

## Unsupervised Learning Methods for Molecular Simulation Data

Aldo Glielmo,<sup>†</sup> Brooke E. Husic,<sup>†</sup> Alex Rodriguez,<sup>†</sup> Cecilia Clementi, Frank Noé, and Alessandro Laio\*

Cite This: *Chem. Rev.* 2021, 121, 9722–9758

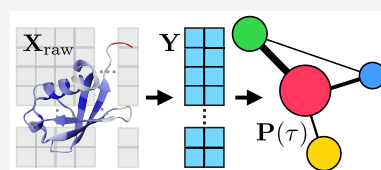
Read Online

ACCESS |

Metrics & More

Article Recommendations

**ABSTRACT:** Unsupervised learning is becoming an essential tool to analyze the increasingly large amounts of data produced by atomistic and molecular simulations, in material science, solid state physics, biophysics, and biochemistry. In this Review, we provide a comprehensive overview of the methods of unsupervised learning that have been most commonly used to investigate simulation data and indicate likely directions for further developments in the field. In particular, we discuss *feature representation* of molecular systems and present state-of-the-art algorithms of *dimensionality reduction*, *density estimation*, and *clustering*, and *kinetic models*. We divide our discussion into self-contained sections, each discussing a specific method. In each section, we briefly touch upon the mathematical and algorithmic foundations of the method, highlight its strengths and limitations, and describe the specific ways in which it has been used-or can be used-to analyze molecular simulation data.



### CONTENTS

1. Introduction	9723	5.2.1. DBSCAN	9739
2. Feature Representation	9724	5.2.2. Density Peak Clustering	9739
2.1. Representations for Macromolecular Systems	9724	6. Kinetic Models	9740
2.2. Representations for Condensed Matter Systems	9725	6.1. Time-Lagged Independent Component Analysis	9740
2.3. Representation Learning	9725	6.2. Variational Approach to Conformational Dynamics	9741
3. Dimensionality Reduction and Manifold Learning	9726	6.3. Markov State Modeling	9741
3.1. Linear Dimensionality Reduction Methods	9727	6.4. Koopman Models and VAMP	9743
3.1.1. Principal Component Analysis	9727	6.5. VAMPnets	9744
3.1.2. Multidimensional Scaling	9727	7. Relevant Software	9745
3.2. Nonlinear Dimensionality Reduction	9728	7.1. Feature Representations	9745
3.2.1. Isomap	9728	7.2. Dimensionality Reduction	9745
3.2.2. Kernel PCA	9728	7.3. Density Estimation	9745
3.2.3. Diffusion Map	9729	7.4. Clustering	9745
3.2.4. Sketch-Map	9730	8. Conclusion and Discussion	9746
3.2.5. <i>t</i> -Distributed Stochastic Neighbor Embedding	9730	Author Information	9748
3.2.6. Deep Manifold Learning Methods	9731	Corresponding Author	9748
4. Density Estimation	9732	Authors	9748
4.1. Parametric Density Estimation	9733	Author Contributions	9748
4.2. Nonparametric Density Estimation	9733	Notes	9748
4.2.1. Histograms	9733	Biographies	9748
4.2.2. Kernel Density Estimation	9733	Acknowledgments	9748
4.2.3. <i>k</i> -Nearest Neighbor Estimator	9734	References	9749
5. Clustering	9735	Note Added after ASAP Publication	9758
5.1. Partitioning Schemes	9735		
5.1.1. <i>k</i> -Means and <i>k</i> -Medoids	9736		
5.1.2. Leader Algorithms: <i>k</i> -Centers and Regular-Space Clustering	9736		
5.1.3. Spectral clustering	9737		
5.1.4. Hierarchical Clustering	9738		
5.2. Density-Based Clustering	9738		

**Special Issue:** Machine Learning at the Atomic Scale

**Received:** November 5, 2020

**Published:** May 4, 2021



## 1. INTRODUCTION

In recent years, we have witnessed a substantial expansion in the amount of data generated by molecular simulation. This has inevitably led to an increased interest in the development and use of algorithms capable of analyzing, organizing, and eventually, exploiting such data to aid or accelerate scientific discovery.

The data sets obtained from molecular simulations are very large both in terms of the number of data points—namely, the number of saved configurations along the trajectory—and in terms of the number of particles simulated, which can be several millions or more. However, we know both empirically and from fundamental physics that such data usually have much lower-dimensional representations that convey the relevant information without significant information loss. A striking example is given by the kinetics of complex conformational changes in biomolecules, which, on long time scales, can be well described by transition rates between a few discrete states. Moreover, symmetries, such as the invariance of physical properties under translation, rotation, or permutation of equivalent particles, can also be leveraged to obtain a more compact representation of simulation data.

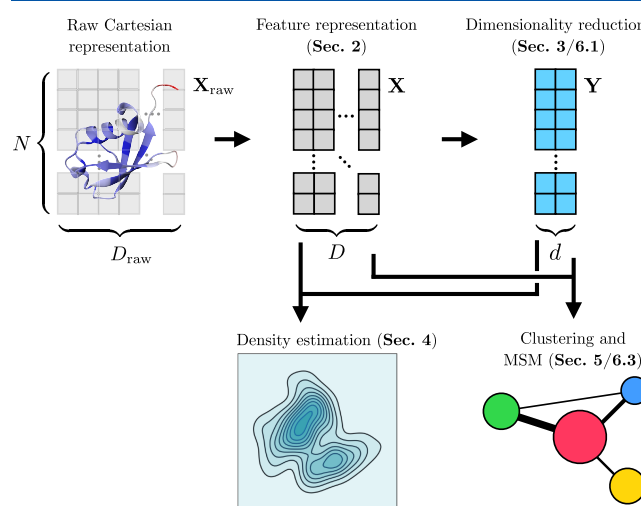
Traditionally, finding such low-dimensional representations is considered a task which can be tackled based on domain knowledge: The analysis of molecular simulations is often performed by choosing a small set of *collective variables* (CVs), possibly complex and nonlinear functions of the coordinates, that are assumed to provide a satisfactory description of the thermodynamic and kinetic properties of the system. If the CV is appropriately chosen, a histogram created from the CV summarizes all the relevant information, even if the molecular system includes millions or billions of atoms. The list of possible CVs among which one can choose is enormous and is still growing. Tools now exist which can describe phenomena of high complexity, such as the packing of a molecular solid, the allostery of a biomolecule, or the folding path of a protein. However, choosing the right CV remains to some extent an art, which can be successfully accomplished only by domain experts, or by investing a significant amount of time in a trial-and-error procedure.

Machine learning (ML) has emerged as a conceptually powerful alternative to this approach. ML algorithms can be broadly divided in three categories.<sup>1</sup> In *supervised* learning, a training data set consisting of input–output pairs is available, and a ML algorithm is trained with the goal of providing predictions of the desired output for unseen input values. This approach has been extensively used to predict specific properties of materials and molecules, such as the atomization energy of a compound or the force acting on an atom during a trajectory. In *unsupervised* learning, no specific output is available for the data in the training set, and the goal of the ML algorithm is to extract useful information using solely the input values. A typical application of this approach in the field of molecular simulation is the construction of low dimensional collective variables, which can compactly yet effectively describe a molecular trajectory. Finally, in *reinforcement* learning, no data at all is used to train the ML model, which instead learns by “trial and error” and by continuously interacting with its environment. Reinforcement learning has proven particularly successful at certain computer science challenges (such as playing board games), and it is now beginning to find application in molecular and materials science. In this Review, we focus exclusively on unsupervised learning

methods. We aim to provide an overview of all algorithms currently used to extract simplified models from molecular simulations to understand the simulated systems on a physical level. The review is oriented toward researchers in the fields of computational physics, chemistry, materials and molecular science, who routinely deal with large volumes of molecular dynamics simulation data, and are hence interested in using, or extending, the techniques we describe.

Some of the algorithms we review are as old as the simulation methods themselves, and have been successfully used for decades. Others are much more recent and their conceptual and practical power is only now becoming clear. Despite the recent surge in, for example, machine learning algorithms for determining CVs, many problems in the field can still be considered open, making this research area extremely active.

We divide our discussion in five sections, each of which will include its own introduction section: *feature representation*, *dimensionality reduction*, *density estimation*, *clustering*, and *kinetic models*. A graphical table of contents depicting the different methods and how they relate to each other is shown in Figure 1.



**Figure 1.** Illustration of the possible steps that can be performed to analyze data from a molecular simulation with an indication of the particular section where these are discussed.

In Section 2, we discuss the choice of feature representation for atomistic and molecular systems, a topic relevant for any analysis or application of learning algorithm. We, then, turn to the description of unsupervised learning algorithms, which we divide in four groups. In Section 3, we review algorithms of *dimensionality reduction*, whose primary objective is to provide a low dimensional representation of the data set that is easy to analyze or interpret. While in molecular simulations the probability density from which the configurations are harvested is in principle known (e.g., the Boltzmann distribution), this probability density is defined over the whole molecular state space, which is too high-dimensional to be visualized and understood. In Section 4, we review algorithms of *density estimation*, which enable the estimation of probability distributions restricted to sets of relevant variables describing the data. These variables can be chosen, for example, using the techniques described in Section 3. In Section 5, we review *clustering* algorithms. Clustering divides the data set into a few distinct groups, or “clusters,” whose elements are similar according to a certain notion of distance in their original space. These

techniques allow to capture the gross features of the probability distribution; for example, the presence of independent probability peaks, even without performing a preliminary dimensionality reduction. Therefore, the techniques described in this section can be considered complementary to those described in Section 3. When clustering is viewed as an assignment of data points to integer labels, clustering itself can be viewed as an extreme form of dimensionality reduction. In Section 6, we present *kinetic models*. While Sections 3–5 focus on modeling methods in which the ordering of the data points is not considered, kinetic models instead exploit dynamical information (i.e., ordering of data points in time) to reduce the dimensionality of the system representation. Altogether, this set of approaches is qualitatively based on the requirement that a meaningful low-dimensional model should reproduce the relevant time-correlation properties of the original dynamics (e.g., the transition rates). Throughout the review we present these techniques highlighting their specific application to the analysis of molecular dynamics, and discussing their advantages and disadvantages in this context. In Section 7 we list the software programs which are most currently used to perform the different unsupervised learning analysis described. Finally, in the Conclusions (Section 8), we present a general perspective on the important open problems in the field.

Some other review articles have a partial overlap with the present work. In particular, ref 2 reviews algorithms of dimensionality reduction for collective variable discovery, while refs 3 and 4 also review some approaches to build kinetic models. In this work, we review not only all these approaches but also other algorithms of unsupervised learning, namely, density estimation and clustering, focusing on the relationship between these different approaches and on the perspectives opened by their combination. Other valuable review articles of potential significance to the reader interested in machine learning for molecular and materials science are ref 5–9.

## 2. FEATURE REPRESENTATION

Throughout this Review, we assume to have obtained molecular data from a simulation procedure, such as Monte Carlo (MC)<sup>10</sup> or molecular dynamics (MD).<sup>11</sup> In general, the data set comes in the form of a *trajectory*: a series of *configurations*, each containing the positions of all particles in the system. These particles usually correspond to atoms but may also represent larger “sites”; for example, multiple atoms in a coarse graining framework. Given a trajectory data set, before any machine learning algorithm can be deployed, we must first choose a specific numerical representation  $\mathbf{X}$  for our trajectory. This amounts to choosing a set of “features” (often referred to as “descriptors” or “fingerprints”) that adequately describes the system of interest.

The “raw” trajectory data set (the direct output of the simulation procedure) is represented by a matrix  $\mathbf{X}_r \in \mathbb{R}^{N \times D_r}$ , where  $r$  stands for “raw”,  $N$  is the number of simulation time points collected, and  $D_r$  is the number of degrees of freedom in the data set. When three-dimensional spatial coordinates are retained from the simulation,  $D_r$  is equal to three times the number of simulated particles. When momenta are also retained,  $D_r$  is equal to six times the number of simulated particles. It is from this representation that we seek to featurize our data set; namely, transform our data into a new matrix  $\mathbf{X} \in \mathbb{R}^{N \times D}$ . For later reference, we denote the function that performs the featurization on each “raw” data point  $\mathbf{x}_{i,r}$  as  $\chi: \mathbb{R}^{D_r} \rightarrow \mathbb{R}^D$ , that is,

$$\mathbf{x}_i = \chi(\mathbf{x}_{i,r}) \quad (1)$$

A trivial featurization is to let  $\mathbf{X} = \mathbf{X}_r$  (and thus  $D = D_r$ ). In this case, the “raw” data output of the simulation is directly used for analysis. For molecular systems, however, this is often disadvantageous, since  $D_r$  is typically very large.

When sufficient prior knowledge of the system is available, it is convenient to choose a feature space such that  $D \ll D_r$  that can appropriately characterize the molecular motion of interest without significant loss of information. Often, the number of degrees of freedom required to encode many physical properties or observables of a molecular system is relatively low.<sup>12,13</sup> In the following, we will review features that have been frequently used to represent atomic and molecular systems. We can divide these features into two fundamentally different groups, which we will discuss in turn. For certain physical systems, such as biomolecules, each atom may be considered as having a unique identity that represents its position in a molecular graph. For such systems, it is often convenient to use features that are *not* invariant with respect to permutations of chemically identical atoms. On the other hand, for most condensed matter systems, and materials, atoms of the same element should be treated as indistinguishable, and descriptors should be invariant under any permutation of chemically identical atoms. We conclude this section with a brief discussion of representation learning, a promising new direction for determining features automatically.

It is worth noticing here that an explicit numerical representation for the molecular trajectory is not always necessary since some algorithms can work even if only a distance measure between molecular structures is defined.<sup>14–22</sup> The main limitation of this approach is that its computational cost scales quadratically with the number of data points.

### 2.1. Representations for Macromolecular Systems

Many of the methods discussed in this Review have been applied to classical simulation data sets of systems such as DNA, RNA, and proteins. A macromolecular simulation of such a system will contain at least one solute (for example, a protein) comprising 100–100 000 atoms connected by covalent bonds which cannot be broken. The system will also typically include several thousand water molecules, as well as several ions and lipids. Other small molecules, for example a drug or other ligand, are often also included.

In most analyses of macromolecules, the focus is on the solute and the solvent degrees of freedom are neglected. In the absence of an external force, we then expect the solute’s dynamics and thermodynamics to be invariant to translation and rotation of the molecule (in three dimensions for fully solvated systems and in two dimensions for membrane-bound systems). A simple manner of obtaining a set of coordinates obeying this invariance is to remove the solvent molecules and, then, superpose (i.e., align) each configuration in the trajectory to a reference structure of the molecule (or of the complex), which is known to be of physical or biological interest. This can be done by finding the rotation and the translation minimizing the root-mean-square deviation (RMSD) with the reference structure.<sup>23</sup>

While in many cases, this simple procedure is adequate, in many others it is suboptimal; for example, when a reference structure is not known or when the configurations are too diverse to be aligned to the same structure. This is the case for simulations of very mobile systems, such as RNA and intrinsically disordered proteins, or for the simulation of folding processes, in which the system dynamically explores extremely diverse configurations that cannot be meaningfully aligned to a



single structure. Beyond RMSD-based approaches, an appropriate choice of features satisfying rotational and translational invariance are the so-called internal coordinates, first introduced by Gordon and Pople.<sup>24</sup> For classical simulations of proteins or nucleic acids, one can consider the bond lengths and the angles formed by three consecutive atoms as approximately fixed, and therefore, the configuration of the molecule can be described by the value of the dihedral angles formed by four consecutive atoms.<sup>25,26</sup> For example, in a protein the configuration of the backbone is defined by the so-called  $\phi$ - and  $\psi$ -dihedral (Ramachandran) angles.<sup>27</sup> Using these coordinates, which by construction are invariant with respect to system rotation and translation, enables a significant dimensionality reduction of the system's representation. For example, an amino acid residue has six backbone atoms, which corresponds to 18 degrees of freedom if one uses their Cartesian coordinates to represent the residue. On the other hand, their position is determined with good approximation by the value of only the two Ramachandran angles.

For larger systems whose dynamics are characterized by more global changes in conformation (as in protein folding or allostery), it is more common to use a contact-based featurization. For example, the "contact distances" between all monomers can be used according to a predesignated site (e.g.,  $\alpha$ -carbons in a protein) or the closest (heavy) atoms of each pair. These contacts can be also defined by selecting a given cutoff distance and then using a contact indicator function.<sup>25</sup>

There are many more specialized feature representations that can be employed according to the problem under study. For example, in simulations that involve two or more solute molecules, such as ligand binding or protein–protein association, distances, angles, and other data specific to the binding site can be used.<sup>28,29</sup> In some cases, featurizations involving assignment to secondary structure elements,<sup>30</sup> solvent-accessible surface area,<sup>31</sup> or hand-picked features may be most appropriate.<sup>32</sup> Furthermore, different types of features can be concatenated (and appropriately scaled). In some cases, featurizations that explicitly or implicitly treat solvent molecules, lipids or ions are important. Examples include when characterizing the solvent dynamics around binding pockets,<sup>33</sup> ions moving through ion channels,<sup>34,35</sup> or the lipid composition around membrane proteins.<sup>36,37</sup> Several publications have compared the suitability of various feature sets for protein data for analyses, particularly in the context of the kinetic models described in Section 6.<sup>38–40</sup>

## 2.2. Representations for Condensed Matter Systems

In many condensed matter systems, such as solvents and materials, the physical properties of interest are invariant with respect to the exchange of equivalent atoms or molecules as such permutations merely exchange the particle labels. In such situations, it is often essential that the system configurations are represented in a permutation-invariant way. For example, in a pure water simulation, water molecules diffuse around and change places. Comparing different configurations by a simple RMSD, which depends on the order of molecules in the coordinate vector, is therefore inadequate.

A direct approach for working with Cartesian coordinates in a way that preserves permutational invariance is to relabel exchangeable particles or molecules such that the distance to a reference configuration is minimized. For example, in a simulation of water, one could define an ice-like configuration where water molecules are aligned on a lattice as a reference

configuration. The (arbitrary) sequence in which waters are enumerated in this configuration defines the reference labeling. Then, for every configuration visited during the simulation, one must determine to which label each water molecule should be assigned (i.e., which permutation matrix should be applied) such that the relabeled configuration will have the minimal RMSD to the reference configuration. This problem can be solved by a bipartite matching method, such as the Hungarian algorithm.<sup>41</sup> Such approaches have been applied to estimate solvation entropies<sup>42</sup> and sampling permutation-invariant system configurations.<sup>43</sup>

Another common strategy for designing permutation-invariant features is to first represent a system as a union of low order constituents ("n-plets") for which a permutation-invariant descriptor is computed. In this approach, invariance can be achieved by enumerating all permutations of subsets of permutable particles or molecules. While this is not possible for permutations (the number of which scales factorially), it is relatively straightforward for pairs or triplets of atoms.<sup>44–46</sup> As a simple example, a system of  $M$  atoms is first broken down into the  $M(M-1)/2$  unique pairs of atoms that constitute it. Then, the distribution functions of interatomic distances between two atomic species is computed. Finally, these distribution functions are binned to produce permutation invariant feature vectors.




