Targeted free energy estimation via learned mappings F

Cite as: J. Chem. Phys. **153**, 144112 (2020); https://doi.org/10.1063/5.0018903
Submitted: 18 June 2020 . Accepted: 25 September 2020 . Published Online: 13 October 2020

Peter Wirnsberger, Andrew J. Ballard, George Papamakarios, Stuart Abercrombie, Sébastien Racanière, Alexander Pritzel, Danilo Jimenez Rezende, and Charles Blundell

Targeted free energy estimation via learned mappings

Cite as: J. Chem. Phys. **153**, 144112 (2020); doi: 10.1063/5.0018903
Submitted: 18 June 2020 • Accepted: 25 September 2020 •
Published Online: 13 October 2020

Peter Wirnsberger,[a] Andrew J. Ballard,[a] George Papamakarios, Stuart Abercrombie, Sébastien Racanière, Alexander Pritzel, Danilo Jimenez Rezende, and Charles Blundell

AFFILIATIONS

DeepMind, London, United Kingdom

[a]Authors to whom correspondence should be addressed: pewi@google.com and aybd@google.com

ABSTRACT

Free energy perturbation (FEP) was proposed by Zwanzig [J. Chem. Phys. **22**, 1420 (1954)] more than six decades ago as a method to estimate free energy differences and has since inspired a huge body of related methods that use it as an integral building block. Being an importance sampling based estimator, however, FEP suffers from a severe limitation: the requirement of sufficient overlap between distributions. One strategy to mitigate this problem, called Targeted FEP, uses a high-dimensional mapping in configuration space to increase the overlap of the underlying distributions. Despite its potential, this method has attracted only limited attention due to the formidable challenge of formulating a tractable mapping. Here, we cast Targeted FEP as a machine learning problem in which the mapping is parameterized as a neural network that is optimized so as to increase the overlap. We develop a new model architecture that respects permutational and periodic symmetries often encountered in atomistic simulations and test our method on a fully periodic solvation system. We demonstrate that our method leads to a substantial variance reduction in free energy estimates when compared against baselines, without requiring any additional data.

I. INTRODUCTION

Free energy estimation is of central importance in the natural sciences. Accurate estimation of free energies, however, is challenging, as many systems are out of reach of experimental methods and analytic theory. Computer-based estimation has, thus, emerged as a valuable alternative. Successful application areas of *in silico* free energy estimation span industry and scientific research, including drug discovery,[1] condensed matter physics,[2] materials science,[3] structural biology,[4] and the effects of mutagenesis.[5] Because of its importance and wide ranging applications, computer-based free energy estimation has been an active field of research for decades.[6]

At the core of many state-of-the-art estimators[7] lies the free energy perturbation (FEP) identity introduced by Zwanzig[8] in 1954,

$$\mathbb{E}_A\left[e^{-\beta\Delta U}\right] = e^{-\beta\Delta F}. \tag{1}$$

Here, $\Delta F = F_B - F_A$ is the Helmholtz free energy difference between two thermodynamic states A and B, each connected to a thermal reservoir at the inverse temperature β. We denote $\mathbf{x}$ as a point in the system's configuration space and define

$$\Delta U(\mathbf{x}) = U_B(\mathbf{x}) - U_A(\mathbf{x}), \tag{2}$$

with $U_A = U(\mathbf{x}; \lambda_A)$ and $U_B = U(\mathbf{x}; \lambda_B)$ being the energy functions associated with A and B, respectively. The parameters λ_A and λ_B could, for example, represent the coupling coefficients of a particle–particle interaction potential. By $\mathbb{E}_A[\cdots]$, we denote an expectation under equilibrium distribution $\rho_A \propto e^{-\beta U_A}$ (and similarly for B). A procedure for computation of ΔF via Eq. (1) is then as follows: a number of samples are first drawn from ρ_A, for example, via a Markov chain Monte Carlo or molecular dynamics simulation. The change in energy, ΔU, associated with instantaneously switching $\lambda_A \to \lambda_B$ is then computed, from which an exponential average is taken. From a statistical point of view, FEP is an application of Importance Sampling (IS), where ρ_A serves as the proposal distribution.[9–11]

While Eq. (1) is exact, the convergence of this estimator for a finite number of samples strongly depends on the degree to which

A and B overlap in configuration space.[12] Indeed, the dominant contributions to the above expectation will come from samples of A that are typical under B, and such contributions become increasingly rare with decreasing overlap.[13]

There exist multiple strategies for mitigating the overlap requirement. Arguably, the most common strategy is a multi-staged approach, in which a sequence of intermediate thermodynamic states is defined between A and B [Fig. 1(a)]. Here, the increased convergence is facilitated by demanding that neighboring pairs of states be chosen to contain sufficient overlap. The quantity of interest, ΔF, is then recovered as a sum over the pairwise differences $\Delta F_{i,i+1}$.[6] The Multistate Bennett Acceptance Ratio[7] (MBAR) estimator is a prominent example of an estimator that follows this strategy. However, multi-staged approaches require samples from multiple states as well as a suitable order parameter to define intermediate states. Furthermore, it is unclear *a priori* how best to discretize the order parameter or how many stages to use.

An alternative, elegant strategy to increasing the overlap is by incorporating configuration space maps. Jarzynski developed Targeted Free Energy Perturbation[14] (TFEP), a generalization of FEP whereby an invertible mapping defined on configuration space transports points sampled from A to a new distribution, A' [see Fig. 1(b)]. Jarzynski showed that a generalized FEP identity can be applied to this process, from which the free energy difference can be recovered. Importantly, if the mapping is chosen wisely, an effective overlap can be increased, leading to quicker convergence of the TFEP estimator. Hahn and Then extended TFEP to the bidirectional setting,[15] whereby the mapping and its inverse are applied to samples from A and B, respectively. Lower-error free energy estimates can then be obtained via the statistically optimal BAR estimator.[16]

Whether in the unidirectional or bidirectional case, the main challenge for targeted approaches is crafting a mapping that is capable of increasing the overlap. Unfortunately, for most real-world problems, the physical intuition needed to develop such a technique is simply lacking. Modern-day machine learning (ML) techniques, however, seem perfectly suited for this task.

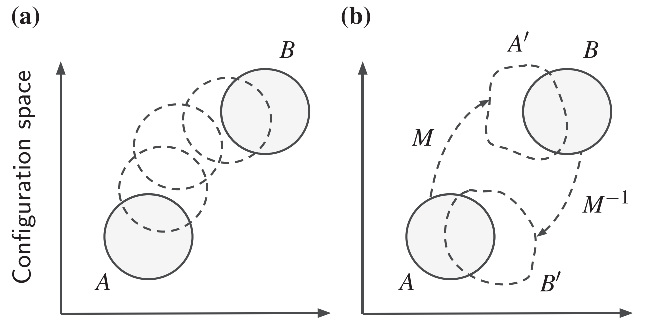

FIG. 1. The overlap problem. Efficient estimates of free energy differences rely on a decent configuration-space overlap between equilibrium distributions. Each panel represents a set of distributions on configuration space, with the circles indicating the regions of appreciable mass. (a) In the commonly used multi-staged approach, a chain of intermediate states is defined between A and B such that a decent overlap exists between each neighboring pair. Free energy differences are computed with respect to each neighboring pair and summed to obtain the difference of interest. (b) In the targeted approach, an invertible mapping M transports the equilibrium distribution A to a new distribution A'. Convergence is improved upon if the mapping results in an increased overlap with B and similarly for the reverse direction.

Since the introduction of TFEP in 2002, research in ML has made remarkable progress in fields of image classification,[17,18] playing video[19] and board games,[20,21] and generative modeling of images.[22,23] ML has also enabled advances in the natural sciences, including state-of-the-art protein structure prediction,[24] neural-network based molecular force fields,[25,26] generative modeling of lattice field theories,[27] new paradigms for sampling equilibrium distributions of molecules,[28] and variational free energy estimates.[29,30]

In this work, we turn targeted free energy estimation into a machine learning problem. *In lieu* of a hand-crafted mapping, we represent our mapping by a deep neural network whose parameters are optimized so as to maximize the overlap. Once trained, the free energy can then be computed by evaluating the targeted estimator with our learned mapping. Below we will consider both unidirectional and bidirectional settings and will refer to them as *Learned Free Energy Perturbation* (LFEP) and *Learned Bennett Acceptance Ratio* (LBAR), respectively.

