Tangent Space Least Adaptive Clustering

James Buenfil [1] **Samson Koelle** [1] **Marina Meila** [1]

Abstract

The biasing of dynamical simulations along collective variables uncovered by unsupervised learning has become a standard approach in analysis of molecular systems. However, despite parallels with reinforcement learning (RL), state of the art RL methods have yet to reach the molecular dynamics community. The interaction between unsupervised learning, dynamical simulations, and RL is therefore a promising area of research. We introduce a method for enhanced sampling that uses non-linear geometry estimated by an unsupervised learning algorithm in a reinforcement-learning enhanced sampler. We give theoretical background justifying this method, and show results on data.

1. Introduction

Molecular dynamics (MD) simulations are an empirical method for understanding the properties of materials from proteins to solar panels (15, 22, 9, 11). In these simulations, positions of individual atoms or sub-molecular agglomerations are sampled through time according to interatomic forces, but despite knowledge of these forces, the overall configurational behavior is unknown. Simulations thus provide information about the low-dimensional dynamics and properties of the simulated system (16). Unfortunately, the computational burden of these simulations is high (25). The potential energy and its gradient are often challenging to compute, and low-probability but important regions like transitions between energy wells require huge numbers of samples to access. Satisfactory exploration of chemical compound behavior in simulation is therefore a challenging and unresolved problem.

A variety of machine learning methods have been proposed to reduce this computational cost (33, 20). For example, potentials generated using supervised learning have been used to sample from hard-to-compute (6) or hard-to-find (26) configurations. Unsupervised learning is also used to assist simulations through discovery of *collective variables* (CVs) that parametrize the *slow manifold* generated by the underlying dynamics (24, 21, 10, 28). Once this data manifold is learned, it can be used to bias future similar simulations towards interesting high-energy or transitional regions. Many methods for biasing simulations based on CVs have been proposed (31, 19, 5).

There are many similarities between reinforcement learning and this collective-variable based sampling paradigm. Simulators balance exploration versus exploitation, and can use tools like differentiable deep networks to identify good actions for progressing the simulation to interesting states (3, 23). The variation of experimental conditions like solvents or temperatures to create a more robust understanding of molecular behavior resembles domain randomization (32). As in simulated robotics (34), a complicated heirarchy of simulation fidelity relates the simulated system to the real world phenomena of ultimate interest (12, 30). Perhaps due to the relatively recent success of reinforcement learning methods, formal application of reinforcement learning principals is still limited. However, it is reasonable to expect that application of these principals should integrate with existing unsupervised-learning based approaches to enhanced sampling using CVs.

One approach that does perform enhanced sampling through a combination of CVs and RL is the REinforcement learning based Adaptive samPling (REAP) method of (27). However, this approach does not use any non-trivial unsupervised learning. Our idea therefore is to adapt this approach from the perspective of non-linear geometry. We therefore introduce Tangent Space Least Adaptive Clustering (TSLC) that favors exploration of variables that parametrize the data manifold. We use local tangent space estimation as a simple and straightforward method for learning the manifold, introduce a new reward based on projection onto tangent spaces, and give an efficient clustering algorithm for determination of sets on which to compute the tangent spaces.

2. Background

We motivate this method in the context of unsupervised learning for dynamical systems.

[1] Department of Statistics, University of Washington, Seattle, WA, USA. Correspondence to: Marina Meila <mmp2@uw.edu>.

Proceedings of the 38^{th} International Conference on Machine Learning, PMLR 139, 2021. Copyright 2021 by the author(s).