Different descriptors have been proposed which build upon the above idea in the context of specific aims. For example, Atom-centered symmetry functions (ACSFs)<sup>47,48</sup> are obtained by expanding 2-body (radial) and 3-body (angular) distribution functions onto specific invariant bases. The ACSF representation was the first widely adopted representation for materials systems. ACSFs are very efficient to evaluate numerically and they have been shown to be able to resolve geometric information well enough to train highly efficient interatomic potentials.<sup>49,50</sup> ACSFs are now widely used in shallow neural networks that replace molecular force fields by learning quantum-chemical energies.<sup>47,48,51,52</sup>

Another commonly used descriptor for materials system is the so-called Smooth Overlap of Atomic Positions (SOAP).<sup>53,54</sup> The SOAP representation is obtained by expanding a smoothed atomic density function onto a radial and a spherical harmonics basis set. This representation can be considered an expansion of 3-body features.<sup>55</sup> The SOAP vector has also been shown to resolve geometric information very precisely.<sup>54</sup>

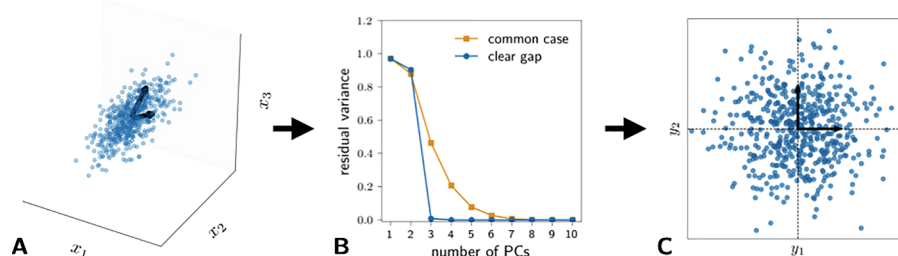
Many other representations based on the  $n$ -plets principle have been suggested. They differ in the specific low-order contributions chosen and in the particular basis in which the low order information is expanded. Notable examples are the Coulomb matrix,<sup>56</sup> the "Bag of Bonds",<sup>57</sup> the histograms of distances, angles, and dihedral angles (HDAD),<sup>58</sup> and the many body tensor representation (MBT).<sup>59</sup> In general, finding informative numerical representations for condensed matter systems is a very hard problem,<sup>60,61</sup> and an exhaustive review of all the approaches which have been proposed to tackle it is beyond the scope of this work. The interested reader can refer to refs 62–64 for a more comprehensive analysis of the different descriptors developed for systems with permutation invariance.

## 2.3. Representation Learning

In the past few years, substantial research has been directed toward learning a feature representation rather than manually selecting it. Good examples of representation learning algorithms for molecules are given by graph-neural networks (GNNs)<sup>65</sup> such as SchNet,<sup>66</sup> PhysNet,<sup>67</sup> DimeNet,<sup>68</sup> Cormor-

Linear manifolds	Curved-twisted manifolds	Highly nonlinear and complex manifolds
		
<ul style="list-style-type: none"> <li>• PCA (Sec. 3.1.1)</li> <li>• MDS (Sec. 3.1.2)</li> <li>• TICA (Sec. 6.1)</li> </ul>	<ul style="list-style-type: none"> <li>• Isomap (Sec. 3.2.1)</li> <li>• Kernel PCA (Sec. 3.2.2)</li> <li>• Diffusion map (Sec. 3.2.3)</li> </ul>	<ul style="list-style-type: none"> <li>• Sketchmap (Sec. 3.2.4)</li> <li>• t-SNE (Sec. 3.2.5)</li> <li>• Deep methods (Sec. 3.2.5 and Sec. 6.5)</li> </ul>
<ul style="list-style-type: none"> <li>• Depend on a single parameter</li> <li>• Very fast and very robust</li> <li>• Work only for linear manifolds</li> </ul>	<ul style="list-style-type: none"> <li>• Depend on few parameters</li> <li>• Relatively fast and stable</li> <li>• Works for manifolds that can be easily made linear</li> </ul>	<ul style="list-style-type: none"> <li>• Difficult to fine-tune to a specific problem</li> <li>• More computationally expensive</li> <li>• Can work for arbitrarily complex manifolds</li> </ul>

**Figure 2.** Different algorithms of dimensionality reduction discussed, divided in three groups. From left to right, the listed algorithms are capable of dealing with manifolds of higher complexity, which comes at the price of a higher computational cost and a larger number of free parameters to choose.



**Figure 3.** Illustration on the working principle of the PCA projection. (A) Two-dimensional linear manifold embedded in a high dimensional space. (B) Eigenspectrum of a covariance matrix of the data, as the manifold is two-dimensional a clear gap appears after the second eigenvalues (blue line); a more typical eigenspectrum is shown in orange. (C) Low-dimensional representation of the data obtained through PCA.

ant,<sup>69</sup> or Tensor field Networks.<sup>70</sup> These networks are usually trained to predict molecular properties, and they do so by performing continuous convolutions across the spatial neighborhoods of all atoms. In contrast to other neural network approaches based on fixed representations,<sup>47,71</sup> in GNNs the representations are not predefined but are instead learnable using convolutional kernels. The feature representation is a vector attached to each atom or particle. It is initialized to denote the chemical identity of the particle (e.g., nuclear charge or type of bead), and it is then updated in every neural network layer depending on the chemical environment of every particle and through the action of convolutional kernels whose weights are optimized during training. The representation found at the last GNN layer can thus be interpreted as a learned feature representation which encodes the many-body information required to predict the target property (e.g., the potential energy of the molecule).

GNNs have been extensively used to predict quantum-mechanical properties<sup>66–68,72</sup> and coarse-grained molecular models.<sup>73,74</sup> See refs 4 and 75 for recent reviews of these topics.

For the rest of this Review, we will assume that the featurization step has been performed; and a sufficiently general numerical representation  $\mathbf{X}$ , appropriate for the system under study, has been obtained.

### 3. DIMENSIONALITY REDUCTION AND MANIFOLD LEARNING

In this section, we review dimensionality reduction techniques that have been used most extensively in the analysis of molecular simulations. All these techniques involve transforming a data

matrix  $\mathbf{X}$  of dimensions  $N \times D$  into a new representation  $\mathbf{Y}$  of dimensions  $N \times d$ , with  $d \ll D$ , with the goal of preserving the information contained in the original data set. It is typically impossible to achieve this task exactly, and the different methods reviewed here can only provide different approximate solutions whose utility needs to be evaluated on a case by case basis. We first describe linear projection methods: Principal component analysis and Multidimensional scaling. These are arguably the simplest and most widely used techniques of dimensionality reduction. However, if the data do not lie on hyperplanes these linear methods can easily fall short. In the subsequent sections we describe a set of nonlinear projection methods that can deal also with data lying on twisted and curved manifolds. In order, we will cover Isomap, Kernel PCA, Diffusion map, Sketch-map, t-SNE, and deep learning methods.

We will present the theory beyond the mentioned algorithms only briefly; the interested reader can refer to existing specialized reviews for more information on such aspects.<sup>3,76,77</sup> Instead, we will focus our attention on the practical issues related to utilizing these methods for the analysis of molecular trajectories and highlight relative merits and pitfalls of each method.

The mentioned algorithms are grouped together for reference in Figure 2. The figure emphasizes that a manifold of increasing complexity can only be efficiently reduced in dimensionality using methods of increasing sophistication, which however will typically require a greater computational power and a longer trial and error procedure to be properly deployed.

### 3.1. Linear Dimensionality Reduction Methods

**3.1.1. Principal Component Analysis.** Principal component analysis (PCA)<sup>78–80</sup> is possibly the best known procedure for reducing the dimension of a data set and further benefits from an intuitive conceptual basis. The objective of PCA is to find the set of orthogonal directions along which the variance of the data set is the highest. These directions can be efficiently found as follows. First, the data is centered by subtracting the average, obtaining a zero-mean data matrix  $\hat{\mathbf{X}}$ . This operation guarantees the translational invariance of the PCA projection. Second, the covariance matrix of the data is estimated as  $\mathbf{C} = \frac{1}{N} \hat{\mathbf{X}}^T \hat{\mathbf{X}}$ . Finally, the eigenvectors of the matrix  $\mathbf{C}$  are found by solving the eigenproblem

$$\mathbf{C} \mathbf{v}_\alpha = \lambda_\alpha \mathbf{v}_\alpha \quad (2)$$

It is simple to show that the direction of maximum variance coincides with  $\mathbf{v}_1$ , the eigenvector corresponding to the largest eigenvalue  $\lambda_1$ .<sup>80</sup> The direction of maximum variance in the subspace orthogonal to  $\mathbf{v}_1$  is  $\mathbf{v}_2$ , and so on. The PCA representation is ultimately obtained by choosing a number of components  $d$ , and projecting the original data onto these vectors as  $\mathbf{Y} = \mathbf{X}\mathbf{V}$ , where  $\mathbf{V}$  is a matrix of dimension  $D \times d$  containing the first  $d$  eigenvectors of  $\mathbf{C}$ . An illustration of this procedure is shown in Figure 3a on a data set obtained by harvesting data points from an 10-dimensional Gaussian stretched in two dimensions.

Importantly, the eigenvalue  $\lambda_\alpha$  is exactly equal to the variance of the data along the given direction  $\mathbf{v}_\alpha$ .<sup>80</sup> For this reason, it is common to select the dimensionality  $d$  most appropriate for a PCA projection by looking at the eigenvalue spectrum as a function of the eigenvalue index. A clear gap in such a plot, as seen in the blue curve in Figure 3b, is an indication that a dimensionality reduction including the components before the gap is meaningful. In real world applications, however, it is common to observe a continuous, although fast, decay of the magnitude of the eigenvalues (orange curve in Figure 3b). In this situation the selection of  $d$  is more arbitrary. A common rule of thumb is to choose the smallest  $d$  that is able to capture a good portion of the total variance of the data set. In practice, one can choose the smallest  $d$  such that  $\sum_{i=1}^d \lambda_i / \sum_{i=1}^D \lambda_i \geq f$ , where  $f$  is a free parameter, which is often set to 0.98 or 0.95.

The use of PCA to analyze biomolecular trajectories was first proposed in ref 81–83. In these papers, it was numerically found that a very small fraction of coordinates were capable of describing the majority of the motion of the molecules studied. These coordinates have often been named *essential coordinates* and, consistently, the methodology has been named “essential dynamics”.<sup>83</sup> Importantly, the variation along the essential degrees of freedom was connected to the functional motion of the protein. In the years that followed, PCA has been extensively used to characterize both functional motions and the free energy surface of small peptides and proteins.<sup>84–90</sup> In ref 91, the use of PCA was also proposed in conjunction with metadynamics for enhanced sampling. In the context of solid state and materials physics, PCA has been commonly used for exploratory analysis, visualization, data organization, and structure–property prediction.<sup>92–94</sup>

In spite of its empirical success, the theoretical foundation of the use of PCA for the analysis of molecular trajectories has been the subject of debate. In particular, soon after essential dynamics was proposed it was argued that the sampling needed to robustly characterize the essential coordinates was beyond the time-

scales accessible to MD simulations.<sup>95,96</sup> Other studies, however, argued that the convergence time needed for the characterization of a stable eigenspace of principal components is reachable and in the range of nanoseconds of simulation time.<sup>97–100</sup> Aside from sampling concerns, the strong assumption of the existence of a linear manifold which correctly captures the important modes of variation of a molecular system can easily fall short, giving rise to systematic errors in analysis and predictions.<sup>101</sup> The approach presented in the next section provides a manner for overcoming this limitation.

**3.1.2. Multidimensional Scaling.** Multidimensional scaling (MDS)<sup>102,103</sup> is closely related to PCA, and equivalent to it under certain conditions. MDS will be the basis for the advanced nonlinear dimensionality reduction techniques which will be described in the following sections. MDS provides a low dimensional description of the data by finding the  $d$  dimensional space which best preserves the pairwise distances between points. It does so by minimizing a cost function quantifying the difference between the pairwise distance  $R_{ij}$  as measured in the original  $D$  dimensional space and the one computed in the low dimensional embedding

$$L(\mathbf{Y}) = \sum_{ij} (R_{ij} - \|\mathbf{y}_i - \mathbf{y}_j\|)^2 \quad (3)$$

If the distance matrix is given by  $R_{ij} = \|\mathbf{x}_i - \mathbf{x}_j\|$ , then, it is simple to show that the vectors  $\mathbf{y}_\alpha$  that minimize eq 3 are found solving the following eigenproblem<sup>104</sup>

$$\mathbf{K} \mathbf{y}_\alpha = \lambda_\alpha \mathbf{y}_\alpha \quad (4)$$

and taking  $\mathbf{y}_\alpha = \sqrt{\lambda_\alpha} \mathbf{v}_\alpha$ . The matrix  $\mathbf{K}$  in eq 4 contains the inner products of all the centered data vectors  $K_{ij} = \hat{\mathbf{x}}_i^T \hat{\mathbf{x}}_j$ , where  $\hat{\mathbf{x}}_i = \mathbf{x}_i - \frac{1}{N} \sum_j \mathbf{x}_j$ . The key trick of MDS is that such a matrix can be obtained from the matrix of distances  $R_{ij} = \|\mathbf{x}_i - \mathbf{x}_j\|$  in a simple way, namely<sup>104,105</sup>

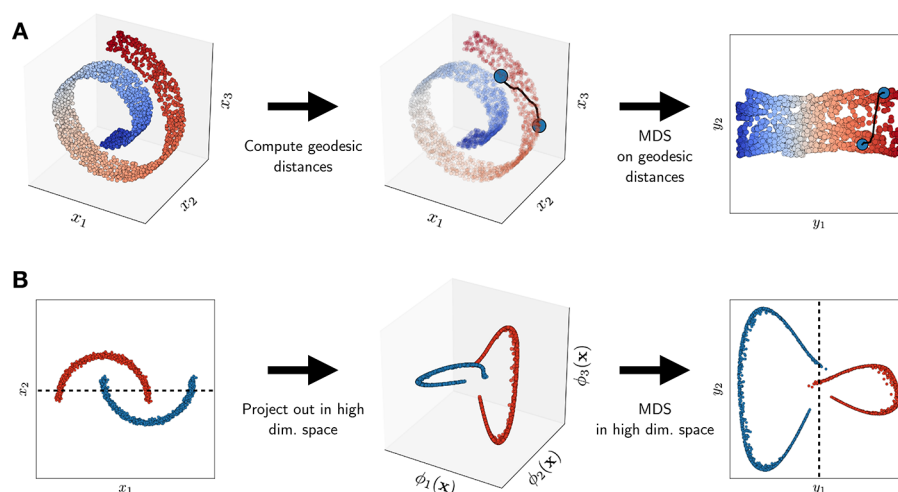
$$K_{ij} = -\frac{1}{2} \left( R_{ij}^2 - \frac{1}{N} \sum_l R_{lj}^2 - \frac{1}{N} \sum_m R_{im}^2 + \frac{1}{N^2} \sum_{lm} R_{lm}^2 \right) \quad (5)$$

It is important to note that the embeddings generated by MDS and PCA are *exactly equivalent* if the distance between the data points is estimated as  $R_{ij} = \|\mathbf{x}_i - \mathbf{x}_j\|$ . This follows from the fact that the eigenvectors of the covariance matrix  $\{\mathbf{v}_\alpha^c\}$  and of those of the  $\mathbf{K}$  matrix  $\{\mathbf{v}_\alpha^k\}$  are related to each other as  $\sqrt{\lambda_\alpha^k} \mathbf{v}_\alpha^k = \mathbf{X} \mathbf{v}_\alpha^c$ .<sup>76</sup>

The covariance matrix  $\mathbf{C}$  and the matrix of inner products  $\mathbf{K}$  have dimensions  $D \times D$  and  $N \times N$ , respectively, which greatly affects the computational cost of the methods. So, for instance, if the number of points greatly exceeds the number of dimensions  $N \gg D$ , PCA is more computationally efficient, while the contrary is true if  $D \gg N$ .

The most important difference between PCA and MDS lies in the fact that MDS can be used also when the data matrix  $\mathbf{X}$  is not available and one is only provided with the matrix  $R_{ij}$  of pairwise distances between points. This is an important feature of MDS, which allows it to also work in the context of nonlinear dimensionality reduction as we will describe in the next section.





**Figure 4.** (A) Illustration of Isomap on a Swiss roll data set. The original 3D data set (left) is projected on a 2D space in a way that optimally preserves geodesic distances between points (middle and right). (B) Illustration of Kernel PCA on a data set with two segments that are not linearly separable. The original 2D data set (left) is first represented in a higher dimensional space, where it becomes linearly separable (middle), and finally, MDS is performed on the transformed data set (right). Note that the transformed data set typically lies in a space of very high—and often infinite—dimensionality, and the 3D embedding shown here only serves illustrative purposes.

### 3.2. Nonlinear Dimensionality Reduction

**3.2.1. Isomap.** The Isometric feature mapping (Isomap) algorithm<sup>106</sup> was introduced with the goal of alleviating the problems of linear methods, such as PCA, which fail to find the correct coordinates whenever the embedding manifold is not a hyperplane. The key idea of Isomap is the generation of a low dimensional representation that best preserves the *geodesic* distances between the data points on the data manifold, rather than the Cartesian distance.

Isomap comprises three steps. First, a graph of the local connectivities is constructed. In this graph, each point is linked to its  $k$ th nearest neighbors with edges weighted by the pairwise distances. Second, an approximation of the geodesic distance between all pairs of points is computed as the shortest path on this graph. Finally, MDS is performed on the matrix  $R_{ij}$  containing the geodesic distances. The final representation thus minimizes the loss function in eq 3, and provides the low dimensional Euclidean projection that best preserves the computed geodesic distances. An illustration of the working principle of Isomap is presented in Figure 4.

A drawback of Isomap is its potential for topological instabilities. Indeed, if the manifold containing the data is not isomorphic to a hyperplane (for example, it is isomorphic to a sphere or to a torus) the procedure becomes ill defined, since a sphere or a torus cannot be mapped to a hyperplane without cutting it. More generally, the quality of the representation generated depends on the quality of the geodesic distances, which can only be estimated approximately. In particular, the algorithm used to compute geodesic distances requires choosing which data points can be considered directly connected, namely close enough that their geodesic distance coincides with their Cartesian distance. Considering connected points which are too far apart can bring to a severe underestimation of the geodesic distance, if the manifold is strongly curved.<sup>107</sup>

The computation of the geodesic distance between all pairs of points is also the main performance bottleneck when using Isomap on large data sets of molecular trajectories. For this reason, when Isomap was first used for the analysis of molecular data sets,<sup>101</sup> it involved the preselection of a small number of landmark points  $n_L$  that were assumed to correctly span the data

manifold. This modified procedure, named “ScImap” (Scalable Isomap), is much faster than the standard Isomap implementation since the geodesic distance is computed only between a small fraction  $n_L \ll N$  of landmark points. The scalability of Isomap to large data sets was further improved with the introduction of “DPES-ScImap” (Distance-based Projection onto Euclidean Space ScImap) in ref.<sup>108</sup> This procedure involves an initial projection of the points onto a low-dimensional Euclidean space.

In ref 101, it is shown that using Isomap in place of PCA allows describing the folding process of a small protein by much less variables. This result is confirmed in refs 109 and 110, in which it is shown that Isomap coordinates faithfully describe the motion of small molecules and the free energy landscape of small peptides. On the other hand, in ref 111, only small improvements were observed when using Isomap in place of PCA for describing the folding process of another peptide. In ref 112, an Isomap embedding was successfully used to generate the collective biasing variables for a metadynamics simulation. In ref 113, Isomap was employed to construct an enhanced MD sampling method.

In general it is clear that the extent to which a nonlinear method like Isomap proves beneficial—or even necessary—depends upon the degree of nonlinearity of the manifold in which the molecular trajectory is (approximately) contained). This nonlinearity depends on the system under study as well as on the type of coordinates chosen to describe it (see section 2). While it is difficult to determine precisely the degree of nonlinearity of a data manifold, an approximate estimate can be obtained by comparing its linear dimension, obtained for instance by an analysis of the PCA eigenspectrum (Figure 3B), with its nonlinear *intrinsic dimension*, which can be computed using several tools.<sup>114,115</sup> An intrinsic dimension much smaller than the linear dimension can then be seen as an indication of the presence of curvature and topological complexity in the manifold. Providing a quantitative measure of this complexity can still be considered an open problem.

