 161 (5) (2024).

[22] A.N. Singh, D.T. Limmer, Reactive path ensembles within nonequilibrium steady-states, arXiv preprint arXiv:2501.19233. (2025).

[23] L.J.S. Lopes, T. Lelièvre, Analysis of the adaptive multilevel splitting method on the isomerization of alanine dipeptide, J. Comput. Chem. 40 (11) (2019) 1198–1208.

[24] H. Jung, R. Covino, A. Arjun, C. Leitold, C. Dellago, P.G. Bolhuis, G. Hummer, Machine-guided path sampling to discover mechanisms of molecular self-organization, Nat. Comput. Sci. 3 (4) (2023) 334–345.

[25] W. Gao, C. Wang, Active learning based sampling for high-dimensional nonlinear partial differential equations, J. Comput. Phys. 475 (2023) 111848.

[26] K. Tang, X. Wan, C. Yang, DAS-PINNs: a deep adaptive sampling method for solving high-dimensional partial differential equations, J. Comput. Phys. 476 (2023) 111868.

[27] X. Wang, K. Tang, J. Zhai, X. Wan, C. Yang, Deep adaptive sampling for surrogate modeling without labeled data, J. Sci. Comput. 101 (3) (2024) 77. https://doi.org/10.1007/s10915-024-02711-1

[28] K. Tang, J. Zhai, X. Wan, C. Yang, Adversarial adaptive sampling: unify PINN and optimal transport for the approximation of PDEs, in: The Twelfth International Conference on Learning Representations, 2024.

[29] Z. Gao, L. Yan, T. Zhou, Failure-informed adaptive sampling for PINNs, SIAM J. Sci. Comput. 45 (4) (2023) A1971–A1994.

[30] Y. Jiao, D. Li, X. Lu, J.Z. Yang, C. Yuan, A Gaussian mixture distribution-based adaptive sampling method for physics-informed neural networks, Eng. Appl. Artif. Intell. 135 (2024) 108770.

[31] G. Czibula, A.-I. Albu, M.I. Bocicor, C. Chira, AutoPPI: an ensemble of deep autoencoders for protein–protein interaction prediction, Entropy 23 (6) (2021) 643.

[32] F.F. Alam, T. Rahman, A. Shehu, Learning reduced latent representations of protein structure data, in: Proceedings of the 10th ACM International Conference on Bioinformatics, Computational Biology and Health Informatics, 2019, pp. 592–597.

[33] A. Hawkins-Hooker, F. Depardieu, S. Baur, G. Couairon, A. Chen, D. Bikard, Generating functional protein variants with variational autoencoders, PLoS Comput. Biol. 17 (2) (2021) e1008736.

[34] M. Frassek, A. Arjun, P.G. Bolhuis, An extended autoencoder model for reaction coordinate discovery in rare event molecular dynamics datasets, J. Chem. Phys. 155 (6) (2021).

[35] J. Sirignano, K. Spiliopoulos, DGM: a deep learning algorithm for solving partial differential equations, J. Comput. Phys. 375 (2018) 1339–1364.

[36] M. Raissi, P. Perdikaris, G.E. Karniadakis, Physics-informed neural networks: a deep learning framework for solving forward and inverse problems involving nonlinear partial differential equations, J. Comput. Phys. 378 (2019) 686–707.

[37] G.E. Karniadakis, I.G. Kevrekidis, L. Lu, P. Perdikaris, S. Wang, L. Yang, Physics-informed machine learning, Nat. Rev. Phys. 3 (6) (2021) 422–440.

[38] W. E, B. Yu, The deep Ritz method: a deep learning-based numerical algorithm for solving variational problems, Commun. Math. Statist. 6 (1) (2018) 1–12.

[39] Y. Liao, P. Ming, Deep Nitsche method: deep Ritz method with essential boundary conditions, Commun. Comput. Phys. 29 (5) (2021) 1365–1384.

[40] Y. Lu, J. Lu, M. Wang, A priori generalization analysis of the deep Ritz method for solving high dimensional elliptic partial differential equations, in: Conference on Learning Theory, PMLR, 2021, pp. 3196–3241.

[41] J. Kästner, Umbrella sampling, Wiley Interdiscipl. Rev.: Comput. Molecul. Sci. 1 (6) (2011) 932–942.

[42] G. Bussi, A. Laio, Using metadynamics to explore complex free-energy landscapes, Nat. Rev. Phys. 2 (4) (2020) 200–212.

[43] A. Barducci, G. Bussi, M. Parrinello, Well-tempered metadynamics: a smoothly converging and tunable free-energy method, Phys. Rev. Lett. 100 (2) (2008) 020603.

[44] L. Dinh, J. Sohl-Dickstein, S. Bengio, Density estimation using real NVP, arXiv preprint arXiv:1605.08803. (2016).

[45] D.P. Kingma, P. Dhariwal, Glow: generative flow with invertible 1x1 convolutions, in: Advances in Neural Information Processing Systems, 2018, pp. 10215–10224.