Dynamical systems Dynamical models allow simulators to progress forward in time t through computation of the derivative of the state vector $x(t) \in \mathbb{R}^D$ w.r.t. t. A simple model of dynamics is given by the Langevin Equation (27)

$$\frac{d}{dt}x(t) = \frac{1}{\gamma}\nabla E(x(t)) + \eta(t)\sqrt{Tk_B\gamma} \tag{1}$$

where is $E(x(t))$ is the potential energy, $\eta(t)$ is a random force and γ is a constant that, together with the temperature T and Boltzmann constant k_B, determines the degree of randomness of the random walk through the configuration space. Langevin dynamics for even small systems are chaotic, and even though we can compute E and its derivatives, we do not in general know the properties of the slow-manifold around which the random walk progresses (1). Additionally, there is no guarantee that this walk will traverse the slow-manifold in a reasonable amount of time. Therefore, these dynamics are a motivating application of unsupervised learning for acceleration of dynamical simulations (17, 2).

Unsupervised learning and tangent spaces Suppose, in the ideal case, that we are able to simulate from the Boltzmann distribution until the density of $x(t)$ is exactly proportional to E. Theoretical analyses often guarantees that these data are distributed on a manifold M containing a central slow-manifold S within the D dimensional feature space (13). Unsupervised learning then estimates a map ϕ such that $\phi(M) = \phi|_M(S)$ is 1) *diffeomorphic* to S and 2) in a low-dimensional space $\mathbb{R}^m$ $(m << D)$.

Given such a ϕ, the typical approaches for accelerating simulation are 1) to find a geometric feature of the configuration parameterizing $\phi(S)$ and move the simulation in its direction (25), or 2) if ϕ is differentiable, to move the simulation in a direction $\nabla\phi^k$ where ϕ^k is a coordinate of the learned latent space (5). In either case, the geometric feature or coordinate functions ϕ^k are a CV that parameterize S.

One issue with the above is that we may not have enough data to learn a good ϕ. In this case, tangent space estimation is a less data-intensive alternative to learning ϕ. The reason is that, if ϕ is suitably well behaved, then the singular value decomposition of its differential in coordinates of $\mathbb{R}^D$ and $\mathbb{R}^m$ gives $D\phi(x) = (T_xS)\Sigma_x(T_{\phi(x)}\phi(S))^T$ (18), where T_xS is the tangent space of the slow-manifold. Biasing simulations by the method of collective variables is essentially moving along the tangent space T_xS. In our approach, we will substitute in a tangent space estimator for T_xS without explicit calculation of ϕ.

REinforcement learning based Adaptive samPling REinforcement learning based Adaptive samPling (REAP) is an application of RL principals to accelerating traversal of configuration spaces in dynamical simulations (27). It is an extension of least counts adaptive sampling (LC), a simple iterative paradigm for enhanced sampling that uses clustering to identify undersampled regions. Our proposed method for sampling is largely motivated by REAP.

In LC, every iteration consists of simulating dynamics from given starting states, clustering the trajectory, and then selecting the next starting states from the smallest, or least count, clusters. Clusters with fewer members correspond to more sparsely sampled regions of the configuration space, and will tend to be towards the boundaries of the explored region.

REAP also proceeds iteratively by simulating, clustering, and selecting new initial states to start sampling from, but the way the next initial states are chosen from the least count clusters is more involved. REAP supposes preselection of a set of possible functions of the state, $\theta_1(x)\ldots\theta_K(x)$: $\mathbb{R}^D \to \mathbb{R}$, also known as collective variables. With no a priori knowledge of S, this set will be larger than ideal, and the REAP algorithm must decide their relative importance on-the-fly. Every iteration, after clustering to find least count clusters, a weight for each CV is recomputed. These weights $w \in \mathbb{R}^K$ are determined by maximizing

$$R = \sum_{i=1}^{K} w_i \sum_{m=1}^{|C_p|} \frac{|\theta_i(s_m) - \langle\theta_i(T)\rangle|}{\sigma_i(T)} \tag{2}$$

where C_p is the set of states in the least count clusters, $\theta_i(s_m)$ is the value of the ith CV at state $s_m \in C_p$, and $\langle\theta_i(T)\rangle$ and $\sigma_i(T)$ are the mean and standard deviation of θ_i over the full trajectory T thus far, respectively. The w_i are constrained to sum to one, be nonnegative, and each w_i may not change by more than a parameter δ from its value in the previous round. These weights are then used to select initial states for the next round of simulation, namely a fixed number of states in C_p which maximize

$$r(s_m) = \sum_{i=1}^{K} w_i \frac{|\theta_i(s_m) - \langle\theta_i(C_p)\rangle|}{\sigma_i(C_p)} \tag{3}$$