**3.2.2. Kernel PCA.** A different strategy for finding a low-dimensional representation for data points embedded in a curved and twisted manifold is Kernel PCA.<sup>116</sup> In Kernel PCA,

data are represented in a new space using a nonlinear transformation  $\phi(\mathbf{x})$ , where  $\phi$  is a high-dimensional vector-valued function. A linear dimensionality reduction is then performed in this space. The transformation  $\phi$  should be chosen in such a way that even if the original data manifold is nonlinear, the transformed data set is approximately linear, allowing for the usage of MDS in the transformed space. An illustration of this concept is provided in Figure 4.

In Kernel PCA, the transformation is not performed explicitly, but obtained through the use of a kernel function  $\kappa(\mathbf{x}, \mathbf{x}')$ . By definition, a kernel function represents a dot product in some vector space. Hence, one sets

$$\kappa(\mathbf{x}_i, \mathbf{x}_j) = \phi(\mathbf{x}_i)^T \phi(\mathbf{x}_j) \quad (6)$$

which implicitly defines the function  $\phi$ . One then implicitly removes the mean from the transformed data<sup>105</sup> by the same procedure used in MDS (see eq 5)

$$\hat{K}_{ij} = K_{ij} - \frac{1}{N} \sum_l K_{li} - \frac{1}{N} \sum_m K_{jm} + \frac{1}{N^2} \sum_{lm} K_{lm} \quad (7)$$

where  $K_{ij} = \kappa(\mathbf{x}_i, \mathbf{x}_j)$ . Finally, the MDS algorithm is used on such a matrix, providing the principal components of the data in the transformed space.

It is crucial to note that the transformed data matrix is never explicitly computed in the algorithm. One only needs to compute the matrix of dot products  $\mathbf{K}$ , and this can be done efficiently through the use of the kernel function. This fact is known as the “kernel trick”,<sup>1</sup> and it allows to transform the data in spaces of very high and often infinite dimensionality, thereby enabling the encoding of highly nonlinear manifolds without knowing suitable feature functions. A drawback, however, is that the kernel matrix has the dimension of  $N^2$ , resulting in unfavorable storage and computing costs for large data sets.

The simplest kernel one can use is the linear kernel  $\kappa(\mathbf{x}_i, \mathbf{x}_j) = \mathbf{x}_i^T \mathbf{x}_j$ , which makes Kernel PCA equivalent to standard PCA. Polynomial kernels of the kind  $\kappa(\mathbf{x}_i, \mathbf{x}_j) = (\mathbf{x}_i^T \mathbf{x}_j)^p$  can increase the feature space systematically for larger values of the parameter  $p$ . The use of a specific kernel, allows recovering Isomap.<sup>117</sup> Perhaps the most widely used kernel for Kernel PCA is the Gaussian kernel  $\kappa(\mathbf{x}_i, \mathbf{x}_j) = e^{-\|\mathbf{x}_i - \mathbf{x}_j\|^2 / 2h^2}$ . The Gaussian kernel depends on the parameter  $h$ , which determines the length scale of distances below which two points are considered similar in the induced feature space.

Kernel PCA is a powerful method for nonlinear dimensionality reduction, since in principle it can overcome the limitations of linear methods without a significant increase in computational cost. However, a practical issue in using Kernel PCA lies in its sensitivity to the specific choice of kernel function used and any parameters it may have. In ref 118, it is suggested to choose the kernel by systematically increasing the kernel complexity (from the simple linear kernel to the polynomial kernel with  $p = 2$  and so on) until a clear gap in the eigenspectrum of the kernel matrix appears (see discussion of Figure 3b). Although this procedure is not guaranteed to be successful in general, in ref.<sup>118</sup> the authors successfully identify the reaction coordinate of a protein with the aid of Kernel PCA and a polynomial kernel. We refer to ref.<sup>119</sup> for more information on the use of Kernel PCA and the various possible choices of kernels to analyze molecular motion.

In the context of materials physics, Kernel PCA has been particularly fruitful when used in conjunction with the SOAP descriptors (reviewed in section 2.2). Indeed, SOAP descriptor forms the basis for accurate interpolators of atomic energies and

forces,<sup>120–122</sup> meaning that this descriptor generates data manifolds in which similar materials lie close to each other. Kernel PCA has been successfully used for visualization and exploration of materials databases,<sup>123</sup> for identifying new materials candidates,<sup>92</sup> and to predict phase stability of crystal structures.<sup>124</sup>

A common feature of PCA, MDS, and Isomap is that their result is strongly affected by the largest pairwise distances between the data points. This can be a problem as the distances deemed relevant for the analysis of molecular trajectories are often those of intermediate value: not the largest, which only convey the information that the configurations are different, and not the smallest, whose exact value is also determined by irrelevant details (for example bond vibrations). Kernel PCA can partially accommodate this issue via the choice of kernel, since the distances which are preserved are those computed in the transformed space that the kernel implicitly generates. In the next subsection, we will describe Diffusion map, another projection algorithm that can be used in these situations.

**3.2.3. Diffusion Map.** Diffusion map<sup>125,126</sup> is another technique similar to Kernel PCA, which enables the discovery of nonlinear variables capable of providing a low-dimensional description of the system. The diffusion map has a direct application to the analysis of molecular dynamics trajectories generated by a diffusion process, as the collective coordinates emerging from it approximate the eigenfunctions of the Fokker–Planck operator of the process. In ref 127, it has been shown that the diffusion map eigenfunctions are equivalent, up to a constant, to the eigenfunctions of an overdamped Langevin equation.

To compute a diffusion map, one starts by computing a Gaussian kernel

$$K_{ij} = \exp(-\|\mathbf{x}_i - \mathbf{x}_j\|^2 / 2h^2) \quad (8)$$

identical with the one introduced in the previous section. The length scale parameter  $h$  entering eq 8 has here a specific physical interpretation. It can be seen as the scale within which transitions between two configurations can be considered “direct” or without any significant barrier crossing. In principle, all pairs of configurations enter eq 8, but in practice, only the pairs closer than  $h$  are relevant in the definition of the kernel  $K_{ij}$ , as the contribution of pairs further apart decays exponentially with their distance. For this reason, in practical implementations usually a cutoff distance multiple of  $h$  is defined and only the distances of pairs within this cutoff are computed.

To provide an approximation of the eigenfunctions of the Fokker–Planck operator, the kernel (eq 8) needs to be properly normalized<sup>128</sup>

$$\tilde{K}_{ij} = \frac{K_{ij}}{\sqrt{\sum_k K_{ik} \sum_s K_{js}}} \quad (9)$$

We note that different normalizations are possible and commonly used to approximate different operators (e.g., graph Laplacian or Laplace–Beltrami instead of Fokker–Planck).<sup>129</sup> From this normalized kernel, one estimates the diffusion map transition matrix as

$$P_{ij} = \frac{\tilde{K}_{ij}}{\sum_j \tilde{K}_{ij}} \quad (10)$$



The resulting elements  $P_{ij}$  can be thought of as the transition probability from data point  $i$  to data point  $j$ . Indeed  $\sum_j P_{ij} = 1$ .

We now consider the spectral decomposition of  $\mathbf{P}$

$$\mathbf{P}\mathbf{v}_\alpha = \lambda_\alpha \mathbf{v}_\alpha \quad (11)$$

Since  $\mathbf{P}$  is a stochastic matrix, it is positive-definite, only one of its eigenvalues is equal to 1, and all the others are positive and smaller than 1. If the collection of configurations used to construct the diffusion matrix samples the equilibrium distribution, in the limit of  $h \rightarrow 0$  and infinite sampling, the eigenvectors  $\mathbf{v}_\alpha$  converge to the eigenfunctions of the Fokker–Planck operator associated with the dynamics. In practice, for finite  $h$  and finite sampling, the eigenvectors  $\mathbf{v}_\alpha$  provide a discrete approximations to these eigenfunctions, and are called “diffusion coordinates”. Reweighting tricks can be used to apply the diffusion map approach to analyze molecular dynamics trajectories out of equilibrium.<sup>130</sup>

It can be shown that the Euclidean distance in the diffusion coordinates space, scaled by the corresponding eigenvalues, defines a “natural” distance metric on the diffusion manifold, the so-called *diffusion distance*:

$$D_\tau^2(\mathbf{x}_i, \mathbf{x}_j) = \left\| p_\tau(\mathbf{x}'|\mathbf{x}_i) - p_\tau(\mathbf{x}'|\mathbf{x}_j) \right\|_{\pi^{-1}}^2 \quad (12)$$

$$= \int d\mathbf{x}' \pi(\mathbf{x}') (p_\tau(\mathbf{x}'|\mathbf{x}_i) - p_\tau(\mathbf{x}'|\mathbf{x}_j))^2 \quad (13)$$

$$= \sum_\alpha \lambda_\alpha^{2\tau} (\mathbf{v}_\alpha(\mathbf{x}_i) - \mathbf{v}_\alpha(\mathbf{x}_j))^2 \quad (14)$$

where  $p_\tau(\mathbf{x}'|\mathbf{x}_i)$  is the probability of being in configuration  $\mathbf{x}'$  after a time  $\tau$  for a diffusion process started at position  $\mathbf{x}_i$ , and  $\pi$  is the equilibrium distribution. Because of the equivalence expressed in eq 14, the diffusion coordinates provide an accurate description of the diffusion process, and are usually robust to noise.

Similarly to the other dimensionality reduction algorithms discussed so far, Diffusion map is particularly useful if the spectrum of  $\lambda$  exhibits a gap, say after the  $d$ -th eigenvalue, with  $\lambda_{d+1} \ll \lambda_d$ . In this case, the sum over the eigenvectors in eq 14 can be truncated including only the first  $d$  terms and the first  $d$  diffusion coordinates can be used to characterize the system. The diffusion distance (eq 14) has inspired the definition of a “kinetic distance”,<sup>131,132</sup> where the same mathematical form is retained but eigenvectors obtained with different spectral methods for the approximation of the dynamics eigenfunctions are used instead.<sup>133,134</sup>

The local scale parameter  $h$  entering the definition 8 is crucial in determining the transition probability between two configurations. Data configurations coming from the Boltzmann distribution are typically distributed very nonuniformly as they are highly concentrated in metastable states and very sparse in transition states. Therefore, a uniform  $h$  may be inadequate in MD applications. To address this, the “locally-scaled diffusion map” has been developed by Rohrdanz et al.,<sup>135</sup> where the parameter  $h$  becomes position-dependent. The density adaptive diffusion maps approach<sup>136</sup> is a related technique to deal with highly nonuniform data densities.

Diffusion map has been applied to analyze the slow transitions of molecules,<sup>137,138</sup> guide enhanced-sampling methods,<sup>139–141</sup> and have been combined with the kinetic variational principles described in section 6.<sup>142</sup>

**3.2.4. Sketch-Map.** While building a low dimensional representation of a molecular trajectory, we might be interested

in preserving the distances falling within a specific window which is assumed to characterize the important modes of the system under study. This is the main motivation for the introduction of the Sketch-map algorithm.<sup>143</sup> Sketch-map finds the projection  $\mathbf{Y}$  which minimizes the following loss function:

$$L(\mathbf{Y}) = \left( \sum_{i \neq j} w_i w_j \right)^{-1} \sum_{i \neq j} w_i w_j [s_x(R_{ij}) - s_y(\|\mathbf{y}_i - \mathbf{y}_j\|)]^2 \quad (15)$$

The above equation differs from the standard MDS loss function (eq 3) in two ways. First, the parameters  $\mathbf{w}$  are introduced to allow to control the importance of the distance between any two points. Second, the distances in the original space and in the projected space are passed through the sigmoid functions  $s_x$  and  $s_y$  before being compared in the loss. Each sigmoid function depends on the three parameters,  $\sigma$ ,  $a$ , and  $b$ :

$$s(d) = 1 - (1 + (2^{a/b} - 1)(d/\sigma)^a)^{-b/a} \quad (16)$$

The parameter  $\sigma$  represents the transition point of the sigmoid ( $s(\sigma) = 1/2$ ), and it should be chosen as the typical distance that is deemed to be preserved. The parameters  $a$  and  $b$  determine the rate at which the functions approach 0 and 1, respectively.

With a careful choice of these parameters, the Sketch-map projection will depend very weakly on distances that are too small or too large, since these will always be squashed either to 0 or to 1, thus giving little or no contribution to the loss of eq 15.

Contrary to the other projection methods described so far, Sketch-map requires solving a highly nonconvex optimization problem (eq 15). In practice, to find a sensible solution, one is forced to use a combination of heuristics,<sup>143</sup> and in general, there are no guarantees on the computation time needed to find a sufficiently good solution.

The advantage of Sketch-map is that, with a proper selection of the parameters in eq 16, it enables the extraction of relevant structures from a trajectory even when simpler methods fail. However, an adequate choice of parameters can require a lengthy trial-and-error operation, which can be particularly challenging given the absence of guarantees on the time complexity of solving eq 15. Sketch-map has been successfully applied for visualization of free energy surfaces,<sup>143</sup> biasing of molecular dynamics simulations,<sup>144</sup> building “atlases” of molecular or materials databases,<sup>145</sup> and for structure–property prediction.<sup>146</sup>

In the following sections, we describe an approach which allows finding a low dimensional representation of the data working on a very different premise: that relative distances between points are harvested from a stochastic process. This method, like Sketch-map, also works if the data manifold is not isomorphic to a hyperplane.

### 3.2.5. t-Distributed Stochastic Neighbor Embedding.

The  $t$ -distributed stochastic neighbor embedding<sup>147</sup> (henceforth “t-SNE”) performs dimensionality reduction on high-dimensional data sets following a different route with respect to the approaches discussed so far. The underlying idea (already present in the original Stochastic neighbor embedding (SNE)<sup>148</sup>) is to estimate, from the distances in the high-dimensional space, the probability of each point to be a neighbor of each other point. Then, the algorithm goal is to obtain a set of projected coordinates in which these “neighborhood probabilities” are as similar as possible to the ones in the original space.

The probability that point  $j$  is a neighbor of  $i$  is estimated as

$$P_{ij} = \frac{K_{ij}}{\sum_{k \neq i} K_{ik}} \quad (17)$$

where  $K_{ij}$  is a Gaussian kernel. In this approach, contrary to what happens in the diffusion map (see eq 10), the length scale parameter  $h_i$  is chosen independently for each point  $i$  by setting

$$2^{-\sum_j P_{ij} \log_2 P_{ij}} = \text{Perp} \quad (18)$$

where Perp is a free parameter called “perplexity”, which roughly represents the number of nearest neighbors whose probabilities are preserved by the projection. Adaptively changing the length scale to match a given perplexity, hence, allows the method to preserve smaller length scales in denser regions of the data set. These probabilities  $P_{ij}$  are then transformed into a joint distribution by symmetrizing the matrix:

$$\bar{P}_{ij} = \frac{P_{ij} + P_{ji}}{2N} \quad (19)$$

The neighborhood probabilities in the original space defined in this manner are then transferred to the projected space. For doing this, one needs to choose a parametric form for this probability distribution. At variance with the original SNE implementation, in which a Gaussian form is assumed, in t-SNE, a Student's  $t$ -distribution with one degree of freedom is employed:

$$\bar{Q}_{ij} = \frac{(1 + \|\mathbf{y}_i - \mathbf{y}_j\|^2)^{-1}}{\sum_{k \neq i} (1 + \|\mathbf{y}_i - \mathbf{y}_k\|^2)^{-1}} \quad (20)$$

The shape of this distribution is chosen in such a way that it mitigates the so-called “crowding problem”,<sup>147</sup> namely the tendency to superimpose points when projecting a high-dimensional data set onto a space of lower dimensionality. Indeed, the heavy tails associated with this distribution allow relaxing the constraints in the distances at the projected space, therefore allowing to a less “crowded” representation.

The two distributions are compared by measuring their Kullback–Leibler (KL) divergence:<sup>149</sup>

$$\text{KL}(\bar{\mathbf{P}} \parallel \bar{\mathbf{Q}}) = \sum_{j \neq i} \bar{P}_{ij} \log \frac{\bar{P}_{ij}}{\bar{Q}_{ij}} \quad (21)$$

The feature vectors  $\mathbf{y}$  entering in eq 20 are initialized (randomly in the original formulation, other possible schemes have been proposed<sup>150</sup>) and iteratively modified to minimize the loss function  $\text{KL}(\bar{\mathbf{P}} \parallel \bar{\mathbf{Q}})$  with a steepest descent algorithm.

The t-SNE loss function eq 21 is nonconvex, making its optimization difficult. In particular, if one uses steepest descent the low-dimensional embedding can differ significantly in runs performed with different initial conditions. This makes t-SNE a method that is, in principle, very powerful and flexible but difficult to use in practice.

Recently, the t-SNE method has been adapted to better fit the needs of molecular simulations. In ref 151, the authors propose a time-lagged version of the method, in the same spirit of TICA (see section 6.1). However, as the authors comment, the time-lagged version distorts the densities from the ones in the original space, which makes the method inappropriate for computing free energies. In ref 152, it is claimed that t-SNE provides a dimensionality reduction which minimizes the information loss and can be used for describing a multimodal free energy surface

of a model allosteric protein system (Vivid). The same authors further employ this approach to obtain a kinetic model.<sup>153</sup>

**3.2.6. Deep Manifold Learning Methods.** Deep learning methods are now frequently used to learn nonlinear low-dimensional manifolds embedding high-dimensional data, and these techniques are also starting to be adopted for analysis and enhanced sampling of molecular simulations.

A popular deep dimensionality reduction method is the autoencoder.<sup>154</sup> Autoencoders work by mapping input configurations  $\mathbf{x}$  through an encoder network  $E$  to a lower-dimensional latent space representation  $\mathbf{y}$ , and mapping this back to the original space through a decoder network  $D$

$$\mathbf{y} = E(\mathbf{x}) \quad (22)$$

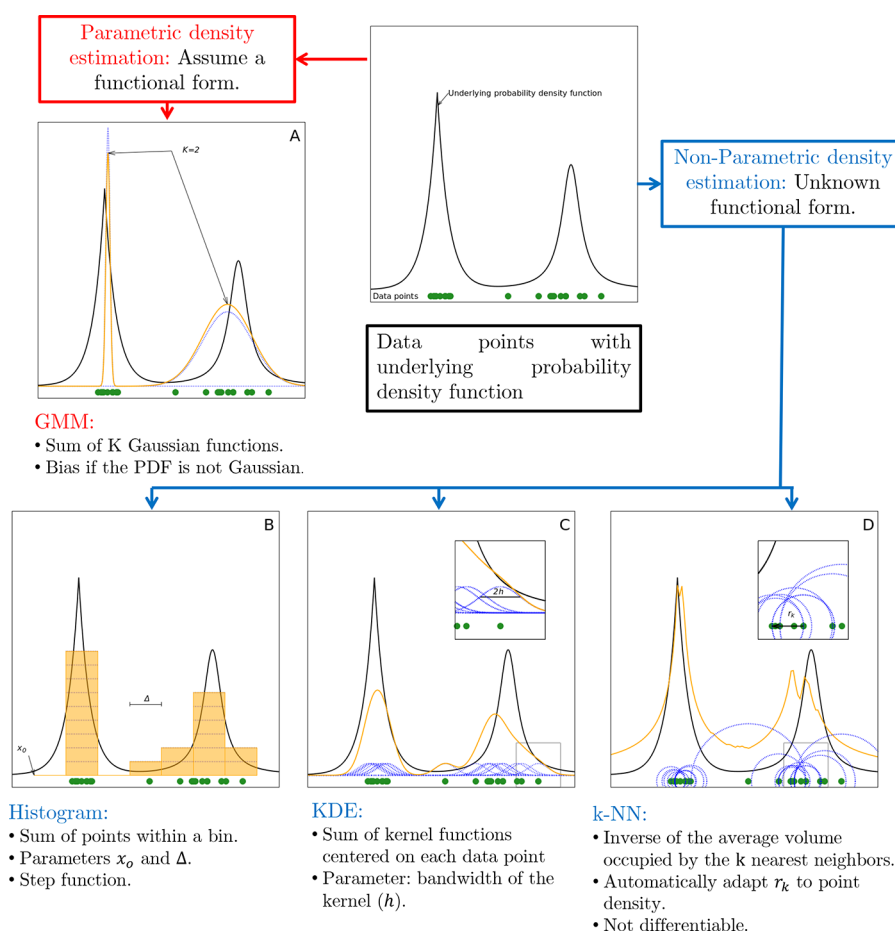
$$\bar{\mathbf{x}} = D(\mathbf{y}) \quad (23)$$

The network learns a low-dimensional representation  $\mathbf{y}$  by minimizing the error between the original data points  $\mathbf{x}_i$  and the reconstructed data points  $\bar{\mathbf{x}}_i$ . Note that if  $E$  and  $D$  are chosen to be linear maps rather than nonlinear neural networks, then the optimal solution can be found analytically by PCA (section 3.1.1): indeed, the encoder  $E$  is given by the matrix of selected eigenvectors, and the decoder  $D$  is given by its transpose.