A key contribution of this work is the development of a mapping that respects the underlying symmetries of our system of study. In particular, our neural network is equivariant to the permutation of identical particles and respects periodic boundary conditions by construction. These are particularly important considerations when modeling atomic systems, as they often obey such symmetries.

The rest of our manuscript is structured as follows. In Sec. II, we summarize the previously developed targeted free energy estimators that we will be making use of. This is followed by the development of suitable training objectives to maximize the overlap (Sec. III). We then demonstrate our method by applying it to a solvation system. In Sec. IV, we describe the experimental setup and discuss inherent symmetries that are exploited to devise a model with the correct inductive biases (Sec. V). Finally, we present the experimental results in Sec. VI and discuss our findings in Sec. VII.

II. THEORETICAL BACKGROUND

In the following, we will refer to A and B as the thermodynamic states, defined by equilibrium densities $\rho_A(\mathbf{x}) = e^{-\beta U_A(\mathbf{x})}/Z_A$ and $\rho_B(\mathbf{x}) = e^{-\beta U_B(\mathbf{x})}/Z_B$, where Z_A and Z_B are the normalization constants (partition functions). In the targeted scheme, configurations $\mathbf{x}$ are drawn from A and mapped to new configurations $\mathbf{y} = M(\mathbf{x})$ via an invertible, user-specified mapping M. The set of mapped configurations can be thought of as samples from a new state A',

$$M : A \to A'. \tag{3}$$

Similarly, we also consider the reverse case where configurations are drawn from B and mapped to B' via the inverse

$$M^{-1} : B \to B'. \tag{4}$$

We refer to this pair of prescriptions as the "forward" and "reverse" process, respectively. For each process, we denote generalized energy differences as

$$\Phi_F(\mathbf{x}) = U_B(M(\mathbf{x})) - U_A(\mathbf{x}) - \beta^{-1} \log|J_M(\mathbf{x})|, \tag{5}$$

$$\Phi_R(\mathbf{x}) = U_A\big(M^{-1}(\mathbf{x})\big) - U_B(\mathbf{x}) - \beta^{-1} \log|J_{M^{-1}}(\mathbf{x})|, \tag{6}$$

where J_M and $J_{M^{-1}} = J_M^{-1}$ are the Jacobian determinants associated with the mappings. As originally shown by Jarzynski,[14] an identity exists, which relates ΔF to an ensemble of realizations of Φ_F,

$$\mathbb{E}_A\Big[e^{-\beta\Phi_F}\Big] = e^{-\beta\Delta F}. \tag{7}$$

Equation (7) can be regarded as a generalization of FEP, as it holds for any invertible M, and reduces to Eq. (1) if M is the identity. An analogous equation holds for the reverse process. Derivations of Eq. (7) can be found in Refs. 14 and 15 or Appendix A.

Hahn and Then extended the above result to the bidirectional case,[15] showing that a fluctuation theorem (FT) exists between the forward and reverse processes

$$\frac{p_F(\phi)}{p_R(-\phi)} = e^{\beta(\phi-\Delta F)}. \tag{8}$$

The functions

$$p_F(\phi) = \int \mathrm{d}\mathbf{x}\, \rho_A(\mathbf{x})\delta(\phi - \Phi_F(\mathbf{x})) \tag{9}$$

and

$$p_R(\phi) = \int \mathrm{d}\mathbf{x}\, \rho_B(\mathbf{x})\delta(\phi - \Phi_R(\mathbf{x})) \tag{10}$$

can be thought of as generalized work distributions associated with the mapping processes and δ is the Dirac delta function. With these bidirectional estimates, Bennett's Acceptance Ratio (BAR) method[16] can be employed as an alternative estimator of ΔF.[15] BAR estimation of ΔF can be formulated as a self-consistent iteration of the equation

$$\mathbb{E}_A[f(\beta(\Phi_F - \Delta F))] = \mathbb{E}_B[f(\beta(\Phi_R + \Delta F))], \tag{11}$$

where $f(x) = 1/(1 + e^x)$ is the Fermi function. For simplicity, in Eq. (11), we restrict ourselves to the case where the number of samples in the forward and reverse directions is equal, but more general formulations exist. BAR has a statistical advantage over FEP as it has been shown to be the minimum variance free energy estimator for any asymptotically unbiased method.[31] Because of this property, BAR is generally the method of choice when samples from both A and B are available.

In summary, our method proceeds in two stages by first computing optimized work values using Eqs. (5) and (6) and then estimating ΔF. For the latter, we can employ the generalized FEP estimator (7) in the unidirectional setting (LFEP) or solve the BAR equations (11) in the bidirectional setting (LBAR). This highlights an important difference between LBAR and other maximum likelihood free energy estimators that assume the work values to be fixed.[7,16,32] Instead, LBAR learns to optimize the work values and subsequently combines them optimally to predict ΔF.

Crucially, the targeted estimators above hold for *every* invertible mapping. That is, given an infinite number of samples, any invertible choice of M will produce a consistent estimate of ΔF. Of course, the finite-sample convergence properties are of more practical importance and will strongly depend on the choice of M.

III. TRAINING OBJECTIVE

In a distributional sense, the forward and reverse processes act to transform ρ_A and ρ_B into $\rho_{A'}$ and $\rho_{B'}$, as depicted in Fig. 1. In what follows, we will refer to the distribution of mapped configurations as the "images" (i.e., $\rho_{A'}$ and $\rho_{B'}$), and the distributions we want them mapped toward as the "targets" [i.e., ρ_B (ρ_A) for the forward (reverse) process]. Due to the deterministic mapping, the bases and images are related by the change of variable formula,

$$\rho_{A'}(M(\mathbf{x})) = \rho_A(\mathbf{x})/|J_M(\mathbf{x})|, \tag{12}$$

$$\rho_{B'}\big(M^{-1}(\mathbf{x})\big) = \rho_B(\mathbf{x})/|J_{M^{-1}}(\mathbf{x})|. \tag{13}$$

The crucial consideration for convergence of our estimators [Eqs. (7) and (11)] is the overlap between the image and target distributions.[14] Indeed, in the limit that images and targets coincide, Jarzynski showed[14] that $p_F(\phi) \to \delta(\phi - \Delta F)$. This implies that the convergence of Eq. (7) is immediate (i.e., only one sample is needed). In this limit, it is also the case that $p_R(\phi) \to \delta(\phi + \Delta F)$ implying that the expectation values on either side of Eq. (11) converge immediately. This overlap argument is further reinforced in Appendix A, where we show that the TFEP estimator can be interpreted as an FEP estimator between A' and B.

We now turn our attention to the construction of a loss function that accurately judges the quality of M. Guided by the considerations of overlap, we consider the Kullback–Leibler (KL) divergence between the image and target. For the forward process, we have

$$\begin{aligned} D_{\mathrm{KL}}[\rho_{A'}\|\rho_B] &= \mathbb{E}_{\mathbf{x}\sim A'}\left[\log \frac{\rho_{A'}(\mathbf{x})}{\rho_B(\mathbf{x})}\right] \\ &= \mathbb{E}_{\mathbf{x}\sim A}\left[\log \frac{\rho_{A'}(M(\mathbf{x}))}{\rho_B(M(\mathbf{x}))}\right] \\ &= \mathbb{E}_{\mathbf{x}\sim A}\left[\log \frac{\rho_A(\mathbf{x})/|J_M(\mathbf{x})|}{\rho_B(M(\mathbf{x}))}\right] \\ &= \mathbb{E}_{\mathbf{x}\sim A}\left[\beta\Phi_F(\mathbf{x}) + \log\frac{Z_B}{Z_A}\right] \\ &= \beta(\mathbb{E}_A[\Phi_F] - \Delta F). \end{aligned} \tag{14}$$

In the above derivation, we invoked a change of variable formula in going from the first to third lines, and used the identity $-\beta\Delta F = \log Z_B - \log Z_A$ to get to the last. An analogous equation can be derived for the reverse process yielding

$$D_{\mathrm{KL}}[\rho_{B'}\|\rho_A] = \beta(\mathbb{E}_B[\Phi_R] + \Delta F). \tag{15}$$

While from Eqs. (14) and (15) it is clear that the KL cannot be accurately estimated unless ΔF is known, in terms of optimizing, ΔF and β can be disregarded as they are constants.