[46] T.Q. Chen, Y. Rubanova, J. Bettencourt, D.K. Duvenaud, Neural ordinary differential equations, in: Advances in Neural Information Processing Systems, 2018, pp. 6571–6583.

[47] Y. Song, J. Sohl-Dickstein, D.P. Kingma, A. Kumar, S. Ermon, B. Poole, Score-based generative modeling through stochastic differential equations, in: International Conference on Learning Representations, 2021.

[48] K. Tang, X. Wan, Q. Liao, Adaptive deep density approximation for Fokker-Planck equations, J. Comput. Phys. 457 (2022) 111080.

[49] K. Tang, X. Wan, Q. Liao, Deep density estimation via invertible block-triangular mapping, Theoret. Appl. Mech. Lett. 10 (2020) 143–148.

[50] X. Wan, S. Wei, VAE-KRNet and its applications to variational Bayes, Commun. Comput. Phys. 31 (4) (2022) 1049–1082.

[51] X. Wan, K. Tang, Augmented KRnet for density estimation and approximation, arXiv preprint arXiv:2105.12866. (2021).

[52] P.-T. De Boer, D.P. Kroese, S. Mannor, R.Y. Rubinstein, A tutorial on the cross-entropy method, Ann. Oper. Res. 134 (1) (2005) 19–67.
[53] R.Y. Rubinstein, D.P. Kroese, The Cross-Entropy Method: a Unified Approach to Combinatorial Optimization, Monte-Carlo Simulation and Machine Learning, Springer Science & Business Media, 2013.
[54] G. Fiorin, M.L. Klein, J. Hénin, Using collective variables to drive molecular dynamics simulations, Mol. Phys. 111 (22–23) (2013) 3345–3362.
[55] Z. He, C. Chipot, B. Roux, Committor-consistent variational string method, J. Phys. Chem. Lett. 13 (40) (2022) 9263–9271.
[56] B. Roux, Transition rate theory, spectral analysis, and reactive paths, J. Chem. Phys. 156 (13) (2022).
[57] L. Schrödinger, The PyMOL molecular graphics system, version 1.8, (No Title) (2015).
[58] M. Alnæs, J. Blechta, J. Hake, A. Johansson, B. Kehlet, A. Logg, C. Richardson, J. Ring, M.E. Rognes, G.N. Wells, The FEniCS project version 1.5, Arch. Numer. Softw. 3 (100) (2015).
[59] A. Logg, K.-A. Mardal, G. Wells, Automated Solution of Differential Equations by the Finite Element Method: The FEniCS Book, 84, Springer Science & Business Media, 2012.
[60] C. Hartmann, O. Kebiri, L. Neureither, L. Richter, Variational approach to rare event simulation using least-squares regression, Chaos 29 (6) (2019).
[61] N. Nüsken, L. Richter, Interpolating between BSDEs and PINNs: deep learning for elliptic and parabolic boundary value problems, J. Mach. Learn. 2 (1) (2023) 31–64.
[62] R. Vershynin, High-Dimensional Probability: An Introduction with Applications in Data Science, 47, Cambridge University Press, 2018.
[63] J. Wright, Y. Ma, High-Dimensional Data Analysis with Low-Dimensional Models: Principles, Computation, and Applications, Cambridge University Press, 2022.
[64] S. Jo, T. Kim, V.G. Iyer, W. Im, CHARMM-GUI: A web-based graphical user interface for CHARMM, J. Comput. Chem. 29 (11) (2008) 1859–1865.
[65] B.R. Brooks, C.L. Brooks, III, A.D. Mackerell, Jr, L. Nilsson, R.J. Petrella, B. Roux, Y. Won, G. Archontis, C. Bartels, S. Boresch, et al., CHARMM: The biomolecular simulation program, J. Comput. Chem. 30 (10) (2009) 1545–1614.
[66] J. Lee, X. Cheng, S. Jo, A.D. MacKerell, J.B. Klauda, W. Im, CHARMM-GUI input generator for NAMD, GROMACS, AMBER, OpenMM, and CHARMM/OpenMM simulations using the CHARMM36 additive force field, Biophys. J. 110 (3) (2016) 641a.
[67] L. McInnes, J. Healy, J. Melville, Umap: Uniform manifold approximation and projection for dimension reduction, arXiv preprint arXiv:1802.03426. (2018).
[68] L. Zeng, X. Wan, T. Zhou, Bounded KRnet and its applications to density estimation and approximation, arXiv preprint arXiv:2305.09063. (2023).
[69] P. Eastman, J. Swails, J.D. Chodera, R.T. McGibbon, Y. Zhao, K.A. Beauchamp, L.-P. Wang, A.C. Simmonett, M.P. Harrigan, C.D. Stern, et al., OpenMM 7: rapid development of high performance algorithms for molecular dynamics, PLoS Comput. Biol. 13 (7) (2017) e1005659.