More involved deep learning approaches to obtain a low-dimensional latent space representation are generative neural networks. Key examples include Variational Autoencoders (VAEs)<sup>155</sup> and Generative Adversarial Networks (GANs).<sup>156</sup> VAEs are structurally similar to autoencoders, but involve a sampling step in the latent space that is required in order to draw samples from the conditional probability distribution of the underdetermined high-dimensional state given the latent space representation,  $p(\mathbf{x}|\mathbf{y})$ . VAEs and other neural network architectures have been widely employed to extract nonlinear reaction coordinates and multidimensional manifolds for molecular simulation.<sup>157–170</sup> The first goal of these approaches is to aid the understanding of the structural mechanisms associated with rare events or transitions. See section 6 for a deeper discussion of variationally optimal approaches to identify rare-event transitions. Second, the low-dimensional representation of the “essential dynamics” learned by these methods is poised to serve the enhance sampling of events that are rare and particularly important to compute certain observable of interest. This has been approached by constructing biased dynamics in the learned manifold space and reweighting the resulting simulations to obtain equilibrium thermodynamic quantities,<sup>162,164–166</sup> as well as by selecting starting points for unbiased MD simulations which can then be similarly reweighted to obtain equilibrium dynamical quantities.<sup>160,161,163,171</sup>

GANs implicitly learn a low-dimensional latent space representation in the input of a generator network. Classical GANs are trained by playing a zero-sum game between the generator and a discriminator network, where the generator tries to fool the discriminator with fake samples and the discriminator tries to predict whether its input came from the generator or was a real sample from a database. In the molecular sciences, GANs have (so far) primarily been used for sampling across chemical space; e.g., to aid the optimization of small molecules with respect to certain properties.

For completeness, we mention that there is a third very popular class of generative neural networks called normalizing flows,<sup>172</sup> which have recently been combined with statistical mechanics in the method of Boltzmann generators,<sup>43</sup> which facilitate rare-event sampling of molecular systems. However,



**Figure 5.** Graphical summary of the density estimation methods. Different density estimates on 20 1D data points (green) extracted from a mixture of a Laplace and a Cauchy distribution (black line).

flows are invertible neural networks and are thereby not inherently performing any dimensionality reduction but rather a variable transformation to a space from which is it easier to sample, but still of the same dimension as the original input. Recently, it has been proposed to combine flows with renormalization group theory to gradually marginalize out some of the dimensions in each neural network layer.<sup>173</sup> This is a promising direction for combining the tasks of dimensionality reduction and rare-event sampling in the molecular sciences.

#### 4. DENSITY ESTIMATION

After the configurations have been projected into a low dimensional space, one can use the representation in this space for estimating the probability density function of the data set. Furthermore, if the representation is of dimension smaller than three, one can directly visualize the probability density  $\rho$ . Equivalently, one can also visualize its logarithm which, if the simulation is performed in the canonical ensemble at temperature  $T$ , is equal to the free energy; that is,  $F(\mathbf{y}) = -k_B T \log(\rho(\mathbf{y}))$ .

Density estimation is of great interest well beyond molecular simulations, and is a key working tool in unsupervised machine learning. Density estimation consists of estimating the underlying probability density function (PDF) from which a given data set has been drawn. Although its main applications relate to data visualization (as in the case of visualizing the free energy surface), it is also part of the pipeline of other analysis methods

such as kernel regression,<sup>174</sup> anomaly detection,<sup>175</sup> and clustering (see Section 5).

The natural connection between density estimation and free energy reconstruction can be exploited to increase the efficiency of simulation analyses. For example, many kinetic analysis methods rely on density estimation<sup>176–180</sup> (see Section 6).

In the case of equilibrium molecular systems, the PDF as a function of a feature vector  $\mathbf{y}$  is in principle known exactly. In the canonical ensemble, this is given by,

$$\rho(\mathbf{y}) \propto \int d\mathbf{x} \exp\left(-\frac{V(\mathbf{x})}{k_B T}\right) \delta(\mathbf{y} - \mathcal{Y}(\mathbf{x})) \quad (24)$$

where  $\mathcal{Y}(\mathbf{x})$  is the function which enables the computation of the feature vector  $\mathbf{y}$  as a function of the coordinates  $\mathbf{x}$ . In practice, however, calculating this integral is not possible, and the resulting PDF can only be estimated approximately following one of the procedures described in this section.

Density estimation methods can be grouped into two categories, namely, parametric and nonparametric methods (see Figure 5). In the first one, a specific functional form for the PDF is chosen and its free parameters are inferred from the data. In contrast, nonparametric methods attempt to describe the PDF without making a strong assumption about its form.<sup>181</sup>

The choice between parametric and nonparametric methods is heavily problem dependent. As a general rule, when the origin of the data permits a reasonable hypothesis for the functional form of the PDF, parametric density estimation should be



preferred, since it quickly converges with relatively few data points. If the functional form is unknown, however, it is often better to sacrifice performance than to risk introducing bias into the estimation.

#### 4.1. Parametric Density Estimation

In parametric density estimation a fixed functional form of the PDF is assumed and one estimates its parameters from the data. For example, if one assumes that the data are sampled from a Gaussian distribution one can then estimate the density by simply computing mean and variance of the data. Such a procedure is common throughout various scientific branches. However, this procedure can lead to an error in the estimate that cannot be reduced by increasing the number of observations, since the error can be brought to zero only if the data points are truly generated from a Gaussian distribution. To add flexibility to this approach and, therefore, alleviate the bias inherent in assuming the form of the distribution a priori, a common technique is to model the PDF as a mixture of  $K$  distributions

$$\rho(\mathbf{y}) = \sum_l^K \pi_l \psi(\mathbf{y}; \{\theta_l\})$$

$$\pi_l \geq 0, \sum_l^K \pi_l = 1 \quad (25)$$

where  $\psi(\mathbf{y}; \{\theta_l\})$  is a function that depends upon the set of parameters  $\theta_l$  and  $\pi_l$  is the weight of this function in the estimate. A common choice for  $\psi$  is a Gaussian, leading to the Gaussian mixture model (GMMs).<sup>182</sup>

The parameters  $\{\theta_l, \pi_l\}$  can be estimated by a maximum likelihood method, namely

$$\pi_1, \theta_1, \dots, \pi_K, \theta_K = \operatorname{argmax} \sum_i^N \log \left( \sum_l^K \pi_l \psi(\mathbf{y}_i; \theta_l) \right) \quad (26)$$

where the sum over  $i$  runs over the  $N$  observations  $\mathbf{y}_i, i = 1, \dots, N$ . This likelihood function can then be optimized through (for example) an expectation-maximization scheme.<sup>182</sup>

A general problem of GMMs is that, since the likelihood to be optimized is a nonlinear function of the parameters, finding its global maximum is typically very difficult. Another critical issue in this approach is the choice of the hyperparameter  $K$ , that is, the number of functions used in the mixture. There is no straightforward relationship between the quality of a density estimate and  $K$  because the best choice strongly depends on the shape of the PDF. The choice of  $K$  is a model selection problem,<sup>183</sup> which can be approached by maximizing the likelihood on a validation data set.<sup>184</sup> Alternative Bayesian approaches for this problem involve maximizing the marginal likelihood with respect to  $K$  or using a Dirichlet process prior distribution for the hyperparameter  $K$ .<sup>185</sup>

There are numerous applications of mixture models in the analysis of molecular simulation. In refs.,<sup>186,187</sup> the free energy surfaces of biomolecules are reconstructed as a sum of Gaussian functions. In refs.,<sup>188,189</sup> GMMs are employed as a basis for an enhanced sampling algorithm in a way that follows the spirit of metadynamics<sup>190</sup> but in which the bias potential is updated to be the sum of few Gaussian functions. In this case, the number of basis functions can be adjusted by using variationally enhanced sampling.<sup>191</sup> Moreover, in ref.,<sup>180</sup> GMMs are used as basis for the MD propagator in a kinetic model (see section 6), while in

ref 192, they are used for atomic position reconstruction from coarse-grained models.

#### 4.2. Nonparametric Density Estimation

**4.2.1. Histograms.** Nonparametric methods avoid making strong assumptions on the functional form of the PDF underlying the data. The most popular nonparametric method, especially in the molecular simulation community, is the histogram.<sup>193–197</sup>

In this method, the space of the data is divided into bins and the PDF is estimated by counting the number of data points within each bin. In one dimension, denoting the center of bin  $I$  as  $y_I$ , and taking  $y_I = y_0 + I\Delta$ , where  $\Delta$  is the bin width, we have

$$\rho(y_I) \sim \frac{n_I}{N \cdot \Delta} \quad (27)$$

where  $n_I$  is the number of configurations within bin  $I$ , and can be computed as  $n_I = \sum_i \chi_{[y_I - \frac{\Delta}{2}, y_I + \frac{\Delta}{2}]}(\mathbf{y}_i)$ , where  $\chi_{[a,b]}(\mathbf{y}) = 1$  if  $\mathbf{y} \in [a, b]$  and 0 otherwise.

Under the assumption that the different observations are independent,  $n_I$  is sampled from a binomial distribution  $n_I \sim B(N, p_I)$  where (in a one-dimensional feature space)  $p_I = \int_{y_I - \Delta/2}^{y_I + \Delta/2} d\mathbf{y} \rho(\mathbf{y})$  is the probability of observing a configuration in the bin. The expected value of  $n_I/N$  is equal to  $p_I$ , which implies that the estimator in eq 27 is correct only if  $\int_{y_I - \Delta/2}^{y_I + \Delta/2} d\mathbf{y} \rho(\mathbf{y})$  can be approximated with  $\Delta \cdot \rho(y_I)$ . If  $\Delta$  is too large, this approximation is not valid and the density estimation becomes too coarse-grained, thereby inducing a systematic error, which is referred to as *bias*.

The variance of  $n_I$  is  $\operatorname{Var}(n_I) = N p_I (1 - p_I)$ , and the statistical error on the density estimate in eq 27 can be estimated as

$$\epsilon(\rho(y_I)) = \frac{\sqrt{\operatorname{Var}(n_I)}}{N \Delta} \quad (28)$$

$$\sim \frac{\rho(y_I)}{\sqrt{n_I}} \quad (29)$$

where to go from the first to the second line we used the fact that  $p_I \ll 1$  and  $p_I \cong \rho(y_I)\Delta$ . If one chooses a value of  $\Delta$  that is too small, the number of configurations within the bin  $n_I$  becomes small, and the resulting error, commonly referred to as *variance*, becomes large. In practice, the value of  $\Delta$  can be chosen by considering the so-called bias/variance trade-off in which both small variance and small bias are desired, but decreasing one often increases the other.<sup>181</sup> This trade-off is common to all nonparametric density estimators.

Histograms are typically used to estimate the density for  $d \leq 3$  since in higher dimensions the estimator becomes noisy since an increasing number of bins will be either empty or only visited a few times. This problem can be alleviated only if one exponentially increases the number of data points with  $d$ : a manifestation of the curse of dimensionality.<sup>198</sup> Finally, another drawback of the histogram is that its estimate of the PDF is not differentiable.

**4.2.2. Kernel Density Estimation.** Kernel density estimation (KDE) partially overcomes the problems associated with histogramming. KDE approximates the PDF of a data set as a sum of kernel functions centered at each data point. In one dimension, the approximation reads

$$\rho(y) \sim \frac{1}{N \cdot h} \sum_i^N \kappa\left(\frac{y - y_i}{h}\right) \quad (30)$$

where the kernel function  $\kappa$  is chosen as a unimodal probability density symmetric around 0. Particularly, it satisfies the properties

$$\kappa(y) \geq 0, \kappa(y) = \kappa(-y), \int_{-\infty}^{\infty} \kappa(y) dy = 1 \quad (31)$$

The estimator depends on the kernel function of choice and the hyperparameter  $h > 0$  (i.e., the bandwidth). Some popular examples of kernels are summarized in Table 1.

**Table 1. Some Widely Used Kernels for Density Estimation**

uniform	$\kappa(y) = \frac{\chi_{[-1,1]}(y)}{2}$
triangle	$\kappa(y) = \chi_{[-1,1]}(y)(1 -  y )$
Epanechnikov	$\kappa(y) = \chi_{[-1,1]}(y) \frac{3}{4}(1 - y^2)$
cosine	$\kappa(y) = \chi_{[-1,1]}(y) \frac{\pi}{4} \cos\left(\frac{\pi}{2}y\right)$
Gaussian	$\kappa(y) = \frac{1}{\sqrt{2\pi}} \exp\left(-\frac{1}{2}y^2\right)$

The Gaussian kernel is arguably the most used. The Epanechnikov kernel is optimal in the sense that it minimizes the mean integrated squared error (MISE).<sup>199</sup> However, experience has shown that the impact of the choice of the kernel on the quality of the estimation is lower than the choice of the bandwidth.<sup>200</sup>

The bandwidth hyperparameter  $h$  plays a role similar to the bin width parameter  $\Delta$  in the histogram method.  $h$  is also referred to as the “smoothing” parameter; since the larger it is, the smoother the resulting PDF estimate is. However, smoothing too much can lead to artificially delete important features on the PDF. The dependence of the MISE on  $h$  can be decomposed into two terms, namely, the *bias*, which scales as  $h^2$ , and the *variance*, which corresponds to the error induced by the statistical fluctuations in the sampling and scales as  $\frac{1}{h^d}$ .<sup>201</sup>

As in the case of the histogram, the optimal value of  $h$  should be chosen as a trade-off between the bias and the variance terms. Much research has been focused on the optimal choice of  $h$  in eq 30.<sup>202–204</sup>

Alternatively, the bandwidth selection problem can be addressed by introducing an adaptive kernel with a smoothing parameter that varies for different data points.<sup>205,206</sup> A position-dependent bandwidth can be obtained by optimizing a global measure of discrepancy of the density estimation from the true density.<sup>181,207</sup> However, this global measure is typically a complex nonlinear function,<sup>208</sup> and its optimization can be prohibitive for large data sets. A more feasible approach, first proposed by Lepskii,<sup>209</sup> is to adapt the bandwidth to the data locally. This approach has been further developed by Spokoiny, Polzehl, and others.<sup>210–215</sup>

Kernel density estimation can also be used in more than one dimension by employing multivariate kernels. However, the number of hyperparameters increases with the number of dimensions. In the case of the multivariate Gaussian kernel, the parameters can be summarized in a  $d \times d$  symmetric matrix  $\mathbf{H}$ ,

which plays a role analogous to  $h$  in the one-dimensional case. The corresponding estimator is

$$\rho(y) = \frac{1}{N \cdot |\mathbf{H}|^{1/2}} \sum_i^N \kappa((y - y_i)^T \mathbf{H}^{-1}(y - y_i)) \quad (32)$$

In comparison with histograms, kernel density estimation has the advantage that it is differentiable. KDE is gaining increasing popularity in the analysis of molecular dynamics simulations, leading even to the development of specific parameters tailored for MD.<sup>216</sup> In addition to their use in visualization of free energy surfaces<sup>217</sup> and the construction of kinetic models,<sup>177</sup> KDEs have been applied in the study of nonexponential and multidimensional kinetics<sup>218</sup> and the computation of entropy differences.<sup>219</sup>

**4.2.3.  $k$ -Nearest Neighbor Estimator.** Another route for estimating the density of a data set is the  $k$ -nearest neighbor ( $k$ -NN) estimator.<sup>220</sup> In this method, the probability density  $\rho_i = \rho(y_i)$  is estimated as

$$\rho_i \sim \frac{k}{N} \frac{1}{V_d r_k^d(y_i)} \quad (33)$$

where  $V_d$  is the volume of the unitary sphere in dimension  $d$  and  $r_k(y_i)$  is the distance between  $y_i$  and its  $k$ -th nearest neighbor point.

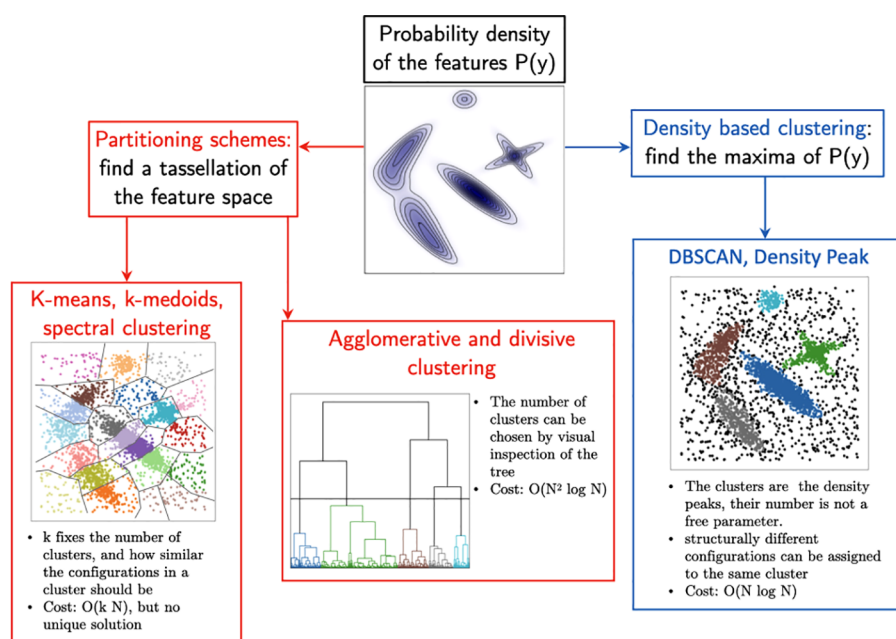
The  $k$ -NN method can be thought of as a special kernel density estimation with local bandwidth selection, where the role played by  $k$  is similar to that of  $h$  in the KDE. The statistical error induced of this estimator is given by<sup>13</sup>

$$\varepsilon(\rho_i) = \frac{\rho_i}{\sqrt{k}} \quad (34)$$

The higher the value of  $k$ , the smaller the variance of the estimator, but the larger the bias, since this error estimate is valid only if the probability density is constant in the hypersphere of radius  $r_k(y_i)$  centered on  $y_i$ . The  $k$ -NN method can in principle be used for estimating the PDF in any dimension  $d$ . However, if  $d = 1$ , the PDF estimated in this manner cannot be normalized since the integral for the whole space of  $1/r$  does not converge.<sup>221</sup>

The  $k$ -NN method is also affected by the curse of dimensionality. Indeed, it can be shown that for any fixed number of data points, the difference between the distance from the  $k$  nearest neighbor ( $r_k$ ) and the distance from the next nearest neighbor ( $r_{k+1}$ ) tends to 0 when  $d \rightarrow \infty$ ,<sup>222</sup> leading to problems in the definition of the density. Much effort has been put into bypassing this limitation.<sup>223</sup> One approach involves avoiding estimating the density in the full feature space and instead estimating it in the manifold that contains the data (which usually has a much lower dimensionality). This approach, first suggested in ref 224, was further developed in ref 13 with a specific focus on the analysis of MD trajectories.