Comparison with the rich universe of RL methods indicates that there are many areas of potential improvement to this approach (8). We focus in particular on adapting Equation 2 with the geometrical underpinnings of the method of collective variables. The REAP objective only considers population statistics like mean and standard deviation - much less information than can be captured through unsupervised learning (29).

3. Methods

We thereby propose a method for determining weights for CVs which differs from REAP in two key ways. First, we select weights which capture global geometry of the underlying data manifold from only local geometry. Second, we replace k-means clustering with our own clustering method CLUST, which is amenable to the sampling problem. Our method, Tangent Space Least Adaptive Clustering (TSLC), like REAP, consists of timestepping, clustering, reweighting, and reinitialization. CVs are known, and weights calculated for each CV at each iteration are used to select points from least count clusters. However, unlike REAP, we assume, as is typical in unsupervised learning for dynamical systems, that the explored potential energy surface lies near a low-dimensional manifold.

Algorithm 1 Tangent Space Least Adaptive Clustering

```
Input: Differentiable energy $E$, collective variables $\theta_1 \dots \theta_K$, number of least count clusters to sample from $N$, number of parallel runs $M$, number of iterations $I$, timesteps per iteration $s$, intrinsic dimension $d$
for $t \in 1 \dots I$ do
    Compute clusters, centers, and least count clusters: $\mathcal{C}, q, C_p \leftarrow \text{CLUST}(x(1 : tsM)), N)$
    Compute $V_i \leftarrow PCA(x(C_i)), d)$ for $i \in 1 \dots |\mathcal{C}|$
    Compute $\nabla\theta_k(q_i)$ for $i \in 1 \dots |\mathcal{C}|$, $k \in 1 \dots K$
    Normalize $\nabla\theta_k(q_i) \leftarrow \frac{n\nabla\theta_k(q_i)}{\sum_{i=1}^{|\mathcal{C}|} ||\nabla\theta_k(q_i)||}$ for $i \in 1 \dots |\mathcal{C}|, k \in 1 \dots K$
    Compute $A$ as in Equation (5)
    Compute $\tilde{w}$, the largest eigenvector of $A$
    Compute $w_k = \tilde{w}_k^2$ for $k \in 1 \dots K$
    Select $S_M$, the $M$ states from $C_p$ with the highest values of Equation (3)
    For each state $s_m$ in $S_M$, timestep Equation (1) $s$ times starting from $s_m$
end for
```

Algorithm description There are three important observations in the development of our choice of weights. One is that the collective variables which characterize the potential and encourage exploration should have gradients which are parallel to the tangent spaces of S. In particular, the projection of gradients onto local tangent spaces should be large. Two is that local tangent spaces can be computed on each cluster, using Principal Component Analysis. Since our data are distributed near S, we can estimate its tangent spaces as long as our clusters are small enough to reflect local information about the manifold, while also large enough to not be dominated by small-scale fluctuations. These requirements on the clusters for our purpose are what motivate our clustering algorithm and final observation, that the volume of the clusters chosen should scale with the volume explored. Since TSLC only differs from REAP in the computation of weights and choice of clustering, we only explain those steps in detail.