Below we consider two separate training regimes for our model. In the unidirectional case, the model was trained only on the forward process, with a loss function

$$\mathscr{L}_{\text{LFEP}} = \mathbb{E}_A[\Phi_F]. \tag{16}$$

In the bidirectional case, the model was trained using both forward and reverse processes with the loss

$$\mathscr{L}_{\text{LBAR}} = \mathbb{E}_A[\Phi_F] + \mathbb{E}_B[\Phi_R]. \tag{17}$$

When samples from both states are available, bidirectional training is preferable. Unlike the unidirectional loss, $\mathscr{L}_{\text{LBAR}}$ explicitly encourages both $\rho_{A'}$ and $\rho_{B'}$ to be mass-covering,[33] which is important for good performance of importance-sampling estimators.[34]

IV. EXPERIMENTAL SETUP

To test our method, we consider a system similar to the one used by Jarzynski[14] consisting of a repulsive solute immersed in a bath of $N = 125$ identical solvent particles. The task is to calculate the free energy change associated with growing the solute radius from R_A to radius R_B (see Fig. 2). In contrast to a hard-sphere solute as used in Ref. 14, we modeled our solute as a soft sphere. This is because any finite particle overlap would lead to infinite forces, which our training method cannot handle.

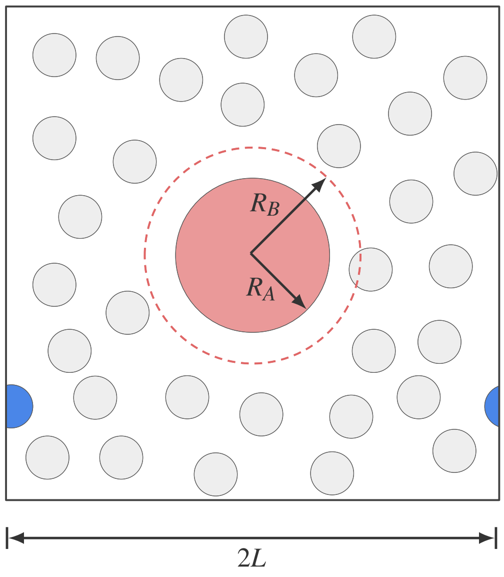

FIG. 2. Illustration of the simulation box. A solute particle (pink), located at the center, is grown from radius R_A to radius R_B, thereby compressing the space accessible to the $N = 125$ solvent particles (gray). Opposite faces of the cubic simulation box with edge length $2L$ are associated with each other such that a particle gets inserted again on the opposite side as it tries to leave the reference box (blue). The reference box is, therefore, topologically equivalent to a torus.

Intuitively, an effective mapping should push solvent particles away from the center to avoid high-energy steric clashes with the expanding solute. Jarzynski followed this intuition, defining a mapping that uniformly compresses the solvent particles amidst an expanding repulsive solute. Although this mapping gave a significant convergence gain when applied to a hard solute,[14] it is not directly applicable to soft solutes. This is because the phase space compression results in a transformed density whose support is not equal to that of the target density, violating the assumption of invertibility.

Below we demonstrate that we no longer need to rely on physical intuition to hand-craft a tractable mapping; this process can be fully automated using the general framework proposed in this work. In order to learn an effective mapping, however, it is crucial that the model be compatible with the inherent symmetries of the underlying physical system.

A. Energy

Periodic boundary conditions (PBCs) confine the N-particle system to a $3N$-dimensional torus, $\mathbf{x} = (r_1, \ldots, r_N) \in \mathbb{T}^{3N}$, where each coordinate of the position vector $r_i \in \mathbb{T}^3$ of particle i lives in the interval $[-L, L]$ (see Fig. 2). The total energy can then be decomposed into a sum of pairwise contributions according to

$$U_\alpha(\mathbf{x}) = \sum_{i=1}^{N}\sum_{j<i} u(|r'_{ij}|) + \sum_{i=1}^{N} v(|r_i|; R_\alpha), \tag{18}$$

where subscript $\alpha \in \{A, B\}$ denotes the state and $r_{ij} = r_j - r_i$. The quantity $r'_{ij} = r_{ij} - 2L\ \text{round}(r_{ij}/(2L))$ is a difference vector whose components correspond to the shortest, signed distance in the respective dimension, and the function round is applied element-wise giving the nearest integral number. The radially symmetric functions $u(r)$ and $v(r)$ represent the Lennard-Jones[35] (LJ) and Weeks–Chandler–Andersen[36] (WCA) potentials. In practice, we truncate LJ interactions using a radial cutoff of L and shift the potential to be zero at the cutoff.[37]

B. Training data

The data used for training and evaluation were generated via molecular dynamics (MD) simulations using the package LAMMPS.[38] Simulation frames were generated via Langevin dynamics and were saved infrequently enough to ensure decorrelated samples (as judged by the potential energy decorrelation time), giving a total of 5×10^4 frames per simulation. A total of 20 independent simulation trajectories were generated for each thermodynamic state, starting from randomly initialized configurations and momenta. Ten independent training and evaluation datasets were then constructed from these simulations by first concatenating and shuffling configurations, and then partitioning them into $N_{\text{train}} = 9 \times 10^4$ training and $N_{\text{test}} = 1 \times 10^4$ test samples for each dataset. The value of N_{train} was chosen such that the baseline BAR estimator, when evaluated on N_{train} data points, yields a reasonably accurate free energy difference. Below we train and evaluate our model on these datasets so as to compute statistical convergence properties of the estimators across independent runs. Further simulation details are summarized in Appendix B.

C. Symmetries

The system under consideration exhibits properties that are widely encountered in the atomistic simulation community: periodic boundary conditions (PBCs) and permutation invariance. PBCs are usually employed to reduce finite-size effects. Permutation invariance arises as a consequence of the energy being invariant to particle permutations—a condition that is satisfied by the solvent particles as they are all identical.

PBCs are a choice of geometry for the space in which particles live: a 3D torus. Without them, the system (including the solute) would admit rigid rotation, reflection, and rigid translation symmetries. With them, only the translation symmetries remain and a discrete set of rotations/reflections. The original rigid translations in $\mathbb{R}^3$ become translations by elements of the torus $\mathbb{T}^3$, leaving the energy invariant. Because of this symmetry, we can fix the solute at the origin of the simulation box without affecting the ratio Z_B/Z_A (as shown in Fig. 2). The remaining set of discrete rotations/reflections is the group of symmetries of a cube (the *octahedral group*), which contains only 48 elements.

We design the mapping M to respect PBCs and permutation invariance by construction. This means that the state A' obtained by transforming A via M is guaranteed to have the 3D torus geometry stipulated by PBCs and to be symmetric with respect to any permutation of solvent particles. Section V discusses in detail how these symmetries are implemented in the model architecture.

Our model architecture does not obey the octahedral symmetries, in the sense that A' is not guaranteed to be symmetric with respect to the 48 permutations and/or reflections of the three coordinate axes. However, since the size of this symmetry group is small, we account for the octahedral symmetries via *training-data augmentation* instead. That is, during the training, we transform every training data point (system configuration) by a random element of the octahedral group. As the training progresses, all 48 transformations of each data point are likely to be seen by the model; thus, the model is trained to learn these symmetries from data. In comparison, exhausting the $N! = 125!$ permutation symmetries by training-data augmentation would be infeasible in any reasonable training time.

V. MODEL

We implement the mapping M using a deep neural network, parameterized by a set of learnable parameters θ. In designing the architecture of the network, we take into account the following considerations.