While in molecular systems the configurations are defined by a high-dimensional feature space, restraints induced by the chemical and physical nature of the atoms prevent the system from moving in many directions (for example, in the direction which significantly shortens a covalent bond). In practice, these restraints reduce the dimension of the space to a value which is referred to as an *intrinsic dimension*. The intrinsic dimension can be estimated, for example, using the approaches in refs 225–228. The probability density can then be estimated using eq 33, where the dimension of the feature space  $d$  is replaced with the



**Figure 6.** Cartoon illustration of the different clustering schemes.

intrinsic dimension, which can be orders of magnitude smaller, making the estimate better behaved. This framework also addresses the problem of finding a  $k$  that yields sufficiently small variance and bias. Ref 13 proposes that the largest possible  $k$  for which the probability density can be considered constant (within a given level of confidence) for each data point separately. This optimal  $k$ , which is point-dependent and denoted by  $k_p$ , is then used to estimate the probability density by a likelihood optimization procedure allowing for density variation up to a first-order correction. The estimator of the density then becomes

$$\log(\rho_i) = -\operatorname{argmax}_{F,a} \left( -F\hat{k}_i + a \frac{\hat{k}_i(\hat{k}_i + 1)}{2} + \sum_{l=1}^{\hat{k}_i} v_{i,l} e^{-F+al} \right) \quad (35)$$

where  $v_{i,l} = V_{ID}(r_l^{ID}(\mathbf{y}_i) - r_{l-1}^{ID}(\mathbf{y}_i))$  is the volume of the hyperspherical shell enclosed between the  $l$ th and the  $(l-1)$ th neighbor of the configuration  $\mathbf{y}_i$ . This approach enables the estimation of the probability density directly in the space of the coordinates introduced in section 2 (for example, the dihedral angles) rather than performing a dimensionality reduction with one of the methods described in section 3 in advance or by explicitly defining a collective variable for describing the system.

In the field of MD simulations,  $k$ -NN has been used as part of the pipeline for more complex analyses<sup>229,230</sup> and for computing both entropies<sup>231,232</sup> and free energies.<sup>233,234</sup>

## 5. CLUSTERING

Clustering is a general-purpose data analysis technique in which data points are grouped together based on their similarity according to a suitable measure (for example, a metric). In molecular simulations, the use of clustering is very common, since clusters can be seen as a way of compactly representing a complex multidimensional probability distribution. Clustering, thus, performs an effective dimensionality reduction in a manner which can be seen as complementary to the approaches described in section 3. It is well-known that there is no

problem-agnostic measure of the success or appropriateness of a given clustering algorithm or its results.<sup>235</sup> Thus, it is crucial to choose a clustering algorithm based on what is known about the data set and what is expected about the result. In particular, in the field of molecular simulations, we can conveniently differentiate the following two classes of techniques.

- **Partitioning schemes:** In these approaches, the clusters are groups of configurations which are similar to each other and different from configurations belonging to other clusters. These clusters define a partition (or tessellation) of the space in which the configurations are defined.
- **Density-based schemes:** In these approaches, the clusters correspond to the peaks of the probability distribution from which the data are harvested (or, equivalently, to the free energy minima). Data points belonging to different clusters are not necessarily far apart if they are separated by a region where the probability density is low (see Figure 6).

The most striking difference between these two distinct approaches emerges if one attempts to cluster a set of data harvested from a uniform probability distribution. Using a partitioning scheme, one can find any number of clusters depending on the chosen level of resolution. On the other hand, using a density based scheme, one will obtain a single cluster. Clearly, which approach one should employ strongly depends upon the purpose of the analysis. For instance, if one wants to find directly the metastable states from a cluster analysis of the system, one should use a density-based clustering approach. If, instead, one wants to find an appropriate basis to represent the dynamics, as in kinetic modeling methods (see section 6), a partitioning scheme may be more appropriate, since these schemes allow one to control how similar the configuration assigned to the same cluster are.

### 5.1. Partitioning Schemes

Partition-based algorithms aim to classify the configurations in sets (i.e., clusters) that include only similar configurations. These algorithm can be further divided into two classes.



- **Centroid-based/Voronoi tessellation algorithms:** In these methods, the number of clusters is determined by a specific parameter, which can be a cutoff specifying the maximum allowed distance between two configurations assigned to the same cluster or, alternatively, the number of clusters. The well-known *k*-means and *k*-medoids algorithms belong to this class, as well as the faster but less optimal *leader algorithms* *k*-centers and regular-space clustering. In all of these methods, each cluster is associated with a so-called centroid, which is a configuration representing the content of the cluster. This centroid induces a so-called *Voronoi*-tessellation, which divides space such that each point is associated with the nearest centroid, in the chosen distance metric.
- **Hierarchical/agglomerative and divisive clustering:** Here, the choice of the number of clusters is deferred, being possible to run the algorithm without setting it in advance. A (usually binary) tree is constructed according to a linkage criterion, and the number of clusters can be selected by appropriately “cutting” the tree, usually by visual inspection and taking into account the scope of the analysis. These approaches are commonly referred to as hierarchical clustering, which can be “agglomerative” or “divisive” depending on whether the data points are first considered to be individual clusters or members of a single cluster, respectively.

**5.1.1. *k*-Means and *k*-Medoids.** Arguably, the most popular clustering schemes in molecular simulations and beyond are the centroid-based methods *k*-means<sup>236</sup> and *k*-medoids.<sup>237</sup> In both cases, the number of clusters *k* is chosen before performing the algorithm.

In *k*-means, also known as Lloyd’s algorithm,<sup>236</sup> the cluster centers are *k* configurations whose position  $\mathbf{x}_{c,i}$ ,  $i = 1, \dots, k$  correspond to the *mean* of the coordinates (or the features) of all the cluster members. For a given choice of *k* cluster centers, the quality of the clustering is defined by a loss  $L(\mathbf{x}_c)$ . This loss is defined as the sum of the square of the distances from each configuration in the data set to the cluster center to which the configuration is assigned. Therefore, the *k*-means clustering problem is formulated as an optimization problem in which the best solution (i.e., set of cluster centers) is the minimum of  $L(\mathbf{x}_c)$  with respect to  $\mathbf{x}_c$ . This optimization is known to be NP-hard.<sup>238</sup> To find an approximate solution, ref.<sup>236</sup> proposes the following iterative procedure:

1. *k* members of the data set are randomly chosen as centers.
2. Each configuration in the data set is assigned to its closest center.
3. The centers are recomputed according to the new assignments.
4. Points 2 and 3 are repeated until the cluster membership does not change.

Because of the glassy nature of  $L(\mathbf{x}_c)$ , the outcome of the algorithm is heavily dependent on the initialization step. Thus, it is often necessary to repeat the procedure above with different initial centers to obtain reasonable results. In light of this drawback, initialization schemes have been proposed that improve the tractability of the problem by speeding up convergence.<sup>239</sup> A “minibatch” *k*-means version has also been proposed, which alleviates the problem of trapping in local minima and it is faster than the original method (in which all the configurations are simultaneously used in the optimization).<sup>240</sup> In a closely related approach, called fuzzy *c*-means,<sup>241,242</sup> each

point does not belong to a single cluster, but rather its degree of membership to all possible clusters is represented by a vector  $\mathbf{u}$  with *k* components such that  $\sum_{i=1}^k u_i = 1$ . *k*-means scales as  $O(Nki)$  in the number of configurations *N* and the number of iterations *i*. The number of iterations *i* needed to achieve convergence depends on how the data is distributed and will often depend on *N*, but in practice, *k*-means is often terminated when a fixed number of iterations has been reached.

In *k*-means, the cluster centers in the first step are configurations in the data set, but in the following steps their positions are adjusted, and (thus), they will no longer correspond to configurations in the original data set. This means that *k*-means is applicable if an explicit feature representation is given, but not if only a distance metric is defined (e.g., when a pairwise RMSD minimal distance is used). The *k*-medoids algorithm<sup>237</sup> ensures that each cluster centroid always corresponds to a configuration in the data set, offering an alternative. In this approach, each cluster center in a given iteration is chosen as the configuration in the original data set which minimizes the sum of the distances from the cluster members. A minibatch strategy can also be employed in *k*-medoids. With sufficient data, *k*-means and *k*-medoids are expected to give qualitatively similar results. The key advantage of the latter is interpretability; for example, in a molecular simulation application each *k*-medoids cluster center can be easily visualized as a configuration present in the raw data set. Additional advantages of *k*-medoids are that it is less sensitive to the presence of outliers and that it can be used with a distance measure between pairs of coordinates other than Euclidean distance. The *k*-medoids algorithms scales as  $O(N^2)$ , and it is hence difficult to use for large data sets.<sup>243</sup> In both *k*-means and *k*-medoids, the choice of *k* is an open problem. A common practice consists in running the algorithm with increasing *k* values, and then plot the loss as a function of *k* (the so-called scree plot<sup>244</sup>) looking for an elbow. Other validation indexes, like the Silhouette<sup>245</sup> can be also employed to this end.

**5.1.2. Leader Algorithms: *k*-Centers and Regular-Space Clustering.** As *k*-means and *k*-medoids may require many passes through the data set, they can become extremely expensive for large data sets. In his book,<sup>246</sup> John Hartigan proposed a *leader algorithm* in which it is necessary to iterate over the data only twice: the first time to assign *k* centroids and the second time to assign all data points to centroids. This results in a fast runtime of  $O(Nk)$  for *N* configurations. The most popular algorithm based on this idea is *k*-centers.<sup>247</sup> The objective of *k*-centers is to maximize the distance between cluster centers. To employ the algorithm, the first cluster center is chosen randomly from the data set. Then, the next cluster center is chosen as the configuration in the data set that is farthest from the previously chosen configuration according to a distance measure. This procedure is iterated until the desired number of centers have been obtained. In the second pass, each configuration in the data set is assigned to the cluster defined by the closest centroid. *k*-centers scales as  $O(Nk)$  for *N* configurations.<sup>247</sup> In the absence of “ties”, *k*-centers is deterministic after the selection of the initial cluster center, but it depends on the order in which the data points are traversed, which may be arbitrary in many applications. For large data sets, the results are approximately independent of these choices. It is important to consider whether the *k*-centers objective is appropriate for the application at hand, since by definition it will choose outlying configurations as cluster

centers,<sup>38</sup> and this may not be desired. Subsampling the data<sup>20</sup> or using a hybrid clustering scheme<sup>248</sup> may mitigate this effect.

Another variant of the leader algorithm, which is equally fast but less sensitive to outliers is *regular space clustering*.<sup>21</sup> Here, instead of fixing the number of clusters, one fixes a minimal distance cutoff  $d_c$ . The first cluster center is chosen at random. Then one cycles through the data set, accepting a new data point as a cluster center when its distance to all existing cluster centers is greater than  $d_c$ .

Another similar partitioning procedure was introduced by Daura et al.<sup>249</sup> In this approach, one also chooses a cutoff distance  $d_c$  under the assumption that configurations which are closer than  $d_c$  are similar enough to be assigned to the same cluster. For each configuration  $i$ , one then estimates the number  $n_i$  of other configurations within  $d_c$ . The first cluster center is the configuration with the highest  $n_i$ ; namely, with more neighbors within a distance  $d_c$ . All the configurations within  $d_c$  from the first center form the first cluster. The second cluster center is then the configuration with the highest  $n_i$  after excluding the configurations are already assigned to the first cluster. The second cluster is formed by all the configurations within  $d_c$  from the second center that do not belong to the first cluster. This procedure is iterated until all the configurations are assigned to a cluster. Its computational cost scales as  $O(N^2)$  for  $N$  configurations, which can be reduced to  $O(N \log(N))$  by using a smart neighbor search algorithm. This procedure is deterministic: given a set of configurations and a cutoff distance  $d_c$  it produces a unique clustering partition, except if for two data points the number of neighbors  $n_i$  is equal.

Since (like  $k$ -medoids) leader algorithms use available data points as centroids, they are compatible with clustering scenarios, where only a pairwise distance metric is given but no explicit feature representation. In early developments of kinetic models for molecular simulation data (see section 6); when pairwise RMSD was used,  $k$ -centers and regular space clustering were frequently employed for clustering steps of kinetic modeling algorithms.<sup>20,21,250–253</sup> Other—and especially more recent—applications are mostly based on  $k$ -means or its minibatch variant<sup>29,40,254–258</sup> and, less frequently,  $k$ -medoids<sup>259</sup>.

**5.1.3. Spectral clustering.** One of the drawbacks of the approaches described in the previous sections is that they all rely on the analysis of the distances between pairs of configurations. Such distances, if computed using all the coordinates, can be affected by the noise induced by the high dimensionality. Moreover, the need to deal with distances does not allow the use of other similarity measures that, for instance, do not respect the triangular inequality.

Spectral clustering addresses the problem of clustering by an approach that does not require computing distances. It uses a set of pairwise similarities to define a weighted undirected graph, in which each data point corresponds to a vertex and the edges connecting two vertices  $i$  and  $j$  are associated with a weight  $W_{ij}$ . For convenience, one can organize these weights in a matrix  $\mathbf{W}$  and define the diagonal degree matrix  $\mathbf{D}$  as  $D_{ii} = \sum_j W_{ij}$  and the graph Laplacian matrix as  $\mathbf{L} = \mathbf{D} - \mathbf{W}$ .

The transformation of the pairwise similarities into a graph can be done in three ways: (1) by connecting all points whose similarities are equal or greater than a given threshold, (2) by connecting each point to the  $k$  points which are most similar to it, and (3) by treating the similarities directly as weighted edges of a fully connected graphs.

Once the graph is built, the method attempts to divide it into clusters in such a way that the edges of the graphs associated with data points belonging to different clusters have small weights, and the edges within a cluster have large weights. One then defines the cost associated with a given partition into  $k$  clusters as

$$\begin{aligned} \text{Cut}(A_1, \dots, A_k) &= \frac{1}{2} \sum_{l=1}^k \text{Cut}(A_l, \bar{A}_l) \\ &= \frac{1}{2} \sum_{l=1}^k \sum_{i \in A_l, j \in \bar{A}_l} W_{ij} \end{aligned} \quad (36)$$

where  $\text{Cut}(A_l, \bar{A}_l)$  defines the cost associated with dividing the graph into a set  $A_l$  and its complement  $\bar{A}_l$  (the elements not belonging to  $A_l$ ). The direct application of this principle is referred to as the “min-cut” approach,<sup>260</sup> which is found empirically to produce imbalanced partitions<sup>261</sup> since in many cases generates clusters with a single element. This problem can be addressed by redefining the cost function taking into account the size of the clusters. This gives rise to a BalancedCut cost function

$$\text{BalancedCut}(A_1, \dots, A_k) = \frac{1}{2} \sum_{l=1}^k \frac{\text{Cut}(A_l, \bar{A}_l)}{\text{size}(A_l)} \quad (37)$$

Two successful ways exist to define the size of the clusters within the BalancedCut objective function. In the RatioCut approach,<sup>262</sup> the size of a cluster  $\text{size}(A_l)$  is simply measured by the number of vertices in the cluster, while in the normalized cut (Ncut) approach<sup>263</sup> the size of the cluster is measured by the cumulative connectivity of all data points belonging to the cluster  $\text{size}(A_l) = \sum_{i \in A_l} D_{ii}$ .

To find the cluster partition that minimizes this quantity one defines  $\mathbf{H}$  as a  $N \times K$  indicator matrix, where  $H_{il}$  has a discrete positive value if the element  $i$  belongs to the clusters  $l$  and zero otherwise. It can be shown that minimizing the balanced cut can be reformulated as a minimization the trace of the matrix  $\mathbf{H}^T \mathbf{H} \mathbf{L}$  with the constraint  $\mathbf{H}^T \mathbf{H} = \mathbf{I}$  (in the RatioCut case) or  $\mathbf{H}^T \mathbf{D} \mathbf{H} = \mathbf{I}$  (in the Ncut case). The latter case can be rewritten as the minimization of the trace of  $\tilde{\mathbf{H}}^T \tilde{\mathbf{L}} \tilde{\mathbf{H}}$  with the constraining  $\tilde{\mathbf{H}}^T \tilde{\mathbf{H}} = \mathbf{I}$ , by defining a normalized indicator function as  $\tilde{\mathbf{H}} = \mathbf{D}^{-1/2} \mathbf{H}$  and a normalized Laplacian as  $\tilde{\mathbf{L}} = \mathbf{D}^{-1/2} \mathbf{L} \mathbf{D}^{-1/2}$ . Unfortunately, both problems are NP-hard because of the discreteness of the values of  $\mathbf{H}$ . In spectral clustering, this condition is relaxed, allowing  $\mathbf{H}$  to take arbitrary real values. In this manner the minimization of the trace can be handled as an eigenproblem, and the trace is minimized by a matrix  $\mathbf{H}$  composed of the first  $k$  eigenvectors of  $\mathbf{L}$ . However, the relaxation of the discreteness condition implies that the columns of  $\mathbf{H}$  do not provide indicator functions but a continuous vector space. Therefore, the final step of a spectral clustering algorithm involves clustering the configurations within the  $\mathbf{H}$  space using a  $k$ -means algorithm (see section 5.1.1).

To summarize, the spectral clustering algorithm consists of (1) building the weighted graph from the similarities/distance matrix, (2) computing the (normalized) graph Laplacian and obtain its first  $k$  eigenvectors, and (3) using these eigenvectors as input for a  $k$ -means clustering step. The value of  $k$  can be chosen with the same criteria described in section 3.1.1; namely, if a gap in the eigenvalue spectrum is present, one should choose the value of  $k$  that preserves eigenvectors corresponding to eigenvalues above this gap.

Apart from the graph-based interpretation illustrated in this section, the algorithm has been derived in other frameworks and has many variants, mostly differing in the way in which the graph is generated or the way in which the Laplacian is normalized. In a specific formulation,<sup>264</sup> spectral clustering can be seen as a Kernel PCA (described in section 3.2.2), followed by a *k*-means clustering step; under certain conditions, it has been shown to be equivalent to kernel *k*-means.<sup>265</sup> The interested reader is encouraged to check specific reviews on spectral clustering<sup>261,266</sup> for more details on the various existing approaches.

This clustering procedure is powerful and robust, and it has been widely applied for analyzing molecular dynamics trajectories.<sup>267–269</sup> The graph can be generated, for example, using the RMSD between structures as a similarity measure combined with a squared exponential kernel.<sup>267</sup> However, spectral clustering techniques differ in many details: the way of generating the graph, the ways of normalizing the Laplacian, the way of choosing *k* in the absence of a clear gap. Some recommendations and rules-of-thumb can be found in ref 266, but a careful system-dependent evaluation is typically necessary.

**5.1.4. Hierarchical Clustering.** A significant drawback of *k*-means and related algorithms is the necessity of choosing *k* a priori. In hierarchical clustering, this problem is circumvented by building a tree structure, called a dendrogram, which represents an ensemble of clustering models with every possible *k*. One can then choose the most appropriate partition a posteriori from the dendrogram. Hierarchical clustering approaches require defining a dissimilarity function and a linkage criterion (the latter is sometimes referred to as an objective function). The dissimilarity function does not need to be a metric. In particular, it does not need to satisfy the triangular inequality, which yields great flexibility.