Weight Computation (1) We compute a clustering of the full trajectory thus far $\text{CLUST}(x_{full}) = \mathcal{C}$. We calculate for each cluster i a d dimensional local tangent space using standard Principal Component Analysis. Denote a suitable choice of d D-dimensional orthogonal vectors which span this space as columns of the matrix $V_i \in \mathbb{R}^{D\times d}$.
(2) The center of the cluster i, q_i, is used as a representative of that cluster. We compute the values of $\nabla\theta_j(q_i)$ for each CV θ_j on each cluster i, which we can can calculate with automatic differentiation since the CVs are analytic. In order to avoid favoring CVs that are scaled larger, we normalize the gradients over the clusters (see Algorithm 1). Let $G_i \in \mathbb{R}^{D\times K}$ be a matrix so that the j-th column is the normalized gradient $\nabla\theta_j(q_i)$.
(3) To find the weights, we solve the following optimization problem: find the unit 2-norm column vector $w \in \mathbb{R}^K$ so that

$$R = \sum_{i=1}^{n} ||V_i V_i^T G_i w||_2^2 \tag{4}$$

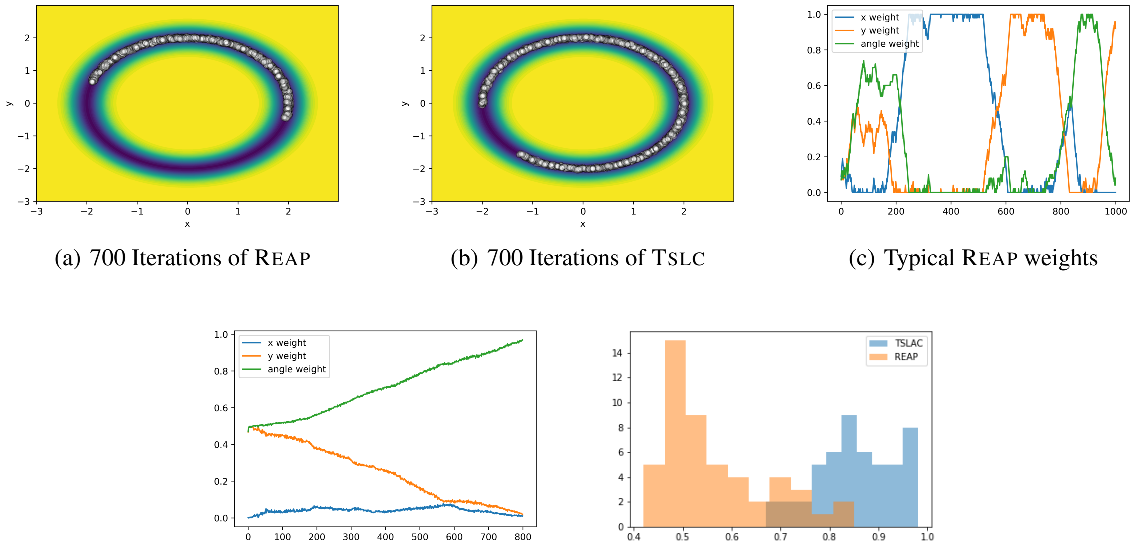
is maximized. This selects a linear combination of the gradients of the CVs that on average makes the projection onto the local tangent spaces largest. Since the columns of each V_i are orthogonal, $||V_i V_i^T G_i w||_2 = ||V_i^T G_i w||_2$. It follows that this objective is

$$R = w^T \left(\sum_{i=1}^{n} G_i^T V_i V_i^T G_i \right) w \equiv w^T A w, \tag{5}$$

to which the solution is the eigenvector of $A \in \mathbb{R}^{K\times K}$ associated with the maximum eigenvalue of A, which exists because A is symmetric. Finally, let the weights w_i be the squares of the entries of w. These are nonnegative and sum to one. The states to start sampling from in the next iteration are the M states in C_p which maximize Equation (3).

Clustering $\text{CLUST}(x, N)$ takes as input x_{full}, the full trajectory so far and N, the number of least count clusters we want, and returns the clustering, centers of each cluster, and least count clusters. It is inspired by the initialization method of (7), extended to K-means by (4), and simplified to take into account the need for speed and for small clusters.