(a) The mapping M must be bijective, and the inverse mapping M^{-1} should be efficient to compute, for any setting of the parameters θ.
(b) The Jacobian determinant J_M should be efficient to compute for any setting of θ.
(c) The network should be flexible enough to represent complex mappings.
(d) The transformed distributions $\rho_{A'}$ and $\rho_{B'}$ should respect the boundary conditions and symmetries of the physical system.

The first three requirements are satisfied by a class of deep neural networks known as *normalizing flows*,[39] which are invertible networks with efficient Jacobian determinants. Since bijectivity is a closed property under function composition, multiple normalizing flows (or "layers") can be composed into a deeper flow, yielding a model with increased flexibility. We implement M as a normalizing flow composed of K invertible layers, that is,

$$M = M_K \circ \cdots \circ M_1. \tag{19}$$

Each layer $M_k : \mathbb{T}^{3N} \to \mathbb{T}^{3N}$ is of the same architecture but it has its own learnable parameters θ_k, and the learnable parameters of M are simply $\theta = (\theta_1, \ldots, \theta_K)$.

Our implementation of M_k is based on the architecture proposed by Dinh *et al.*,[40] which is often referred to as the *coupling layer*. Let r_i^ν, $\nu \in \{1, 2, 3\}$, be the spatial coordinates of the particle with position vector $r_i \in \mathbb{T}^3$. To simplify the notation, we will also use r_i^ν to denote the inputs to layer k (that is, the particles transformed by $M_{k-1}\circ\cdots\circ M_1$) and drop the dependence on k. Let $\mathbb{I}_k$ be a subset of the indices $\{1, 2, 3\}$ associated with layer k. Then, M_k is defined as follows:

$$M_k\big(r_i^\nu\big) = \begin{cases} r_i^\nu & \nu \in \mathbb{I}_k \\ G(r_i^\nu; \psi_i^\nu) & \nu \notin \mathbb{I}_k, \end{cases} \tag{20}$$

where

$$\psi_i^\nu = C_i^\nu\Big(r_1^{\mathbb{I}_k}, \ldots, r_N^{\mathbb{I}_k}; \theta_k\Big). \tag{21}$$

By $r_i^{\mathbb{I}_k}$, we refer to the set of coordinates of r_i that are indexed by $\mathbb{I}_k$, and C_i^ν is the output of C for coordinate ν of particle i. The functions G and C are implemented by neural networks. The parameters of G associated with coordinate ν of particle i are denoted by ψ_i^ν and computed by C; we have again dropped the dependence of ψ_i^ν on k to simplify the notation. The parameters of C are the learnable parameters of the layer, θ_k. An illustration of the coupling layer defined by Eqs. (20) and (21) is shown in Fig. 3.

In simple terms, M_k works as follows. We partition the coordinates into two sets; the coordinates indexed by $\mathbb{I}_k$ are left invariant, whereas the remaining coordinates are transformed element-wise.

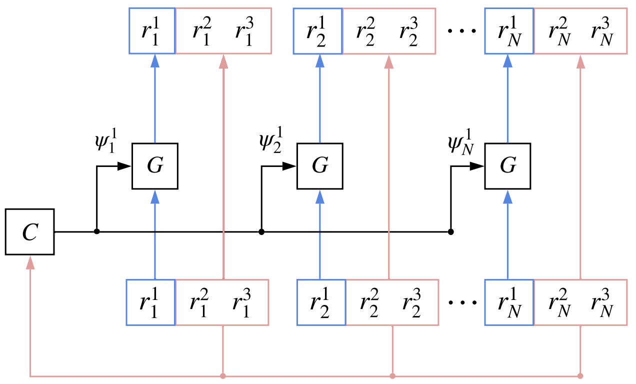

FIG. 3. Illustration of a coupling layer M_k. A subset of coordinates, here $\mathbb{I}_k = \{2, 3\}$, remains unchanged while all other coordinates are transformed by a *circular spline* G. The parameters ψ_i^ν of G are produced by a separate model C, which we implemented using the *transformer* architecture.[41]

Each transformed coordinate undergoes a different transformation depending on the value of ψ_i^ν, which itself is a function of all the coordinates that remain invariant. To ensure that each coordinate gets the opportunity to be transformed as a function of every other coordinate, each layer M_k uses a different partition $\mathbb{I}_k$, and we cycle through the partitions {1}, {2}, {3}, {1, 2}, {2, 3}, and {1, 3} across layers.

For the mapping M to be bijective, a sufficient condition is that $G(\cdot;\psi):[-L,L]\to[-L,L]$ be strictly increasing for any setting of ψ. In that case, the inverse M_k^{-1} is obtained by simply replacing G with G^{-1} in Eq. (20), and the Jacobian determinants can be computed efficiently as follows:

$$\log|J_{M_k}| = \sum_{i=1}^{N}\sum_{\nu\notin\mathbb{I}_k}\log\frac{\partial G}{\partial r_i^\nu}(r_i^\nu;\psi_i^\nu). \tag{22}$$

Finally, the inverse and the Jacobian determinant of the composite mapping M can be computed by

$$M^{-1} = M_1^{-1}\circ\cdots\circ M_K^{-1}, \tag{23}$$

$$\log|J_M| = \sum_{k=1}^{K}\log|J_{M_k}|. \tag{24}$$

To ensure that the transformed distributions $\rho_{A'}$ and $\rho_{B'}$ obey the required boundary conditions, the implementation of G must reflect the fact that $r_i^\nu = -L$ and $r_i^\nu = L$ are identified as the same point. For this to be the case, a sufficient set of conditions is as follows:

$$G(\pm L;\psi) = \pm L, \tag{25}$$

$$\frac{\partial G}{\partial r_i^\nu}(-L;\psi) = \frac{\partial G}{\partial r_i^\nu}(L;\psi) > 0, \tag{26}$$

for any setting of the parameters ψ. To satisfy the above conditions, we implement G using *circular splines*, which were recently proposed by Jimenez Rezende *et al.*[42] and are based on the rational-quadratic spline flows of Durkan *et al.*[43] Our implementation of the circular splines is detailed in Appendix C.

In addition to the above, we also need to make sure that the parameters ψ_i^ν in Eq. (21) are periodic functions of the invariant coordinates $r_1^{\mathbb{I}_k},\ldots,r_N^{\mathbb{I}_k}$. This can be easily achieved by the following feature transformation:

$$r_i^{\mathbb{I}_k}\mapsto\left(\cos\left(\frac{\pi}{L}r_i^{\mathbb{I}_k}\right),\sin\left(\frac{\pi}{L}r_i^{\mathbb{I}_k}\right)\right), \tag{27}$$

where cos and sin act element-wise. The above feature transformation is injective, so no information about $r_i^{\mathbb{I}_k}$ is lost. We apply this feature transformation to each particle i at the input layer of network C.

Finally, to ensure that the transformed distributions $\rho_{A'}$ and $\rho_{B'}$ are invariant to particle permutations, it is necessary that the Jacobian determinant J_M also be invariant to particle permutations. In our architecture, this can be achieved by taking C to be *equivariant* to particle permutations. Specifically, let σ be a permutation of the set {1, . . ., N}. We say that C is equivariant with respect to σ if

$$\psi_{\sigma(i)}^\nu = C_i^\nu\left(r_{\sigma(1)}^{\mathbb{I}_k},\ldots,r_{\sigma(N)}^{\mathbb{I}_k};\theta_k\right), \tag{28}$$

that is, if permuting the particles has the effect of permuting the parameter outputs $(\psi_1^\nu,\ldots,\psi_N^\nu)$ in exactly the same way. From Eq. (22), we can easily see that the above property implies that σ leaves J_{M_k}, and hence J_M, invariant, because we sum over all particles and the sum is permutation invariant. Previous studies have made similar observations.[44,45] Our implementation of C is based on the architecture proposed by Vaswani *et al.*,[41] often referred to as the *transformer*, which we use in a permutation-equivariant configuration. The implementation details of our transformer architecture are in Appendix C.

VI. RESULTS

In this section, we evaluate the performance of our method for the solvation system illustrated in Fig. 2. We focus on the bidirectional BAR and LBAR estimators in the main text, due to their advantages over unidirectional approaches as discussed in Sec. II. We refer to Appendix D for a discussion of the unidirectional counterparts.