Hierarchical clustering algorithms can be classified into two categories; namely, agglomerative and divisive algorithms. In agglomerative clustering, each configuration is initially assigned to a different cluster. At every iteration two existing clusters are combined, so the next level of the dendrogram has one fewer cluster. In divisive clustering, the data set starts as a single cluster of all *N* configurations, and at each iteration an existing cluster is split into two. Divisive clustering can be computationally intractable because the number of possible divisions scales unfavorably as clustering proceeds; therefore, we restrict the remainder of this overview to agglomerative clustering only (the reader is referred to ref 237 for a detailed discussion of this topic).

Agglomerative clustering is initialized by considering every configuration as a different cluster to create *N* singleton clusters. Then, the closest two configurations are combined, leaving *N* – 1 clusters. Which pair of clusters is linked in each step of the algorithm is decided by a *linkage criterion*, which is different in different algorithms. In single linkage<sup>270</sup> at a given step of the algorithm one links together the two clusters *A* and *B*, which are closer according to the dissimilarity measure  $\min(d(a, b))$  for  $a \in A, b \in B$ . In complete linkage,<sup>271</sup> the dissimilarity measure used to decide which clusters are linked is  $\max(d(a, b))$  for  $a \in A, b \in B$ , and in average linkage<sup>272</sup> is the average value of the distance between the elements of the two clusters. Finally, in Ward linkage,<sup>273</sup> one agglomerates the pair of clusters that leads to minimum increase of the variance of  $d(a, b)$  estimated for the data belonging to the cluster created upon agglomeration. Agglomeration continues until only a single cluster remains.

The dendrogram comprising the clustering model can then be “cut” for any number of clusters  $2 \leq k \leq n$ . The change in the

value of the linkage criterion between levels of the dendrogram can be used to inform where it should be cut<sup>274</sup> (cutting the dendrogram before the sharpest increase is called the “elbow method”). It is clear that the choice of linkage criterion will significantly impact the result; for example, for randomly distributed data in two dimensions, single linkage will create snake-like clusters and complete linkage will create circle-like clusters.<sup>275</sup>

In molecular kinetics (see section 6), agglomerative clustering is less frequently employed than partitioning schemes, such as *k*-means, since the computation of every pairwise dissimilarity scales as  $n^2$ . When agglomerative clustering has been used to build kinetic models, for example, the simulation data to be clustered was first significantly subsampled to accommodate this scaling.<sup>252,276</sup> It was shown in ref 277 in the context of kinetic modeling of molecular simulations (see section 6.3) that agglomerative clustering with Ward’s linkage produces similarly accurate results to clustering with *k*-means; indeed, the Ward objective function can be linked to that of *k*-means.<sup>278</sup>

## 5.2. Density-Based Clustering

In molecular dynamics simulations, the configurations are harvested from a probability distribution. This distribution is often characterized by the presence of relatively isolated probability peaks that typically correspond to metastable states of the system. Density-based clustering algorithms can be used to find these peaks directly. In these approaches, one first estimates the density of each configuration using one of the approaches described in section 4. Then, one looks for the peaks in this density—these peaks define the clusters. Since a probability peak can in principle have any shape (for example, it can be strongly elongated in one direction), the configurations assigned to a cluster are not necessarily similar. Different density-based clustering algorithms differ in their strategies for finding the density peaks.

Possibly the oldest and most famous algorithm for finding the peaks of a density in feature space is the mean-shift approach.<sup>279</sup> The idea at the basis of this algorithm is simple: For each data point, follow the gradient of the density in ascending direction until you arrive at a local maximum. All the points arriving at the same maximum define then a cluster. Thus, starting from a feature vector *y*, one first estimates the direction of the gradient of the density as

$$m(y) = \frac{\sum_i^N \kappa\left(\frac{y - y_i}{h}\right) y_i}{\sum_i^N \kappa\left(\frac{y - y_i}{h}\right)} \quad (38)$$

where  $\kappa\left(\frac{y - y_i}{h}\right)$  is a kernel function, satisfying the conditions in eq 31. If the kernel bandwidth *h* is small and the data points are many, *m* points in the direction of increasing  $\rho$ . Therefore, one can update the feature vector *y* to  $y + m(y)$ , and iterate the procedure until *y* does not change significantly anymore. This approach, very popular in image analysis, has been also applied to the analysis of molecular dynamics trajectories,<sup>280,281</sup> but it is rather computationally expensive since one is forced to run the iterative procedure described above for each point in the data set. Moreover, the approach inherits the problems of kernel density estimation. If the bandwidth *h* is too small, the algorithm tends to converge to spurious maxima, whose existence is only due to undersampling. If, instead, *h* is too large, some relevant maxima can be missed due to excessive smoothing. The approaches described in the following sections partially



overcome these problems and are, in particular, less sensitive to noise and more computationally efficient.

**5.2.1. DBSCAN.** Density-based spatial clustering of applications with noise (DBSCAN)<sup>282</sup> defines clusters as connected regions with density above a threshold surrounded by regions with a density below this threshold. In the original formulation, the density threshold is defined by two parameters: a neighborhood distance ( $h$ ) and the minimum number of configurations within this distance needed for considering a given configuration above the density threshold (MinPts).

In practice, one first estimates the density by counting how many configurations are within the neighborhood defined by  $h$  for each configuration  $i$ . Note that in this method the densities estimated at a given configuration are proportional to those estimated using a uniform kernel with  $h = \epsilon$  (see eq 30). Then, the configurations whose number of neighbors within  $\epsilon$  is greater than or equal to *MinPts* are considered above the threshold and called *core points*. A configuration  $j$  is said to be “directly reachable” from the configuration  $i$  if  $i$  is a core point and  $j$  is within  $\epsilon$  of  $i$ . A point  $j$  is “reachable” from  $i$  if there is a sequence of points  $i_1, \dots, i_n$  with  $i = i_1$  and  $j = i_n$  in which each  $i_{l+1}$  is directly reachable from  $i_l$  for all  $l$ . Two configurations  $i$  and  $j$  are called “density-connected” if there is a configuration  $k$  such that both  $i$  and  $j$  are reachable from  $k$ . It is important to note that while the reachability property is not symmetric (e.g., configuration  $j$  can be reachable from configuration  $i$  without  $i$  being reachable from  $j$ ), the density connection is a symmetric property. Using these definitions, a cluster is defined as a set of configurations that are all density-connected. Configurations that are not core points nor density connected are classified as *noise points*.

Since it is a density based method, DBSCAN does not require one to specify the number of clusters in the data a priori, in contrast to most of the partitioning schemes. Moreover, DBSCAN can find arbitrarily shaped clusters. However, the choice of the proper combination of parameters MinPts and  $\epsilon$  can be difficult when the densities across configurations are not uniform or when hierarchical structures are present.<sup>283</sup> To solve these issues, several variants have been proposed (see ref.<sup>284</sup> for a recent survey). Of particular relevance are OPTICS<sup>285</sup> and HDBSCAN,<sup>286,287</sup> both of which provide a hierarchical view of the cluster structure.

DBSCAN had been employed to find representative structures from MD simulations,<sup>288,289</sup> as well as to find regions characterized by different molecular densities,<sup>290,291</sup> in this case using molecules as data points. HDBSCAN has recently gained attention also in the analysis of MD trajectories,<sup>292–295</sup> mostly due to its capability to reveal hierarchical structures.

**5.2.2. Density Peak Clustering.** Density peak clustering<sup>296</sup> finds the density peaks according to a different procedure than DBSCAN. Density peak clustering is based on the idea that if a configuration is close to a local maximum of the probability density, then it is surrounded by neighbors with lower density—or, equivalently, it is likely to be at a relatively large distance from any configurations with a higher density.

These simple qualitative criteria are implemented as follows. For each configuration, one first estimates the density  $\rho_i$  of configuration  $i$  (using any of the approaches described in section 4). Then, one computes the minimum distance between the configuration  $i$  and any other configuration with higher density

$$\delta_i = \min_{j: \rho_j > \rho_i} R_{ij} \quad (39)$$

where  $R_{ij}$  is the distance between configurations  $i$  and  $j$ . According to eq 39, the value of  $\delta_i$  of the point with highest density remains undefined, so it is assigned to a value higher than the rest by convention. Cluster centers are identified as configurations for which the value of  $\delta_i$  and  $\rho_i$  are both anomalously large. This is because a center is expected to have both a high density and a large distance from configurations with higher density.

To select the centers in practice, it is proposed to plot the value of  $\delta_i$  as a function of  $\rho_i$  for each configuration. In this visualization, the configurations corresponding to density peaks emerge as outliers and can be recognized by the user in an interactive way.

Depending on the application, this interactive step may not be feasible; in this case, an automatic criterion is needed. However, defining a quantitative criterion for automatically choosing the centers according to this qualitative definition is nontrivial. In the original implementation<sup>296</sup> it is proposed to find the number of clusters by a criterion similar to that used in spectral clustering and PCA; namely, one considers the values of  $\gamma_i = \delta_i \rho_i$  sorted in descending order. If a gap is present (say, before  $\gamma_k$ ), set the cluster centers to be all configurations with  $\gamma_i > \gamma_k$ .

Once the cluster centers are determined, the rest of configurations are assigned to the same cluster as their nearest neighbor with higher density.

The original procedure<sup>296</sup> has been successively improved and modified in order to address some of its pitfalls. Faster versions have been generated, both improving the quality of the implementation<sup>297</sup> and making use of a preliminary  $k$ -means clustering step.<sup>298</sup> In ref 299, a “divide-and-conquer” strategy has been proposed to automatically detect the cluster centers. Improvements on the estimation of the density have been addressed with kernel-like<sup>300</sup> and  $k$ -NN-like approaches<sup>301–303</sup> (see section 4).

In ref 304, the authors of the original method adopt a different approach to address many of its drawbacks. In short, the idea is to use the adaptive  $k$ -nearest neighbor estimator introduced in ref 13 for computing the density at each configuration. Then, a configuration is considered a possible density peak if its computed density is the highest within its neighborhood (the neighborhood is automatically defined by the algorithm as  $\hat{k}_i$ ; see section 4.2.3). However, these density peaks may be a result of statistical fluctuation. This is addressed by using the error associated with the density estimation as follows: a density peak is considered “genuine” if the difference between the density at the maximum and the density at the saddle point is greater than  $Z$  times the sum of the errors of both estimations.  $Z$  is the only parameter of the method and is a measure of the statistic significance of the peaks. In this approach, all the possible saddles are first located as density maxima at the borders among peaks. Then, those peaks that do not pass the test of statistic significance are lumped together. This saddle analysis provides as additional feature a *topography* of the data set; namely, a tool that permits the hierarchical relationships between clusters to be considered.

The density peaks method has been optimized for the analysis of MD simulations,<sup>305</sup> and with some modifications, the analysis of enhanced sampling MD simulations.<sup>306</sup> Density peaks has been successfully employed for extracting binding poses from protein simulations<sup>307</sup> and analyzing the type of sites involved in the jump of diffusing mobile ions in solid state simulations.<sup>308</sup>

It has been recently shown<sup>179,296,304,309</sup> that if one uses a density-based clustering approach to find the peaks of a

probability density estimated in a high-dimensional feature space, these peaks correspond very closely to the so-called “Markov states” of a molecular system, which will be introduced and discussed in section 6. The procedure described in these references effectively bypasses two of the steps, which are normally followed in the derivation of a Markovian model, which will be discussed in the next section: the dimensionality reduction from a large feature space  $x$  to a more compact representation  $y$ , and the clustering performed using one of the approaches described in sections 5.1.2 and 5.1.1. The drawback of the density-clustering based procedure is that the Markov states are an output of the clustering, and one can not attempt improving them, following the protocol described below in Section 6. One can only verify a posteriori if the states define a Markov model by estimating a transition probability restricted to these states, and verifying if the implied time scales are independent of the time lag. Exploiting these techniques together with density-clustering is a possibly interesting research line.

## 6. KINETIC MODELS

When running MD simulations, one does not generate statistically independent data points from the equilibrium distribution but rather *trajectories* in which the configurations are time ordered and in general correlated with each other. The information embedded in the time ordering can be exploited to perform an effective and grounded dimensionality reduction, and the resulting model can be used to extract kinetic information, such as transition rates, pathways, and time-correlation functions.

The inclusion of time correlations in the dimensionality reduction techniques described in section 3, leads to time-lagged independent component analysis and related linear methods (section 6.1), whereas the clustering framework described in section 5 leads to Markov state modeling (MSM) (section 6.3). Both methods can be unified under a “variational approach” (section 6.2), a framework, which can also be extended to nonequilibrium simulations (section 6.4) and can be used to obtain neural network representations (section 6.5).

An MD simulation can be formally described by the dynamical operator  $\mathcal{P}(\tau)$  that propagates the probability density of the system at state  $t$ ,  $\rho(\mathbf{x}_{r,t})$ , to that of the system at time  $t + \tau$ . One can write

$$\rho(\mathbf{x}_{r,t+\tau}) = \mathcal{P}(\tau)\rho(\mathbf{x}_{r,t}) \quad (40)$$

$$= \int_{\mathbf{x}_{r,t}} \rho(\mathbf{x}_{r,t}) p(\mathbf{x}_{r,t+\tau}|\mathbf{x}_{r,t}) d\mathbf{x}_{r,t} \quad (41)$$

where  $p(\mathbf{x}_{r,t+\tau}|\mathbf{x}_{r,t})$  is the conditional probability density of finding the system at state  $\mathbf{x}_{r,t+\tau}$  given that it was at state  $\mathbf{x}_t$  a number of time steps  $\tau$  before. Note that eq 40 is a purely formal definition as the transition density  $p(\mathbf{x}_{r,t+\tau}|\mathbf{x}_{r,t})$  which usually cannot be explicitly computed. This propagator view is still extremely useful, however, as it is the basis for the development of the kinetic models and algorithms that we will describe here.

One can also define the matrix  $\mathbf{P}(\tau)$  as the discretized version of the propagator in eq 40. In the context of MSMs, for example,  $\mathbf{P}(\tau)$  contains the conditional transition probabilities between disjoint sets, or clusters, of state space. When modeling using the view of the propagator and its discretized part, the unknown quantities relating to the true physical system are approximated by known quantities obtained from the data. Throughout this

section, we will use a “hat” to indicate the estimated or approximate quantity when the distinction is important.

### 6.1. Time-Lagged Independent Component Analysis

Time-lagged independent component analysis (TICA) was originally developed in the field of signal processing.<sup>310</sup> For a time series of  $T$  ordered configurations  $\{\mathbf{x}_t\}$ , we can define the covariance matrix ( $\mathbf{C}_{00}$ ), as well as the time-covariance matrix at “lag time”  $\tau$  ( $\mathbf{C}_{0\tau}$ ) as

$$\mathbf{C}_{00} = \frac{1}{T} \sum_{t=1}^{T-\tau} \mathbf{x}_t \mathbf{x}_t^T \quad (42)$$

$$\mathbf{C}_{0\tau} = \frac{1}{T} \sum_{t=1}^{T-\tau} \mathbf{x}_t \mathbf{x}_{t+\tau}^T \quad (43)$$

where  $\tau$  should be sufficiently small to resolve the dynamics of interest. The generalized eigenvalue problem that characterizes TICA is then given by,

$$\mathbf{C}_{0\tau} \hat{\mathbf{V}} = \mathbf{C}_{00} \hat{\mathbf{V}} \hat{\Lambda} \quad (44)$$

where the eigenvalues  $\hat{\lambda}_\alpha$  are contained in the diagonal of  $\hat{\Lambda}$ . Each eigenvector  $\hat{\mathbf{v}}_\alpha$  is a column of  $\hat{\mathbf{V}}$  and characterizes a latent coordinate with maximal autocorrelation, which is defined as,

$$\begin{aligned} \text{autocorr}(\hat{\mathbf{v}}_\alpha) &= \mathbb{E}[\hat{\mathbf{v}}_\alpha(\mathbf{x}_t) \hat{\mathbf{v}}_\alpha(\mathbf{x}_{t+\tau})], \\ \text{such that } \mathbb{E}[\hat{\mathbf{v}}_\alpha(\mathbf{x}_t) \hat{\mathbf{v}}_\beta(\mathbf{x}_t)] &= \delta_{\alpha\beta} \end{aligned} \quad (45)$$

As will be shown in section 6.2, TICA can be seen as a special case of the variational approach of conformation dynamics for time-reversible dynamics in equilibrium and for linear bases. Under these conditions, the time-covariance matrix  $\mathbf{C}_{0\tau}$  is symmetric (for  $T \rightarrow \infty$ ),<sup>311</sup> and as a consequence, the eigenvalues in eq 44 are real valued.

To guarantee that also at finite time  $T$ ,  $\hat{\lambda}_1 = 1$  and  $\hat{\lambda}_2 < 1$  are achieved, one can use a symmetrized estimator for matrices,  $\mathbf{C}_{00}$  and  $\mathbf{C}_{0\tau}$ , as described in ref.<sup>312</sup>

The latent coordinates  $\{\hat{\mathbf{v}}_\alpha\}$  are the (orthogonal) slow processes within the dynamical system obtained from linear combinations of the degrees of freedom in  $\{\mathbf{x}_t\}$ . An estimate of the relaxation time scales of these processes can be obtained from their corresponding eigenvalues and the lag time according to

$$\hat{\tau}_\alpha = -\frac{\tau}{\ln|\hat{\lambda}_\alpha|} \quad (46)$$

TICA can be used for dimensionality reduction by analyzing the eigenvalue spectrum and truncating the basis  $\{\hat{\mathbf{v}}_i\}$  at an observed spectral gap. Alternatively, one can employ kinetic mapping<sup>131</sup> or commute mapping,<sup>132</sup> which involve weighting the TICA components by their eigenvalues or time scales, respectively. These approaches are similar to the definition of diffusion distance (recall eq 14) in the context of diffusion map (see section 3.2.3), but in this case the low-dimensional manifold is embedded into a space in which geometric distances correspond to times required to transition between pairs of points.<sup>131,132</sup> A reweighting procedure has also been recently developed to remove the bias of TICA estimates that arise from non-equilibrium sampling. While TICA is a linear method, it can be kernelized to accommodate nonlinear coordinates<sup>313,314</sup> (see the treatment of Kernel PCA in section 3.2.2).

TICA was first used for molecular data to identify slow modes in MD simulations of proteins.<sup>315</sup> Shortly after, it was employed

as a preprocessing step in the construction of MSMs (see section 6.3).<sup>251,316</sup> TICA has been leveraged to analyze a variety of biomolecular systems from both simulation and experimental data including the dynamics of protein folding,<sup>252</sup> disordered proteins,<sup>317</sup> protein–peptide, and protein–protein association,<sup>29,255</sup> protein conformational change and ligand binding,<sup>318</sup> binding-induced folding,<sup>256</sup> and kinase functional dynamics.<sup>257</sup> TICA has also been integrated into enhanced sampling algorithms.<sup>319,320</sup>

## 6.2. Variational Approach to Conformational Dynamics

The Variational Approach to Conformation dynamics (VAC)<sup>321</sup> is a principled approach to estimate the eigenvalues and eigenvectors of the dynamical propagator  $\mathcal{P}(\tau)$ .<sup>14</sup> Using VAC theory, it was shown that for a given feature representation  $\mathbf{x}_t = \chi(\mathbf{x}_{t,i})$ , the TICA algorithm produces the variationally optimal approximation to the long-time scale dynamics of  $\mathcal{P}(\tau)$ .<sup>316</sup>

Specifically, the long-time dynamics is governed by the largest eigenvalues  $\lambda_\alpha$  and eigenfunctions  $\psi_\alpha$  of the dynamical propagator, for which corresponding relaxation time scales are given by eq 46.<sup>14,19,21,322</sup> At this point it is convenient to consider not directly the propagator  $\mathcal{P}(\tau)$ , but the so-called transfer operator  $\mathcal{P}_\mu(\tau)$ ,

$$\mathcal{P}_\mu(\tau)\psi_\alpha = \psi_\alpha\lambda_\alpha \quad (47)$$

which propagates probability densities that are normalized by the equilibrium density.<sup>14</sup> This technical point goes beyond the scope of the current review; for now, it is sufficient to say that  $\mathcal{P}_\mu(\tau)$  and  $\mathcal{P}(\tau)$  both encode the system dynamics, and that for time-reversible dynamics we can easily switch between the two operators and their respective eigenfunctions (the interested reader may refer to ref 21 for details).