In TSLC, we replace k-means clustering with a clustering procedure that recomputes the number of clusters n_{clu} to use at every iteration. We calculate the number of clusters starting from the following premise. In a uniform random walk, the distance from the start of the data grows asymptotically as $\sqrt{n}$, where n is the total number of sampled points. We expect our sampling to explore faster than a random walk, and the distance to grow as n^γ for a parameter $\gamma \in (1/2, 1)$. Hence, the volume explored will grow as $(n^\gamma)^d$ and to maintain clusters of approximately



(a) 700 Iterations of REAP

(b) 700 Iterations of TSLC

(c) Typical REAP weights

(d) Typical TSLC Weights

(e) REAP and TSLC Counts of Percentage of Circle Explored

equal volume, the number of centers should grow proportionally with this number. Therefore n_{clu} is taken to be $floor(b(n^{\gamma})^{d})$, where b is a parameter. Then, a number of centers $n'_{clu} = floor(n_{clu} \log n_{clu}) + l$ are sampled uniformly from x_{full} where l is a parameter. To these n'_{clu} centers, we apply the Fastest First Traversal algorithm (14), pruning the centers until only n_{clu} are left. Finally, we perform a single step of k-means with these initialized centers to assign points to clusters and to recalculate the centers. The result is n_{clu} clusters which retain approximately the same volume every iteration of TSLC and whose centers are spread uniformly over the explored region.

4. Experiments

Comparison of TSLC with REAP We compare TSLC with REAP on a toy circular potential function embedded in $D = 10$ dimensions. The potential is lowest on the circle(in the x_1-x_2 plane) and its magnitude decays exponentially with the square of the distance away from the circle. It is given by $V = -c * exp(-a * dist^2)$, where $a = 10$, $c = 250$, and $dist$ is the distance from the circle in Euclidean space. Both methods are given the same CVs: the coordinate projections functions of the state x, $\theta_i = x_i$ for $i = 1...D$, and $\theta_0 = arctan(x_2/x_1)$, the angle on the circle. The definition of θ_0 must be adjusted so that it is correctly defined on every quadrant of the x_1-x_2 plane.

To compare, we run both sampling methods $I = 700$. iterations, and look at the percentage of the circle explored. REAP was implemented using code provided by (27), available online. Timestepping was done the same way as in (27). Both timestep from two initial states $s = 5$ times each every iteration, and sample $M = 2$ new points from the $N = 2$ least count clusters found by their respective clustering algorithms. REAP uses $\delta = .02$ and $k = 20$ for k-means for all iterations, while ours uses parameters $b = .07$, $l = 2$ and $\gamma = .7$. We repeat this procedure 50 times each and plot histograms of the percentages of the circle explored in Figure 1(e). On average, REAP explored approximately 56% of the circle in the given time, while TSLC explored 85%, indicating strong performance of TSLC. In Figure 1(a) we show what one round of 700 iterations of REAP typically looks like using the same methodology as above. The plot shown is a contour plot of the potential V only as a function of x_1 and x_2, with the other 8 state variables set to 0. In 1(b) we do the same for TSLC. In Figures 1(c) and 1(d) we plot weights of θ_1, θ_2, and θ_0 that REAP and TSLC select over typical runs, as a function of iteration number. TSLC is able to identify θ_0 as the CV which characterizes the potential, and gradually becomes more confident in this, while REAP tends to select one variable at a time which it believes is the most important.

5. Conclusion

We have described a novel reinforcement learning-based approach for simulating from dynamical systems. This method is inspired by unsupervised learning methods currently used in molecular dynamics, and is a step towards a synthesis of unsupervised-learning CV based methods with RL.

6. Acknowledgements

This material is based upon work supported by the National Science Foundation under Grant DMS-1810975, the ARCS Foundation Award, and the GO-MAP Graduate Excellence Award.

References

[1] Christian Beck. Dynamical systems of langevin type. *Physica A: Statistical Mechanics and its Applications*, 233(1):419–440, November 1996.