To capture statistical variation, our training and analysis procedure was performed ten times, each using independent training and evaluation datasets. Our loss profiles and free energy estimates below report averages as well as statistical variation across these runs.

We first report training results in Fig. 4, where the full-batch loss is plotted as a function of the number of training steps. We

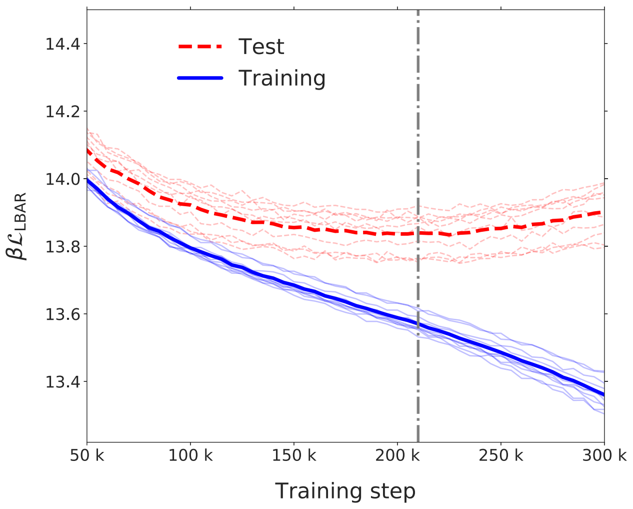

FIG. 4. Loss profile for bidirectional training. The bidirectional loss [Eq. (17)] is evaluated for training (solid lines) and test (dashed lines) datasets. Thin lines correspond to the individual runs, each trained on an independent dataset, and thick lines correspond to their respective averages. The loss keeps decreasing for the training set but exhibits a minimum for the test set at around 2.10 × 10^5 steps (vertical line). This is roughly the point where the model starts to overfit to the training data.

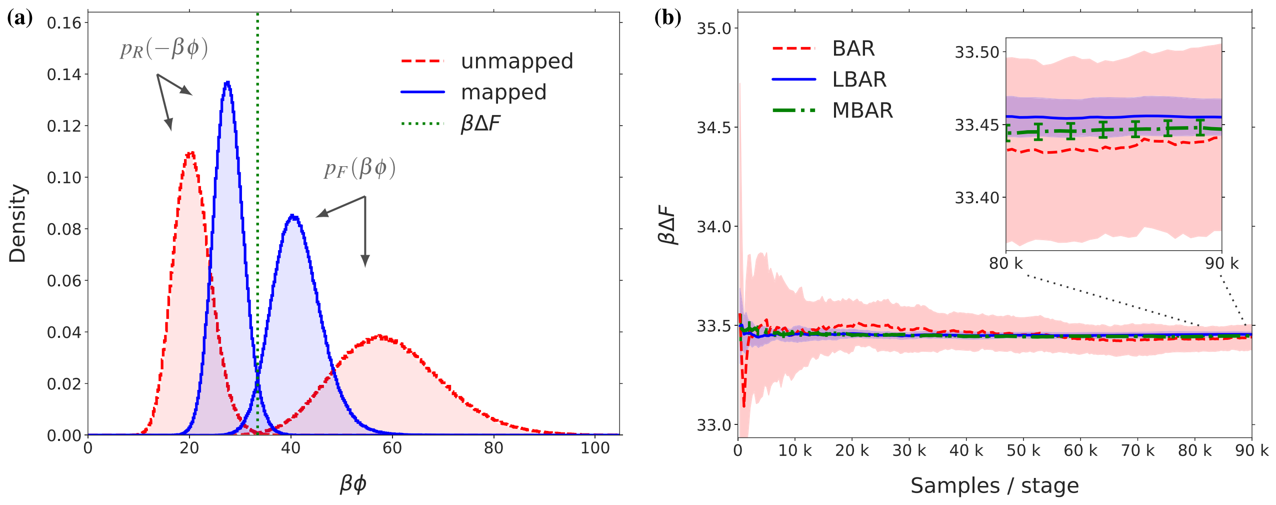

FIG. 5. Enhanced convergence of the learned LBAR estimator. (a) Normalized histograms of forward and reverse work ($\beta\phi$) values with (blue solid lines) and without (red dashed lines) the mapping. The vertical line indicates the ground-truth free energy estimate computed by MBAR. (b) Running averages of the ΔF estimate as a function of the number of evaluated samples per stage for the BAR (red dashed lines), LBAR (blue solid lines), and MBAR (green dashed-dotted lines) estimators. We note that the *x* axis shows the samples per stage; the MBAR estimate was computed using 15 stages (as opposed to two for BAR and LBAR), and thus, in total it uses 7.5 times more samples compared to the other two methods. The lines and shaded regions of BAR and LBAR estimates correspond to average and one standard deviation over the ten independent runs. The vertical green bars report the statistical error of the MBAR estimator.[7]

observe a pattern commonly encountered in ML: after an initial decrease of both training and test loss, the latter develops a minimum. At around the minimum, the model stops generalizing and starts to overfit to the training data. We, therefore, employ a technique called *early stopping*[46] and use the model parameters corresponding to the minimum test loss for all further evaluations. It is worth emphasizing, however, that the precise location of the minimum does not have an appreciable effect on the quality of the free energy estimates reported for the bidirectional estimator (results not presented). We also note the small variation among the independent runs, suggesting that there is no significant dependence of the performance on a particular dataset.

Next, we probe the overlap resulting from our learned mapping. In Fig. 5(a), we plot the distributions of learned work values for the forward and reverse directions compared against their unmapped counterparts for which the mapping is the identity. These distributions are typically unimodal[6] and satisfy the inequalities $-\mathbb{E}_B[\Phi_R] \leq \Delta F \leq \mathbb{E}_A[\Phi_F]$ for any invertible mapping. The free energy difference ΔF corresponds precisely to the point of intersection of the forward and reverse distributions. Both of these facts are a consequence of the fluctuation theorem [Eq. (8)]. Intuitively, it is clear then that an effective mapping should increase the overlap between the distributions to facilitate locating the intersection. This is precisely the enhancement we observe for the mapped work values. The two modes are shifted toward each other and share a significantly larger overlap than the unmapped distributions. We also see that the mapping strongly reduces the variance of the distributions. In fact, with a perfect mapping, we would expect the forward and reverse distributions to collapse onto a single delta distribution located at ΔF.

We now turn to the statistical convergence of the free energy estimates. In Fig. 5(b), we plot a running average of the estimate as a function of the number of evaluated samples per stage. The solid lines report averages of the estimate over the independent runs and shaded regions represent one standard deviation of the runs. We first validate the correctness of our method against a converged MBAR estimator. Here, MBAR employed 15 stages and, thus, 7.5 times more samples in total. From Fig. 5(b), we see that the variation of our estimate overlaps nicely with the MBAR error estimates. We next compare the efficiency of our method against the baseline BAR estimator, where we see in Fig. 5(b) a clear variance reduction of LBAR across a wide range of evaluated sample sizes. The full-batch LBAR standard deviation we observe is approximately 19% of that reported by BAR. Moreover, training and evaluation of the model occurred on the same dataset, demonstrating that an effective mapping can be learned in a *data-efficient* manner. This is an important practical consideration but not at all obvious *a priori*. We could, in principle, even combine samples from the training and test datasets for estimation of ΔF but have used the test set only to detect overfitting.

VII. DISCUSSION

In this work, we turn TFEP into a machine learning problem by combining it with state-of-the-art ML techniques. TFEP previously required hand-crafting a tractable mapping on configuration-space, a significant challenge for many realistic systems. We proposed to represent the mapping by a suitable neural network and identified training objectives for unidirectional and bidirectional cases.

We then tested their performance on a prototype solvation system – the growth of a soft sphere in a fluid of solvent particles in periodic boundary conditions. While this system is relatively simplistic from a physical standpoint, it poses a significant challenge for ML models due to the system's underlying permutational symmetry and periodic boundary conditions. Our experimental results indicate that both LFEP and LBAR estimators can lead to a significant variance reduction compared to their respective baselines and, therefore, clearly highlight the potential of this approach. Interesting directions for future work include a systematic analysis of the error of the learned estimators and a detailed comparison with MBAR. We believe it is possible that optimal estimation strategies on complex systems will contain a combination of staging and mapping.