The aim of the VAC framework is to approximate the leading eigenvalue  $\lambda_\alpha$  and eigenfunction  $\psi_\alpha$  and in this sense it has a similar aim as the variational approach in quantum mechanics.<sup>323</sup> Suppose we want to do that by a linear superposition of feature functions:

$$\hat{\psi}_\alpha = \hat{\mathbf{v}}_\alpha^T \chi(\mathbf{x}_r) \quad (48)$$

$$= \hat{\mathbf{v}}_\alpha^T \mathbf{x} \quad (49)$$

Given the covariance matrices defined in eqs 42 and 43), we define the discrete propagator:

$$\mathbf{P}(\tau) = \mathbf{C}_{00}^{-1} \mathbf{C}_{0\tau}$$

If one chooses indicator functions as features (i.e.,  $\mathbf{x}_i$  is 1 when  $\mathbf{x}_r$  lies in the  $i$ th cluster and 0 otherwise), the matrix  $\mathbf{C}_{00}$  simply contains the counts of the number of data points in each state, the matrix  $\mathbf{C}_{0\tau}$  is a transition count matrix and  $\mathbf{P}(\tau)$  is a transition probability matrix. For other types of features,  $\mathbf{P}(\tau)$  is a so-called Koopman matrix.<sup>312,324</sup> One then performs an eigenvalue decomposition of  $\mathbf{P}(\tau)$ :

$$\mathbf{P}(\tau)\hat{\mathbf{v}}_\alpha = \hat{\mathbf{v}}_\alpha\hat{\lambda}_\alpha \quad (50)$$

According to VAC, the eigenvalues  $\hat{\lambda}_\alpha$  are best approximations to the true eigenvalues  $\lambda_\alpha$  and  $\hat{\psi}_\alpha$  are best approximations to the true eigenfunctions  $\psi_\alpha$  within what the linear superposition of feature functions (eq 48) allows. Note that this statement is true in the absence of statistical errors, namely, in the limit  $T \rightarrow \infty$  in eq 43.

Specifically, this solution maximizes the VAC- $r$  variational score for the first  $d$  eigenvalue-eigenvector pairs, which is bounded from above by the VAC- $r$  score of the highest  $d$  eigenvalues of the operator  $\mathcal{P}(\tau)$ .<sup>325</sup>

$$\text{VAC}_r \equiv \sum_{\alpha=1}^d \hat{\lambda}_\alpha^r \leq \sum_{i=1}^d \lambda_i^r \quad (51)$$

where the equality only holds when the approximation is exact. VAC also implies that the approximated eigenvalues underestimate the true eigenvalues. By virtue of eq 46, this means that, in the limit of infinite data, a data-driven approximation to the true system may predict dynamics that are too fast, but never too slow.<sup>326</sup> However, the approximation can be excellent for practical purposes when a sufficiently long lag time  $\tau$  can be chosen.<sup>327</sup>

Because the VAC defines a scoring function, it enables us to optimize the model's description of the true system's global kinetics by variationally maximizing eq 51. For a given basis set used to represent  $\{\mathbf{x}_t\}$ , the procedure above is equivalent to TICA, and the TICA eigenvalue problem (44) yields the optimal linear approximation to  $\{\psi_\alpha\}$ .<sup>316</sup> However, the existence of a variational score allows us to go beyond linear models and instead parametrize e.g., kernel or deep neural network models by interpreting eq 51 as a loss function.

If, instead, one needs to compare different basis sets for the same  $\tau$  and  $d$ , the variational score (eq 51) can be used to choose the best basis set.<sup>38,40</sup> Since the data available will always be finite, cross-validation should always be used when performing such comparisons,<sup>38,40,328,329</sup> which is possible by exploiting the scalar score in eq 51. The choice of  $\tau$  defines the particular propagator and its matrix approximation. To this point, we have only indicated that  $\tau$  must be sufficiently small to resolve the dynamics of interest. When TICA or the VAC approach are employed in the context of Markovian dynamics (see section 6.3 below),  $\tau$  should also be chosen to be sufficiently large so that the dynamics are indeed Markovian, or “memoryless”. This is discussed further in section 6.3.

The variational approach has been expanded to interpretations involving kinetic variance,<sup>131</sup> commute distances,<sup>132</sup> and diffusion mapping.<sup>142</sup> Basis sets designed to be variationally optimized have also been developed,<sup>330,331</sup> and a deep learning approach to optimize this loss function was reported in ref 332. This method of variationally optimizing a basis is crucial to modern MSM construction, which is discussed in the following section.

## 6.3. Markov State Modeling

In section 6.1, we discussed a strategy for reducing the dimensionality of a molecular trajectory data set  $\{\mathbf{x}_t\}$  to a set of coordinates that optimally describe the relaxation process of the system. In practice, this may involve reducing hundreds or thousands of spatial degrees of freedom to only a few coordinates.

A different and more drastic way of compressing the information on a trajectory is to assign each configuration to a finite number of groups, as seen in section 5). If the groups represent “states” of the system, then this kind of clustering is called a Markov state model (MSM).<sup>22</sup> A state is defined by an indicator basis functions that return 1 if and only if the system is in the corresponding state. With this choice, the covariance matrices  $\mathbf{C}_{00}$  and  $\mathbf{C}_{0\tau}$  become matrices and the propagator  $\mathbf{P}(\tau)$  becomes a transition probability matrix (section 6.2). It was



already understood by Zwanzig that a transition matrix  $\mathbf{P}(\tau)$  between metastable sets serves as an accurate approximation of the kinetics between these sets when the chosen lag time  $\tau$  is longer than the time required to relax within the states.<sup>333</sup>

More recently, it has been proven that in a meaningful MSM  $\mathbf{P}(\tau)$  must approximate the leading eigenvalues and eigenfunctions of  $\mathcal{P}(\tau)$ .<sup>327</sup> This was an important step forward, as it confirmed that MSM state definitions do not need to be metastable: The MSM approximation error of a metastable state decomposition can be reduced by using a finer state discretization whose individual states are not metastable but give rise to a better approximation of the eigenfunction. It is especially important to have a finer discretization in the transition regions between two metastable states, where the slow-process eigenfunctions change a lot. This led to a tendency of constructing MSMs with many states. With the introduction of the VAC and TICA into MSM theory (see section 6.2), a more systematic approach was available to approximate the leading eigenfunctions of  $\mathcal{P}(\tau)$  using less states, and the MSM quality significantly improved.

From a set of state assignments, an MSM is constructed by counting the pairwise transitions at a lag time  $\tau$  and storing those counts in a matrix. Using eq 40, this observed matrix is converted into a transition probability matrix, where each row sums to 1 and represents the discrete probability distribution of transitioning from the state at the row index to any other state including itself. An alternative approach to using a transition probability matrix is estimating a transition rate matrix,<sup>19,334</sup> whose appeal is that it can propagate the modeled dynamics in continuous time instead of discrete time steps  $\tau$ . Note, however, that neither strategy can resolve the fast part of the true dynamics (typically faster than  $\tau$ ), as MSMs and other master equation models coarse-grain the dynamics in space and time.

One typically wants the transition probability matrix to be not only row-stochastic but also to model a system that is ergodic and at thermodynamic equilibrium. To ensure ergodicity, every state must be accessible from every other state given a long enough simulation time. The system is not ergodic if different regions of its representation space are not kinetically connected. In that case, a subset of the system must be used which is locally ergodic. In practice, this is selected by identifying the largest connected subgraph of the graph implied by the transition count matrix  $\mathbf{C}_{0\tau}$ .

Time-reversibility can be imposed on a reversibly connected transition matrix  $\mathbf{P}(\tau)$  via the detailed balance condition,

$$\pi_i P_{ij}(\tau) = \pi_j P_{ji}(\tau) \quad \forall i, j \quad (52)$$

Here,  $\boldsymbol{\pi}$  is the vector of equilibrium probabilities, that is,  $\boldsymbol{\pi}^T \mathbf{P}(\tau) = \boldsymbol{\pi}^T$ . In other words, the equilibrium flux from any state  $i$  to  $j$  must be equal to the equilibrium flux from state  $j$  to  $i$  for all states. This also implies that there are no cycles in the flux. A great deal of research has gone into enforcing this constraint, which is not automatically satisfied when applying direct transition estimation to a finite simulation trajectory. Several studies have derived unbiased estimators for reversible MSMs (i.e., ones that obey detailed balance).<sup>19–21,335,336</sup> An unbiased estimator for a reversible TICA model has also been recently introduced.<sup>312</sup>

The resulting transition matrix is a special case of the  $\mathbf{P}$  matrix discussed in section 6.2, and the VAC can be applied to its eigendecomposition when detailed balance is obeyed. Although the approximated eigenfunctions  $\{\hat{\mathbf{v}}_i\}$  are step functions in

feature space, MSMs can have great expressive power because the feature transformations from  $\mathbf{x}_t$  to  $\mathbf{x}_{t+\tau}$ , where MSM states are defined, can be nonlinear.

As noted above, the lag time  $\tau$  must be sufficiently large to have a good MSM approximation, while being small enough to resolve states of interest—see ref 337 for a mathematical analysis of this trade-off and ref 326 for a qualitative discussion. To check whether the MSM with lag time  $\tau$  is indeed a good approximation of the long-time dynamics, we can assess its adherence to the Chapman–Kolmogorov property<sup>16</sup>

$$\mathbf{P}(n\tau) = [\mathbf{P}(\tau)]^n \quad (53)$$

In practice, to determine a suitable lag time  $\tau$  one often first conducts a so-called “implied time scales” test by observing if the time scales in eq 46 converge to a constant as a function of the lag time.<sup>16</sup> One can easily prove that if the time scales in eq 46 do not depend on  $\tau$ , then the Chapman–Kolmogorov property holds.<sup>16</sup>

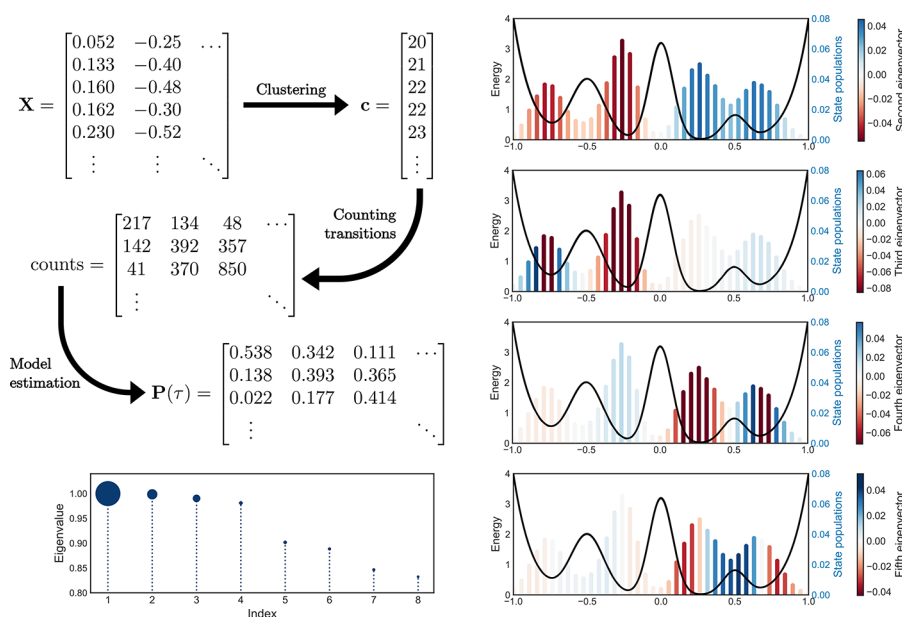
Practically, a straightforward MSM construction protocol proceeds roughly as follows:

1. Transform the spatial coordinates obtained from a simulation data set into a set of features, such as contact distances or dihedral angles (see section 2).  
Ideally, these features capture symmetries in the system (e.g., roto-translational invariance).
2. Optionally perform a basis set transformation of the features, for example as by applying TICA (section 6.1). The use of TICA as a preprocessing step for MSMs facilitates kinetic proximity within the states.<sup>251,316</sup>
3. Perform clustering (see section 5) on the data set to obtain discrete, disjoint states.
4. Construct the observed counts matrix and estimate a transition probability matrix for a chosen lag time, often using detailed balance constraints. The transition probability matrix and the state space decomposition define the MSM.
5. Assess the validity of the Markov assumption at that lag time using implied time scales or the Chapman–Kolmogorov test.

Applying the VAC (section 6.2) to MSM construction entails performing the steps above for the same lag time and assessing its variational score (as defined in eq 51) for the same number of eigenvalues. This must be done under cross-validation, that is, by performing multiple instances of fitting the MSM to a training set and evaluating it on a held out test set.<sup>328</sup> This procedure is discussed in detail in ref 38. For more detailed reviews of the theory underlying MSMs, we refer the reader to refs 21, 22, and 338.

The number of microstates (clusters) in an MSM, following the variational optimization of its parameters is often too large to lead to a readily interpretable model. For MSMs constructed from data sets with millions of points in time, hundreds of microstates are often found to be appropriate.<sup>38,277</sup> The further coarse-graining of MSM state space into so-called “macrostates” has been part of the MSM construction pipeline since the first analyses of small peptide and protein folding systems.<sup>17–19,339,340</sup> Indeed, algorithms to find macrostates have been developed alongside MSM methods since the first formalizations of the latter.<sup>14,341</sup> A summary of these methods and pertinent references are in ref 22.

Given a valid MSM, the eigenvalues and eigenvectors of  $\mathbf{P}(\tau)$  can be analyzed and interpreted. According to the Perron–



**Figure 7.** Schematic illustration describing the MSM construction. First, the data  $\mathbf{X}$  is reduced to a sequence of integer states  $\mathbf{c}$ . Then, transitions among those states after a duration of a lag time  $\tau$  are counted and stored in a count matrix. Next, the MSM itself is estimated from the count matrix to create a transition matrix  $\mathbf{P}(\tau)$ . The eigenvalues of  $\mathbf{P}(\tau)$  correspond to the time scales of the process according to  $t_\alpha \equiv -\tau/\log|\lambda_\alpha|$ . The sizes of circles in the bottom left plot correspond to time scale magnitudes; it can be observed that small changes in eigenvalues result in large changes in time scales due to the logarithmic transformation. On the right, the state populations are shown for the first four dynamical eigenvectors (corresponding to eigenvalue indexes 2–5), along with the underlying potential. The heights of the bars indicate state populations, and the colors indicate flux into (blue) and out of (red) various states.

Frobenius theorem, the dominant eigenvalue is 1 and its corresponding eigenvector contains the equilibrium populations of the states. Every subsequent eigenvalue is smaller than 1 and its corresponding eigenvector represents a dynamical process characterizing flux among the MSM states. The time scales of these processes can be determined from their eigenvalues according to eq 46. A schematic illustrating of the MSM construction for a one-dimensional potential<sup>21</sup> is provided in Figure 7. As MSMs reveal the long-time relaxations, they are naturally well-suited to compute observables of kinetic experiments, such as time-correlation spectroscopy.<sup>19,322,342</sup>

MSMs have proven to be of great use to various applications in studies of molecular—particularly biomolecular—kinetics. These includes the analysis of biophysical processes, such as peptide dynamics<sup>17–19,335,343–355</sup> and their oligomerization,<sup>356</sup> protein folding,<sup>276,339,340,357–365</sup> or a simplified model thereof,<sup>366</sup> ligand binding,<sup>367–373</sup> and conformational change.<sup>257,374–376</sup> Dynamics of other processes, such as carbon nanotubes<sup>377</sup> and solid state atomic systems,<sup>378</sup> were also described using MSMs.

Importantly, MSMs can also reveal the mechanisms of transitions between long-lived states. Transition state theory, originally developed in ref 379, was formulated for MSMs and master equation models in refs 254 and 350 and first applied in ref 254 to compute the full ensemble of protein folding paths of a miniprotein. Similar analyses were conducted for ligand binding,<sup>380,381</sup> kinase activation,<sup>382</sup> and protein–protein association.<sup>29</sup>

Whereas the methods described to this point rely on the time-ordering of the data set, some of the density peaks clustering methods described in section 5.2.2 can produce MSMs without requiring time information nor explicit dimensionality reduction or clustering steps.<sup>179,296,304,309</sup> While this approach is powerful for the above reasons, one drawback of the density peaks

approach to MSM construction is that the Markov states are an output of the clustering; thus, one can not attempt to improve them variationally as can be done with the standard protocol described above. Instead, one can only verify a posteriori if the states define an adequate MSM by estimating a transition probability matrix restricted to these states and, subsequently, examining the implied time scales.

#### 6.4. Koopman Models and VAMP

The algorithms described in sections 6.1, 6.2, and 6.3 require the use of reversible data that obeys the detailed balance condition (eq 52). However, this assumption of microscopic reversibility is not appropriate in all cases. For example, some simulations are deliberately performed out of equilibrium, such as studies of membrane channel conductivity in an external electrostatic potential, or when simulating a force-probe experiment.

To deal with both equilibrium and nonequilibrium data, one can use Koopman models<sup>312,324</sup> and the variational approach to Markov processes (VAMP), which is a generalization of the VAC.<sup>329,383</sup> In the nonequilibrium analogue of TICA one estimates the same covariance matrices as in TICA, and additionally

$$\mathbf{C}_{\tau\tau} \equiv \frac{1}{T} \sum_{t=1}^{T-\tau} \mathbf{x}_{t+\tau} \mathbf{x}_t^T \quad (54)$$

which is an instantaneous covariance matrix as  $\mathbf{C}_{00}$  but at the time-point  $t + \tau$ . If dynamics are stationary, that is, the dynamical propagator does not depend on the absolute value of the time, we expect  $\mathbf{C}_{00} = \mathbf{C}_{\tau\tau}$  in the statistical limit, but if the molecular system is driven by an external force, that is, as in force-induced protein unfolding, or if the simulation trajectories are simply too short to have equilibrated, we will have  $\mathbf{C}_{00} \neq \mathbf{C}_{\tau\tau}$  even in the limit of infinitely many simulation trajectories. Then, the

variationally optimal latent coordinates are obtained by a singular value decomposition of a modified propagation matrix

$$\bar{\mathbf{P}}(\tau) \equiv \mathbf{C}_{00}^{-1/2} \mathbf{C}_{0\tau} \mathbf{C}_{\tau\tau}^{-1/2} = \bar{\mathbf{U}} \hat{\Sigma} \bar{\mathbf{V}}^T \quad (55)$$

where the latent coordinates  $\{\mathbf{u}_\alpha\}$  for the data are stored in the columns of  $\bar{\mathbf{U}} \equiv \mathbf{C}_{00}^{-1/2} \bar{\mathbf{U}}$ , the latent coordinates  $\{\mathbf{v}_\alpha\}$  for the time-lagged data are the columns of  $\bar{\mathbf{V}} \equiv \mathbf{C}_{\tau\tau}^{-1/2} \bar{\mathbf{V}}$ , and the singular values  $\hat{\sigma}_\alpha$  comprise the diagonal of  $\hat{\Sigma}$ .<sup>258,329</sup> Note that the singular value decomposition in eq 55 reduces to a standard eigendecomposition when  $\mathbf{C}_{0\tau}$  is symmetric.