[2] Adam Block, Youssef Mroueh, Alexander Rakhlin, and Jerret Ross. Fast mixing of Multi-Scale langevin dynamics under the manifold hypothesis. June 2020.

[3] Luigi Bonati, Yue-Yu Zhang, and Michele Parrinello. Neural networks-based variationally enhanced sampling. *Proc. Natl. Acad. Sci. U. S. A.*, 116(36):17641–17647, September 2019.

[4] Sebastien Bubeck, Marina Meilă, and Ulrike von Luxburg. How the initialization affects the stability of the k-means algorithm. *ESAIM: Probability and Statistics*.

[5] Wei Chen and Andrew L Ferguson. Molecular enhanced sampling with autoencoders: On-the-fly collective variable discovery and accelerated free energy landscape exploration. *J. Comput. Chem.*, 39(25): 2079–2102, September 2018.

[6] Stefan Chmiela, Alexandre Tkatchenko, Huziel E Sauceda, Igor Poltavsky, Kristof T Schütt, and Klaus-Robert Müller. Machine learning of accurate energy-conserving molecular force fields. *Sci Adv*, 3(5): e1603015, May 2017.

[7] Sanjoy Dasgupta and Leonard Schulman. A probabilistic analysis of em for mixtures of separated, spherical gaussians. *Journal of Machine Learnig Research*, 8: 203–226, Feb 2007.

[8] Vincent Francois-Lavet, Peter Henderson, Riashat Islam, Marc G Bellemare, and Joelle Pineau. An introduction to deep reinforcement learning. November 2018.

[9] Christopher D Fu, Luiz F L Oliveira, and Jim Pfaendtner. Determining energy barriers and selectivities of a multi-pathway system with infrequent metadynamics. *J. Chem. Phys.*, 146(1):014108, January 2017.

[10] Aldo Glielmo, Brooke E Husic, Alex Rodriguez, Cecilia Clementi, Frank Noé, and Alessandro Laio. Unsupervised learning methods for molecular simulation data. *Chem. Rev.*, May 2021.

[11] Cory Hargus, Katherine Klymko, Jeffrey M Epstein, and Kranthi K Mandadapu. Time reversal symmetry breaking and odd viscosity in active fluids: Green-Kubo and NEMD results. *J. Chem. Phys.*, 152(20): 201102, May 2020.

[12] Judith A Harrison, J David Schall, Sabina Maskey, Paul T Mikulski, M Todd Knippenberg, and Brian H Morrow. Review of force fields and intermolecular potentials used in atomistic computational materials research. *Applied Physics Reviews*, 5(3):031104, September 2018.

[13] Eric J Heller. The correspondence principle and intramolecular dynamics. *Faraday Discuss. Chem. Soc.*, 75(0):141–153, January 1983.

[14] Dorit S. Hochbaum and David B. Shmoys. A best possible heuristic for the k-center problem. *Mathematics of Operations Research*, 10(2):180–184, May 1985.

[15] M Karplus and J Kuriyan. Molecular dynamics and protein function. *Proc. Natl. Acad. Sci. U. S. A.*, 102 (19):6679–6685, May 2005.

[16] João Marcelo Lamim Ribeiro, Davide Provasi, and Marta Filizola. A combination of machine learning and infrequent metadynamics to efficiently predict kinetic rates, transition states, and molecular determinants of drug dissociation from G protein-coupled receptors. *J. Chem. Phys.*, 153(12):124105, September 2020.

[17] Yueheng Lan, Timothy C Elston, and Garegin A Papoian. Elimination of fast variables in chemical langevin equations. *J. Chem. Phys.*, 129(21):214115, December 2008.

[18] John M. Lee. *Introduction to Smooth Manifolds*. Springer, 2003.

[19] Marina Meila, Samson Koelle, and Hanyu Zhang. A regression approach for explaining manifold embedding coordinates. November 2018.

[20] Frank Noé, Alexandre Tkatchenko, Klaus-Robert Müller, and Cecilia Clementi. Machine learning for molecular simulation. *Annu. Rev. Phys. Chem.*, 71: 361–390, April 2020.