Improving TFEP via learned mappings relates to the general idea of improving importance sampling by learning the proposal distribution, which has been explored substantially in machine learning and statistics. For instance, recent works in machine learning have proposed training a flexible deep-learning model of the proposal distribution to improve importance sampling[47] or more sophisticated variants such as bridge sampling[48] and sequential Monte Carlo.[49–51] In turn, these approaches can be traced back to methods for adaptive importance sampling[52] and adaptive sequential Monte Carlo[53] in statistics. One recent instance of these approaches that relates closely to our work is *Neural Importance Sampling*,[47] which uses expressive normalizing flows to learn good proposal distributions for importance sampling. Many of the above works have noted that the choice of loss function is important, with the forward KL divergence and chi-squared divergence being standard choices. These observations are in line with our observations of the differences between the unidirectional and bidirectional training losses.

We note that our learned free energy estimators and equivariant, periodic model architecture can be combined with the work on *Boltzmann Generators*,[28] which uses flow-based models in combination with statistical re-weighting to sample from desired Boltzmann distributions. In particular, it would be interesting to see how our targeted estimators compare to the ones considered in Ref. 28. Furthermore, we note that neural network based free energy estimation is an active field of research. For example, two recent studies have independently proposed targeted unidirectional[54] and bidirectional[55] estimators similar to the ones suggested here. Both studies employ autoregressive networks to compute free energy estimates of a lattice spin model with discrete states. In addition, Ref. 55 also estimates free energies of a small protein in the gas phase (no periodicity) using normalizing flows. As noted in that study, however, their model lacks permutation equivariance, which is likely a drawback when applying it to the type of solvation system we consider here. We also believe that permutation equivariant, periodicity-respecting networks, such as the one considered here, will be key to scaling up the approach to system sizes commonly used in atomistic simulations.

Finally, our results demonstrate that we can estimate free energy differences between two states directly and data efficiently, i.e., using fewer MD samples for training and evaluation than the base estimator would require to converge. Other studies, for example, Refs. 28 and 55, follow a different strategy and estimate free energy differences by learning two separate mappings that share a common reference state. We believe that our direct approach may be preferable in cases where one state is a small perturbation of the other. It will be interesting to see how these different approaches compare with respect to data efficiency and variance reduction, and under which circumstances one is preferable to the other.

AUTHORS' CONTRIBUTIONS

P.W. and A.J.B. contributed equally to this work.

ACKNOWLEDGMENTS

The authors would like to thank their colleagues Shakir Mohamed, Michael Figurnov, Ulrich Paquet, James Kirkpatrick, Daan Wierstra, Craig Donner, Alex Vitvitskyi, John Jumper, Amber Nicklin-Clark, Steph Hughes-Fitt, Alex Goldin, and Guy Scully for their help and for stimulating discussions.

APPENDIX A: ALTERNATE DERIVATION OF TFEP AND INTERPRETATION

In this section, we derive and interpret the unidirectional TFEP estimator as a multi-staged FEP estimator. This interpretation allows us to reason about TFEP using intuition from FEP.

Given explicit densities for A, A', and B, we can formally decompose ΔF into a sum over two terms,

$$\Delta F = \Delta F_{AA'} + \Delta F_{A'B}, \tag{A1}$$

which can each be computed separately using the FEP estimator [Eq. (1)]. We next define the energy of A' as

$$\begin{aligned} U_{A'}(\mathbf{x}) &= U_A\big(M^{-1}(\mathbf{x})\big) + \beta^{-1}\log|J_M(\mathbf{x})| \\ &= -\beta^{-1}\log[\rho_{A'}(\mathbf{x})Z_A], \end{aligned} \tag{A2}$$

such that $\rho_{A'} \propto e^{-\beta U_{A'}}$, where we have used Eq. (12) in going from the first to second line. Conveniently, one of these stages comes for free, as $\Delta F_{AA'} = 0$,

$$\begin{aligned} Z_{A'} &= \int \mathrm{d}\mathbf{x}\, e^{-\beta U_{A'}(\mathbf{x})} \\ &= Z_A \int \mathrm{d}\mathbf{x}\, \rho_{A'}(\mathbf{x}) \\ &= Z_A. \end{aligned} \tag{A3}$$

In going from the first to second line, we have used Eq. (A2). Combining Eqs. (A1)–(A3), as well as the FEP estimator [Eq. (1)], we arrive at our final result

$$\begin{aligned} e^{-\beta\Delta F} &= e^{-\beta\Delta F_{A'B}} \\ &= \mathbb{E}_{A'}\Big[e^{-\beta\Delta U'}\Big], \end{aligned} \tag{A4}$$

where

$$\begin{aligned} \Delta U'(\mathbf{x}) &= U_B(\mathbf{x}) - U_{A'}(\mathbf{x}) \\ &= U_B(\mathbf{x}) - U_A\big(M^{-1}(\mathbf{x})\big) - \beta^{-1}\log|J_M(\mathbf{x})| \end{aligned} \tag{A5}$$

is the energy difference between A' and B. Although Eq. (A4) is an explicit estimate between A' and B, it is just a reformulation of the

TFEP estimator [Eq. (7)] as can be seen by the equivalence of $\Delta U'$ and Φ_F [compare Eqs. (5) and (A5)]. This interpretation of TFEP allows us to apply the intuition on convergence we have built for FEP. Specifically, we can accelerate convergence if A' shares a large overlap with B.

APPENDIX B: SYSTEM

To generate the training data, we performed MD simulations of the system illustrated in Fig. 2 using the simulation package LAMMPS.[38] The system is similar to the one studied in Ref. 14 but we replaced hard solute–solvent interactions by a Weeks–Chandler–Andersen[36] (WCA) potential. Below, we represent our energy, length, and mass units in terms of the LJ well depth ε, the LJ diameter σ, and the solvent particle mass m, respectively.[37] From this, our unit of time is defined as $\tau = \sigma\sqrt{m/\varepsilon}$. Quantities expressed in these reduced units are denoted by an asterisk. We used a cubic simulation box with an edge length of $2L^* = 6.29$ and employed cutoff radii of L^* and $\sqrt[6]{2}R_\alpha$ for LJ and WCA interactions, where $\alpha \in \{A, B\}$ labels the state. The solute radii were taken to be $R_A^* = 2.5974$ and $R_B^* = 2.8444$. Both LJ and WCA potentials shared the same value for ε, and we set $\sigma_{\text{WCA}} = R_\alpha$.

To simulate a specific state, we first assigned random positions to all solvent particles. The solute was placed at the origin of the box and kept stationary throughout the simulation. After performing energy minimization, the system was equilibrated in the canonical ensemble for a period of $5 \times 10^4\tau$. The equations of motion were integrated using the velocity Verlet algorithm[56] with a time step of 0.002τ. We employed a Langevin thermostat with a relaxation time of 0.5τ to keep the system at a temperature of $T^* = 3.2155$. To prevent drift of the center of mass motion, we set the total random force to zero. During the $5 \times 10^5\tau$ long production run, configurations were sampled every ten reduced time units, yielding a total of 5×10^4 samples. For each state, 20 such simulations were generated starting from random initializations and random number seeds. The resulting 1×10^6 samples were then partitioned as described in the main text.

We followed the same protocol to generate samples for MBAR. In addition to the two states corresponding to R_A and R_B, we considered 13 intermediate states with a constant radial increment $\Delta R_{i,i+1}$. Using two different random seeds, we obtained 1×10^5 samples for each state and evaluated all 15 energy functions on each sample. MBAR estimates of ΔF were then computed from this combined energy matrix using the package pymbar.[7]

APPENDIX C: MODEL IMPLEMENTATION DETAILS

We implement G using *circular splines*[42] so that G satisfies the boundary conditions in Eqs. (25) and (26). Briefly, G is a piecewise function that consists of S segments. The parameters ψ are a set of $S + 1$ triplets (x_s, y_s, d_s), where $[x_{s-1}, x_s]$ defines the domain of segment s, $[y_{s-1}, y_s]$ defines its image, and d_{s-1}, d_s define its slopes at the endpoints (all slopes are required to be positive). Each segment is a strictly increasing rational-quadratic function, constructed using the interpolation method of Gregory and Delbourgo.[57] The conditions in Eqs. (25) and (26) are satisfied by setting $x_0 = y_0 = -L$, $x_S = y_S = L$, and $d_0 = d_S > 0$. We can increase the flexibility of G by increasing the number of segments S. With a large enough S, circular splines can approximate arbitrarily well any strictly increasing function from $[-L, L]$ to itself that satisfies Eqs. (25) and (26).

We implement C using the *transformer architecture* of Vaswani *et al.*[41] so that C satisfies the permutation-equivariance condition in Eq. (28). Briefly, the network architecture is as follows. Each input $r_i^{\mathbb{I}_k}$ undergoes the feature transformation in Eq. (27) and is then mapped to a fixed-sized vector h_i via an affine transformation identically for each i. Then, $(h_1, \ldots, h_N)$ is processed by a sequence of *transformer blocks*, each of which is composed of two *residual layers*.[18] The first residual layer is

$$h_i \leftarrow h_i + \text{MHA}_i(h_1, \ldots, h_N), \tag{C1}$$

where MHA is a *multi-headed self-attention layer* as described by Vaswani *et al.*,[41] and MHA_i is its ith output. The second residual layer is

$$h_i \leftarrow h_i + \text{MLP}(h_i), \tag{C2}$$

where MLP is a standard *multi-layer perceptron*[58] that is applied identically for each i. After repeating the transformer block a specified number of times (each time with different learnable parameters), each vector h_i is mapped to the output $(\psi_i^\nu)_{\nu\notin\mathbb{I}_k}$ via an affine transformation identically for each i. To help with generalization and stability, we applied *layer normalization*[59] to the inputs of MHA and MLP, as well as to the output. Because MHA is permutation-equivariant and every MLP and affine transformation is applied identically for each i, it follows that C is permutation-equivariant, and therefore, the transformed distribution is permutation-invariant as desired.

All models were trained using the Adam optimizer.[60] A summary of the hyperparameters is provided in Table I.

TABLE I. Model and training hyperparameters.

Transformer	
Number of blocks	4
Number of heads	4
Value dimension	128
Key dimension	128
Embedding dimension	128
Circular spline flow	
Number of segments (S)	32
Adam optimizer	
Batch size	256
Learning rate	10^{-4}
β_1	0.9
β_2	0.999

APPENDIX D: RESULTS FOR LFEP

In this section, we discuss the free energy estimates obtained with LFEP, using unidirectional training with the $\mathscr{L}_{\text{LFEP}}$ loss [Eq. (16)] as a proxy for D_{KL} [Eq. (14)]. To estimate the KL, we

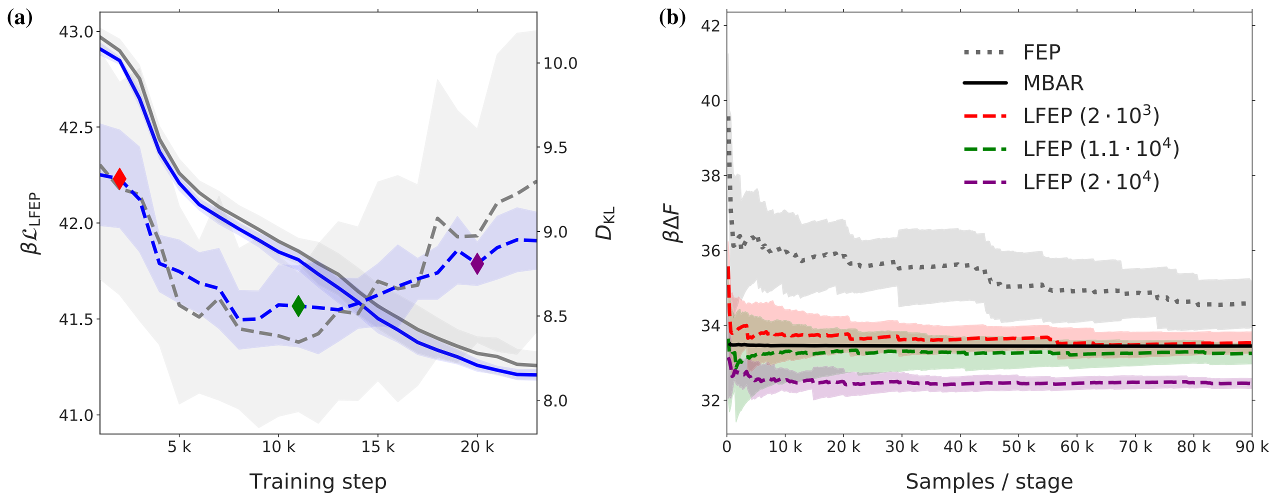

FIG. 6. Enhanced convergence of the learned LFEP estimator. (a) Estimates of the loss $\mathscr{L}_{\mathrm{LFEP}}$ (solid lines) and D_{KL} (dashed lines), averaged across ten different datasets, as a function of training progress for test (gray) and training (blue) datasets. Shaded regions correspond to one standard deviation estimated across the independent runs. Colored diamonds highlight the selected training steps for the evaluation of ΔF. (b) Running averages of the ΔF estimate as a function of the number of evaluated samples per stage for the FEP (dotted line), MBAR (solid line), and LFEP (dashed lines) estimators. For the latter, the numbers within parentheses correspond to the training steps at which we evaluated the mapping. The lines and shaded regions of FEP and LFEP estimates correspond to the average and one standard deviation over the runs, respectively.

replaced ΔF in Eq. (14) with the LFEP estimate of that quantity. Figure 6(a) shows the evolution of both quantities during training. The results feature an interesting behavior in the initial training regime. While test and training loss are still decreasing, the KL already exhibits a pronounced minimum due to a drift of the ΔF estimate toward lower values. This is further illustrated in Fig. 6(b), which compares the convergence of ΔF for three different mappings corresponding to specific training steps [colored symbols in Fig. 6(a)]. We see that the variance is reduced significantly in all three cases. However, only the first mapping (training step 2×10^3) agrees with the MBAR baseline, while we can already observe a small bias for the second mapping (training step 1.10×10^4) that becomes even more pronounced as the training progresses (training step 2×10^4). We note that this behavior is consistent with other experiments that we performed in the unidirectional setting (data not shown).

This is problematic in that the minimum of the KL would be a natural point to stop training but does not yield the lowest bias. This is also in contrast to our findings for the bidirectional case, where the quality of the mapping was best in the vicinity of the minimum in the test loss. One possible explanation for these observations is the well-known zero-forcing property[33] of $\mathscr{L}_{\mathrm{LFEP}}$. That is, $\mathscr{L}_{\mathrm{LFEP}}$ does not encourage the transformed distribution to be mass-covering, which is known to negatively impact the performance of importance-sampling estimators.[34]

DATA AVAILABILITY

The data that support the findings of this study are openly available at https://github.com/deepmind/deepmind-research/tree/master/learned_free_energy_estimation.