[21] J. Preto and C. Clementi. Fast recovery of free energy landscapes via diffusion-map-directed molecular dynamics. *Physical Chemistry Chemical Physics*, 16(36): 19181–19191, 2014.

[22] S Y Reddy and Vikram K Kuppa. Molecular dynamics simulations of organic photovoltaic materials: Structure and dynamics of oligothiophene. *J. Phys. Chem. C*, 116(28):14873–14882, July 2012.

[23] João Marcelo Lamim Ribeiro, Pablo Bravo, Yihang Wang, and Pratyush Tiwary. Reweighted autoencoded variational bayes for enhanced sampling (RAVE). *J. Chem. Phys.*, 149(7):072301, August 2018.

[24] M. A. Rohrdanz, W. Zheng, M. Maggioni, and C. Clementi. Determination of reaction coordinates via locally scaled diffusion map. *The Journal of chemical physics*, 134(12), 2011.

[25] Mary A. Rohrdanz, Wenwei Zheng, and Cecilia Clementi. Discovering mountain passes via torchlight: Methods for the definition of reaction coordinates and pathways in complex macromolecular reactions. *Annual Review of Physical Chemistry.64:295-316*, 64: 295–316, 2013.

[26] Andrew W. Senior, Richard Evans, John Jumper, James Kirkpatrick, Laurent Sifre, Tim Green, Chongli Qin, Augustin Žídek, Alexander W. R. Nelson, Alex Bridgland, Hugo Penedones, Stig Petersen, Karen Simonyan, Steve Crossan, Pushmeet Kohli, David T. Jones, David Silver, Koray Kavukcuoglu, and Demis Hassabis. Improved protein structure prediction using potentials from deep learning. *Nature*, 577(7792):706–710, 2020. doi: 10.1038/s41586-019-1923-7. URL https://doi.org/10.1038/s41586-019-1923-7.

[27] Zahra Shamsi, Kevin J Cheng, and Diwakar Shukla. Reinforcement learning based adaptive sampling: REAPing rewards by exploring protein conformational landscapes. *J. Phys. Chem. B*, 122(35):8386–8395, September 2018.

[28] Hythem Sidky, Wei Chen, and Andrew L Ferguson. Machine learning for collective variable discovery and enhanced sampling in biomolecular simulation. *Mol. Phys.*, 118(5):e1737742, March 2020.

[29] Amit Singer, Radek Erban, Ioannis G Kevrekidis, and Ronald R Coifman. Detecting intrinsic slow variables in stochastic dynamical systems by anisotropic diffusion maps. *Proc. Natl. Acad. Sci. U. S. A.*, 106(38): 16090–16095, September 2009.

[30] Justin S Smith, Benjamin T Nebgen, Roman Zubatyuk, Nicholas Lubbers, Christian Devereux, Kipton Barros, Sergei Tretiak, Olexandr Isayev, and Adrian E Roitberg. Approaching coupled cluster accuracy with a general-purpose neural network potential through transfer learning. *Nat. Commun.*, 10(1):2903, July 2019.

[31] Pratyush Tiwary and Michele Parrinello. From metadynamics to dynamics. *Phys. Rev. Lett.*, 111(23):230602, December 2013.

[32] Josh Tobin, Rachel Fong, Alex Ray, Jonas Schneider, Wojciech Zaremba, and Pieter Abbeel. Domain randomization for transferring deep neural networks from simulation to the real world. March 2017.

[33] Yihang Wang, João Marcelo Lamim Ribeiro, and Pratyush Tiwary. Machine learning approaches for analyzing and enhancing molecular dynamics simulations. *Curr. Opin. Struct. Biol.*, 61:139–145, April 2020.

[34] Wenshuai Zhao, Jorge Peña Queralta, and Tomi Westerlund. Sim-to-Real transfer in deep reinforcement learning for robotics: a survey. September 2020.