REFERENCES

[1] M. R. Shirts, D. L. Mobley, and S. P. Brown, "Free-energy calculations in structure-based drug design," in *Drug Design: Structure- and Ligand-Based Approaches*, edited by K. M. Merz, Jr., D. Ringe, and C. H. Reynolds (Cambridge University Press, 2010), pp. 61–86.
[2] S. Auer and D. Frenkel, Nature **409**, 1020 (2001).
[3] P. F. Damasceno, M. Engel, and S. C. Glotzer, Science **337**, 453 (2012).
[4] T. Curk, P. Wirnsberger, J. Dobnikar, D. Frenkel, and A. Šarić, Nano Lett. **18**, 5350 (2018).
[5] K. Hauser, C. Negron, S. K. Albanese, S. Ray, T. Steinbrecher, R. Abel, J. D. Chodera, and L. Wang, Commun. Biol. **1**, 70 (2018).
[6] *Free Energy Calculations: Theory and Applications in Chemistry and Biology*, Springer Series in Chemical Physics Vol. 86, edited by C. Chipot and A. Pohorille (Springer-Verlag, Berlin, 2007).
[7] M. R. Shirts and J. D. Chodera, J. Chem. Phys. **129**, 124105 (2008).
[8] R. W. Zwanzig, J. Chem. Phys. **22**, 1420 (1954).
[9] R. M. Neal, Stat. Comput. **11**, 1573 (2001).
[10] D. Arnaud, N. de Freitas, and N. Gordon, *Sequential Monte Carlo Methods in Practice*, Information Science and Statistics (Springer, New York, 2001).
[11] J. S. Liu, *Monte Carlo Strategies in Scientific Computing* (Springer Science & Business Media, 2008).
[12] A. Pohorille, C. Jarzynski, and C. Chipot, J. Phys. Chem. B **114**, 10235 (2010).
[13] C. Jarzynski, Phys. Rev. E **73**, 046105 (2006).
[14] C. Jarzynski, Phys. Rev. E **65**, 046122 (2002).
[15] A. M. Hahn and H. Then, Phys. Rev. E **79**, 011113 (2009).
[16] C. H. Bennett, J. Comput. Phys. **22**, 245 (1976).
[17] A. Krizhevsky, I. Sutskever, and G. E. Hinton, *Advances in Neural Information Processing Systems* (Curran Associates, Inc, 2012).
[18] K. He, X. Zhang, S. Ren, and J. Sun, in *IEEE Conference on Computer Vision and Pattern Recognition* (IEEE, 2016).
[19] V. Mnih, K. Kavukcuoglu, D. Silver, A. A. Rusu, J. Veness, M. G. Bellemare, A. Graves, M. Riedmiller, A. K. Fidjeland, G. Ostrovski, S. Petersen, C. Beattie, A. Sadik, I. Antonoglou, H. King, D. Kumaran, D. Wierstra, S. Legg, and D. Hassabis, Nature **518**, 529 (2015).

[20]D. Silver, A. Huang, C. J. Maddison, A. Guez, L. Sifre, G. van den Driessche, J. Schrittwieser, I. Antonoglou, V. Panneershelvam, M. Lanctot, S. Dieleman, D. Grewe, J. Nham, N. Kalchbrenner, I. Sutskever, T. Lillicrap, M. Leach, K. Kavukcuoglu, T. Graepel, and D. Hassabis, Nature **529**, 484 (2016).
[21]D. Silver, T. Hubert, J. Schrittwieser, I. Antonoglou, M. Lai, A. Guez, M. Lanctot, L. Sifre, D. Kumaran, T. Graepel, T. Lillicrap, K. Simonyan, and D. Hassabis, Science **362**, 1140 (2018).
[22]A. Brock, J. Donahue, and K. Simonyan, in International Conference on Learning Representations, 2019.
[23]T. Karras, S. Laine, and T. Aila, in *IEEE Conference on Computer Vision and Pattern Recognition* (IEEE, 2019).
[24]A. W. Senior, R. Evans, J. Jumper, J. Kirkpatrick, L. Sifre, T. Green, C. Qin, A. Žídek, A. W. R. Nelson, A. Bridgland, H. Penedones, S. Petersen, K. Simonyan, S. Crossan, P. Kohli, D. T. Jones, D. Silver, K. Kavukcuoglu, and D. Hassabis, Nature **577**, 706 (2020).
[25]T. Morawietz, A. Singraber, C. Dellago, and J. Behler, Proc. Natl. Acad. Sci. U. S. A. **113**, 8368 (2016).
[26]L. Zhang, J. Han, H. Wang, R. Car, and W. E, Phys. Rev. Lett. **120**, 143001 (2018).
[27]M. S. Albergo, G. Kanwar, and P. E. Shanahan, Phys. Rev. D **100**, 034515 (2019).
[28]F. Noé, S. Olsson, J. Köhler, and H. Wu, Science **365**, eaaw1147 (2019).
[29]D. Wu, L. Wang, and P. Zhang, Phys. Rev. Lett. **122**, 080602 (2019).
[30]S.-H. Li and L. Wang, Phys. Rev. Lett. **121**, 260601 (2018).
[31]M. R. Shirts, E. Bair, G. Hooker, and V. S. Pande, Phys. Rev. Lett. **91**, 140601 (2003).
[32]P. Maragakis, M. Spichty, and M. Karplus, Phys. Rev. Lett. **96**, 100602 (2006).
[33]T. Minka, "Divergence measures and message passing," Technical Report MSR-TR-2005-173, Microsoft Research Cambridge, 2005.
[34]R. M. Neal, "Estimating ratios of normalizing constants using linked importance sampling," Technical Report 0511, Department of Statistics, University of Toronto, 2005.
[35]J. E. Jones and S. Chapman, Proc. R. Soc. London, Ser. A **106**, 463 (1924).
[36]J. D. Weeks and D. Chandler, J. Chem. Phys. **54**, 5237 (1971).
[37]D. Frenkel and B. Smit, *Understanding Molecular Simulation*, 2nd ed. (Academic Press, San Diego, 2002).
[38]S. Plimpton, J. Comput. Phys. **117**, 1 (1995).
[39]G. Papamakarios, E. Nalisnick, D. Jimenez Rezende, S. Mohamed, and B. Lakshminarayanan, arXiv:1912.02762 (2019).
[40]L. Dinh, J. Sohl-Dickstein, and S. Bengio, in International Conference on Learning Representations, 2017.
[41]A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, Ł. Kaiser, and I. Polosukhin, *Advances in Neural Information Processing Systems* (Curran Associates, Inc, 2017).
[42]D. Jimenez Rezende, G. Papamakarios, S. Racanière, M. S. Albergo, G. Kanwar, P. E. Shanahan, and K. Cranmer, arXiv:2002.02428 (2020).
[43]C. Durkan, A. Bekasov, I. Murray, and G. Papamakarios, *Advances in Neural Information Processing Systems* (Curran Associates, Inc, 2019).
[44]C. Bender, K. O'Connor, Y. Li, J. J. Garcia, M. Zaheer, and J. Oliva, in AAAI Conference on Artificial Intelligence, 2020.
[45]J. Köhler, L. Klein, and F. Noé, arXiv:1910.00753 (2019).
[46]R. Caruana, S. Lawrence, and L. Giles, *Advances in Neural Information Processing Systems* (MIT Press, 2001).
[47]T. Müller, B. McWilliams, F. Rousselle, M. Gross, and J. Novák, ACM Trans. Graphics **38**, 145 (2019).
[48]G. Papamakarios and I. Murray, in Probabilistic Integration Workshop at Advances in Neural Information Processing Systems, 2015.
[49]S. Gu, Z. Ghahramani, and R. E. Turner, *Advances in Neural Information Processing Systems* (Curran Associates, Inc, 2015).
[50]B. Paige and F. Wood, in International Conference on Machine Learning, 2016.
[51]T. A. Le, A. G. Baydin, and F. Wood, in International Conference on Artificial Intelligence and Statistics, 2017.
[52]O. Cappé, R. Douc, A. Guillin, J.-M. Marin, and C. P. Robert, Stat. Comput. **18**, 447 (2008).
[53]J. Cornebise, É. Moulines, and J. Olsson, Stat. Comput. **18**, 461 (2008).
[54]K. A. Nicoli, S. Nakajima, N. Strodthoff, W. Samek, K.-R. Müller, and P. Kessel, Phys. Rev. E **101**, 023304 (2020).
[55]X. Ding and B. Zhang, arXiv:2005.00638 (2019).
[56]W. C. Swope, H. C. Andersen, P. H. Berens, and K. R. Wilson, J. Chem. Phys. **76**, 637 (1982).
[57]J. A. Gregory and R. Delbourgo, IMA J. Numer. Anal. **2**, 123 (1982).
[58]I. Goodfellow, Y. Bengio, and A. Courville, *Deep Learning* (MIT Press, 2016).
[59]J. L. Ba, J. R. Kiros, and G. E. Hinton, arXiv:1607.06450 (2016).
[60]D. P. Kingma and J. Ba, in International Conference on Learning Representations, 2015.