The linear VAMP described above—that is, the nonreversible analogue of TICA—is also called time-lagged canonical correlation analysis, or TCCA.<sup>4,329</sup> For reversible data, eq 55 is equivalent to eq 44.<sup>258</sup> Furthermore, MSMs are a special case of Koopman models when the data is both reversible and represented by a basis of indicator functions.<sup>329</sup> The interested reader is further referred to ref 258, which demonstrates the equivalence of TICA and TCCA under reversible conditions, as well as the relationship of both algorithms to CCA in general.

Importantly, like the VAC, VAMP defines a variational score that can be used to optimize arbitrary shallow or deep machine learning architectures to obtain MSMs or Koopman models for equilibrium or nonequilibrium data:<sup>329</sup>

$$\text{VAMP}_r \equiv \sum_{\alpha=1}^d \hat{\sigma}_\alpha^r \leq \sum_{\alpha=1}^d \sigma_\alpha^r \quad (56)$$

where  $\hat{\sigma}_\alpha$  are the data-driven estimates of the singular values  $\sigma_\alpha$  of the true dynamical operator  $\mathcal{P}(\tau)$ .<sup>329</sup>

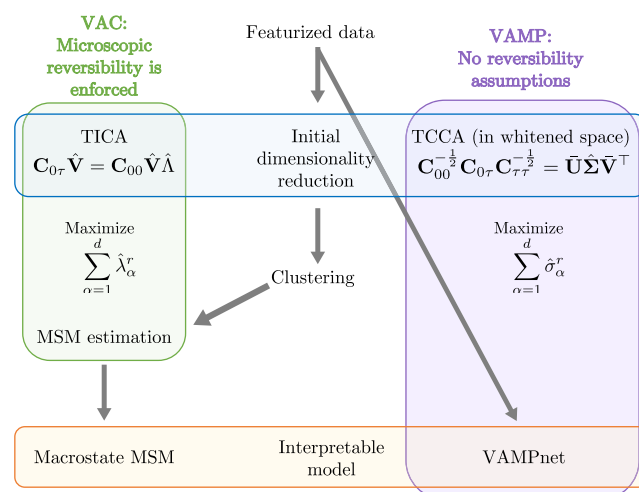
Such a procedure is described in ref 40 and is used to perform feature selection in protein folding. The application of Koopman models to the dynamics of an ion channel in the presence of an electrical gradient is demonstrated in ref 35. The optimization of the VAMP score (eq 56) with neural networks has been presented in ref 157 and is briefly discussed in the following section. Figure 8 depicts the relationships among the variational algorithms discussed in Sections 6.1–6.4.

## 6.5. VAMPnets

In section 6.1, we began with TICA, a dimensionality reduction method that incorporates the temporal ordering of a data set to obtain a low dimensional embedding that best preserves the slowest dynamical processes in the data. Then, in section 6.2, we described the VAC, a framework for variationally approximating the slow dynamical modes of the system. The VAC framework has two key implications. First, it allows us to interpret the TICA embedding as the best variational approximation of the system's slow mode for a linear basis. Second, it allows us to compare different bases, as the VAC score can be used to assess the quality of the resulting model.

In section 6.3, we discussed Markov state modeling, a VAC-compatible method that employs an indicator function basis to partition a data set into discrete, disjoint states. The VAC enables various parameters involved in Markov state modeling (e.g., feature choices, clustering method, and number of clusters) to be compared to determine the parameter set that best approximates the dynamics in the true data.

Sections 6.1–6.3 are designed for reversible data; that is, assumed to be sampled from an equilibrium ensemble. Section 6.4, then, demonstrates that TICA and the VAC can both be generalized to accommodate nonreversible dynamics through TCCA and VAMP, respectively (note that TCCA is equivalent to the linear VAMP). Together, these advances led to the



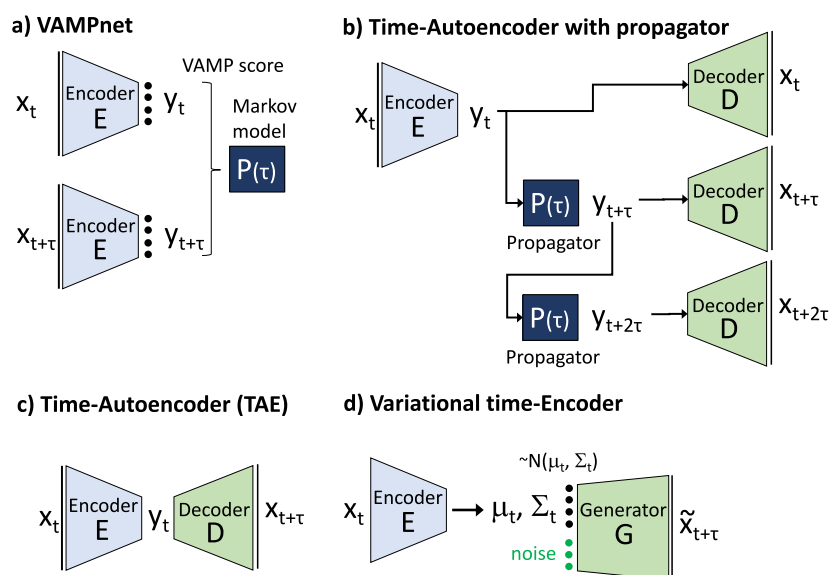
**Figure 8.** From featurized data, an analysis of molecular kinetics can proceed under the assumption of microscopic reversibility (i.e., equilibrium) or not. In the former case, TICA can be applied to reduce dimensionality and may serve as a final or intermediate model. When TICA serves as a step in the MSM construction pipeline, clustering is performed in TICA space and the MSM is estimated from the cluster assignments. Both TICA and MSMs adhere to the VAC, which acts on the eigenvalues of the propagator approximation. A macrostate MSM can be created if further interpretability is desired. In the latter case (no reversibility assumption), TCCA is used instead of TICA to reduce the dimensionality of the data set. From there, clustering can be performed and a reversible MSM can be constructed, or the TCCA results can be regarded as the final model. The relevant variational principle is VAMP, which acts on the singular values of the propagator approximation. A VAMPnet bypasses the majority of the construction pipeline by creating an interpretable model directly from featurized data. VAMPnets employ the VAMP criterion as a loss function in a neural network scheme.

development of VAMPnets:<sup>157</sup> a machine learning framework that leverages the variational approach to learn a kinetic model of the data by optimizing a loss function.

It was first proposed by McGibbon and Pande<sup>328</sup> that the variational score introduced in eq 51 can be interpreted as a loss function and, therefore, can be used to optimize, for example, the parameters of a MSM. Particularly, McGibbon and Pande advocated for the use of cross-validation when determining hyperparameters to accommodate for the necessarily finite sample size as the variational principle is only exact for infinite data.<sup>328</sup> In the generalization of the reversible VAC (eq 51) to the nonreversible VAMP (eq 56), Wu and Noé also focused on the need for cross-validation under variational optimization of hyperparameters.<sup>329</sup>

With VAMPnets,<sup>157</sup> this insight enables the replacement of the Koopman model construction pipeline with a neural network architecture. It has been shown that constructing a reasonable MSM, following the pipeline described in section 6.3 requires a great deal of trial-and-error.<sup>38–40,277,384</sup> To avoid extensive hyperparameter searching in such a sequence of steps, VAMPnets purport to use a self-supervised neural network architecture to collapse the construction procedure (steps 1–4 for MSMs as enumerated in section 6.3) to a single step, in which the VAMP score (eq 56) is used to optimize the featurization network and Markov or Koopman model in an end-to-end fashion (Figure 9a). With VAMPnets, the user only needs to provide the initial features and the neural network learns a deep feature representation of a low-dimensional latent space that





**Figure 9.** Overview of network structures for learning Markovian dynamical models: (a) VAMPnets,<sup>157</sup> (b) time-autoencoder with propagator,<sup>386,387</sup> (c) time-autoencoder,<sup>159</sup> and (d) variational time-encoder.<sup>158</sup>

approximates the eigenfunctions (or singular functions) corresponding to the slow dynamical processes. In this space, a Markov state model or Master equation model of the molecular kinetics is readily obtained. The pipeline is fully automated when the input features are the “raw” simulation data (i.e., Cartesian coordinates); alternatively, features can be determined in a preprocessing step. Several related architectures have been developed which are schematically presented in Figure 9.

VAMPnets have proven useful on protein folding data sets.<sup>157</sup> Similar work on protein folding has followed that employs the VAMP basis function in an autoencoder framework<sup>158,159,161</sup> (Figure 9c and d).

The VAMPnet architecture has also been recently used to analyze functional materials.<sup>385</sup> Similar architectures have been developed to learn deep dynamics models of fluid mechanical systems in an end-to-end fashion<sup>386,387</sup> (Figure 9b).

## 7. RELEVANT SOFTWARE

### 7.1. Feature Representations

Among many other tools, the *PyEMMA*<sup>384,388</sup> Python library contains the construction of standard geometric and knowledge-based features for biomolecular systems, particularly protein systems. The *MDTraj* Python library<sup>389</sup> (which is internally used by *PyEMMA*) is also a resource for creating feature representations from MD simulation data. *PLUMED*<sup>25</sup> allows computing a very large number of collective variables, and even defining ad hoc functions. The three packages *DScribe*,<sup>63</sup> *PANNA*,<sup>390</sup> and *Librascal*<sup>391</sup> can be used to compute all the most widely used numerical features for condensed matter systems.

### 7.2. Dimensionality Reduction

*Scikit-learn*,<sup>392</sup> which itself sources from *NumPy*,<sup>393</sup> *SciPy*,<sup>394</sup> and *Matplotlib*,<sup>395</sup> is a widely used statistical learning package that encompasses a large number of unsupervised learning methods and tools for supervised learning. Linear dimensionality reduction methods like PCA and MDS are included in the package, along with more complex and nonlinear methods like Isomap, Kernel PCA, and t-SNE. The Diffusion Maps method

can be found in the *pyDiffMap* package, available at ref 396. An implementation of the original Sketch-map algorithm<sup>143</sup> can be downloaded from ref 397. Algorithms of deep manifold learning (such as deep autoencoders) are not implemented in any specific standard package, since there is no standard way of constructing such architectures. Libraries, such as *Pytorch*,<sup>398</sup> *TensorFlow*,<sup>399</sup> and *Keras*,<sup>400</sup> can be however used to quickly implement such deep learning architectures. The models are typically available as Supporting Information in the relevant references. For example, the time lagged autoencoder from ref 159 is available at ref 401.

### 7.3. Density Estimation

The *Scikit-learn* library also contains a variety of algorithms for density estimation. Parametric density estimators such as the Gaussian mixture model can be found there, along with nonparametric estimators, such as the histogram estimator and the kernel density estimator. The Point Adaptive *k*-NN estimator can be found at ref 402 or at ref 403.

### 7.4. Clustering

*Scikit-learn* also implements many clustering algorithms. Algorithms like *k*-means and its extensions, as well as DBSCAN, spectral clustering, and hierarchical clustering can all be found there. The *Deeptime* python library, available at,<sup>404</sup> includes *k*-means clustering and some of its extensions. In *METAGUI3*,<sup>306</sup> several clustering algorithms like *k*-medoids or density peaks clustering have been adapted to its use in the context of biased simulations. An implementation of the Advanced density peaks clustering can be found at ref 402 or at ref 403.

Various software packages have been developed that facilitate the construction of the kinetic models discussed in section 6. As mentioned above, *PyEMMA* contains tools for the featurization of biomolecules. In fact, *PyEMMA* was designed to provide an all-in-one tool to create a variety of kinetic models, including MSMs, from unprocessed simulation data. A variety of tutorials are available online and are described in detail in ref 388. The more recent *Deeptime* software provides a generalization of many of these methods beyond biomolecular systems, and performance improvements. Both *PyEMMA* and *Deeptime* can be used for full kinetic model creation, but both packages are

written in a modular style that enables the use of a particular algorithm for just one step in kinetic model building. Thus, both packages can be used for dimensionality reduction, clustering, and transition matrix estimation without going through the entire construction pipeline. The *Enspara* library<sup>405</sup> is an alternative library for Markov state modeling that provides specialized data structures to improve scalability to very large data sets.

## 8. CONCLUSION AND DISCUSSION

Following the increase in processing power and data storage capabilities of the last several decades, molecular simulations have now arrived at an unprecedented level of complexity. Millisecond long simulations of millions of particles are now relatively common, and simulations of more than one billion particles can be achieved on specialized computer architecture.<sup>406,407</sup> In this work, we have provided a comprehensive overview of the unsupervised learning techniques that have proven to be the most useful for the analysis of molecular simulation data.

We started our overview in section 2 with a discussion on the various ways in which a trajectory can be encoded into a numerical representation. Here, the goal is to transform the “raw” Cartesian coordinates of the constituent particles into a more compact numerical representation that preserves all the relevant information on the trajectory. This procedure is a necessary prerequisite for performing any analysis on a trajectory data set, and it is of fundamental importance since it will affect any algorithm subsequently deployed on the chosen representation. For instance, an overly restrictive representation might lead to a systematic bias, while a representation that is too redundant might lead to an increase in computational cost.

The automatic choice of an informative yet compact data representation can be considered an open challenge. It is still common practice to exploit expert knowledge of the system in order to omit degrees of freedom that are deemed irrelevant for the scope of the analysis. For instance, many protein folding studies ignore solvent degrees of freedom since they are considered unimportant for defining the protein states. However, it is well-known that solvent does play a role,<sup>408,409</sup> in simple chemical reactions as well as in complex biomolecular transitions, and researchers have developed dedicated collective variables capable of capturing these phenomena.<sup>410–412</sup> In the spirit of unsupervised learning, ideally one would like to treat the “solvent” in an agnostic manner, at the same level of the “solute”, and automatically learn the most relevant features from data. We believe that substantial improvements are still possible in this direction.

A similar challenge can be identified in the fields of material science and solid state physics. Here, the guiding principle for choosing a data representation is to exploit the knowledge of the symmetries of the system, such as the invariance with respect to exchanges of identical atoms or to rigid rotations. The SOAP and ACSF features are built to automatically satisfy these properties, and represent prominent examples of data representation. However, this manner of representing the configurations becomes memory-intensive when the number of atomic species is more than 3 or 4. Moreover, the definition of these features involves the choice of some important hyperparameters, such as the size of the neighborhood used to compute the features: if this size is too small one risks to overlook important details in the medium-range organization of the system, while if the size is too large the number of features

increases rapidly. Making SOAP and ACSF features definition more general and robust can hence be considered another important open challenge.

As a general guiding principle for choosing a feature space, unless very specific system knowledge is available, it is not advisable to select by hand a very small set of coordinates to describe the system, since any data representation can be made more compact using specifically designed methods of *dimensionality reduction*.

In section 3, we reviewed the algorithms which can be exploited to capture the structure of the data manifold as it appears in the original space using a lower-dimensional representation. The most important and well-known approach to perform this task is PCA, which has been extensively used to analyze molecular simulations for many decades, well before any other unsupervised learning method. This approach is computationally efficient, it is grounded on a robust and simple theory, and it is exact if the data manifold coincides with or is contained in a hyperplane. Several alternative methods have been developed which generalize PCA to the case in which the data manifold is curved and twisted. The most prominent examples are Isomap and Kernel-PCA, which allow “ironing” a curved manifold.

The topology of the embedding manifold poses a hard constraint upon the maximum level of dimensionality reduction which can be achieved with these methods. Consider, for example, a case in which the data points lie on the surface of a three-dimensional sphere. Even if the manifold is two-dimensional, it is impossible—in theory and in practice—to find a two-dimensional representation of the data that preserves the neighborhood relationship of all the data points. In other words, if the embedding manifold is not topologically equivalent to an hyperplane, it is impossible to reduce the dimension of the representation up to the “natural” threshold, which would be the intrinsic dimension of the data.

The front end of research in this field is the development of approaches capable of performing a dimensionality reduction in highly nonlinear and topologically complex manifolds. A remarkable attempt in this direction is the Sketch-map approach, which was developed specifically for visualizing and analyzing molecular simulations. This approach has in principle the potential to go well beyond PCA and Kernel-PCA, but at the price of abandoning the simplicity of linear algebra. Indeed, in the Sketch-map approach the low-dimensional representation is found by optimizing a highly nonlinear functional, which moreover depends on several hyperparameters whose values are system-dependent. Other recent approaches to perform nonlinear dimensionality reduction in molecular simulations are based on neural networks. As discussed in section 3.2.6, autoencoders are a natural choice to address this task. We believe that these approaches can still be significantly improved, for example by porting the impressive know-how that has been developed in image recognition and language processing to the molecular world. For instance, state-of-the-art networks for image recognition are typically very deep and strongly overparameterized, and it is now well understood that this choice helps developing robust and transferable models. This change of paradigm has been so far only partially digested by the atomistic simulation community, where it is still common to exploit architectures with very few hidden layers, and with relatively few parameters.

Another key tool for recapitulating the results of a molecular simulation is estimating the probability density or, equivalently,

the free energy (section 4). Density estimation is a topic that has received much attention in data science.<sup>201</sup> In molecular simulations, choosing the best density estimation method is highly nontrivial, especially if one aims to estimate the probability density as a simultaneous function of many coordinates. Moreover, the probability density in a system characterized by the presence of metastable states varies by several orders of magnitude *by definition*. If information on a plausible form of the density distribution is available, a viable option is estimating the free energy by a parametric method (e.g., mixture models). An alternative strategy, which we described in detail in this review, is estimating the probability density directly on the embedding manifold; that is, without performing any explicit dimensionality reduction beyond the choice of the features describing the system.<sup>13</sup> An open challenge in the field is the development of a free energy estimator, which can be used in high-dimensional feature spaces and, at the same time, can provide an explicit and differentiable function of the coordinates. An important first step forward in this direction has been made, once again using neural networks,<sup>413</sup> where the NN is optimized to return the value of the free energy and its gradient as a function of many collective variables.

In section 5, we review the clustering algorithms that are most commonly used for analyzing molecular simulations. Clustering can be seen as an unsupervised dimensional reduction technique, in which a multidimensional data landscape is mapped to a finite set of states. The use of clustering in the analysis of molecular simulations has been ubiquitous since the seminal work of Brünger et al.,<sup>414</sup> but as highlighted in this Review, the interpretation of a set of clusters will be radically different for different modeling approaches. In *k*-means and related methods, the clusters correspond to a *partition* of the data landscape, which in some approaches approximates a Voronoi tessellation. This partition can be made as fine-grained as desired according to a fixed parameter, which, in the simplest case, is simply the number of clusters. In more advanced approaches, such as hierarchical methods or spectral clustering, the optimal clustering model can be inferred directly from the data, for example, by analyzing the structure of a tree in dendrogram-based approaches. Partitioning schemes are essential for inferring a dynamic model from a simulation since, as discussed in section 6, they provide an appropriate basis for estimating the transition probabilities. In density-based clustering the clusters have a one-to-one correspondence with probability maxima (or, equivalently, to free energy minima).

In section 6, we present an overview of algorithms for dimensionality reduction that explicitly exploit the time-ordering of a molecular dynamics trajectory. The key idea at the basis of many approaches is that the eigenvectors of the matrix approximation of the dynamical propagator describe the slow kinetic modes of a system, which are also assumed to be the most relevant. A straightforward, linear way to estimate these slow modes is through TICA, which has become a cornerstone of many kinetic analyses. In 2013, it was shown that TICA provides the *optimal* linear approximations to the leading eigenfunctions of the true dynamical propagator. This is a special case of the VAC, a variational framework that defines a scalar score for assessing the fidelity of the dynamical modes of a system.

The VAC is a powerful approach that enables the systematic optimization of a kinetic model. One kinetic modeling paradigm that greatly benefits from the development of the VAC is that of

Markov state modeling (see section 6.3). The standard protocol for constructing a MSM leverages the featurization protocols described in section 2, the dimensionality reduction techniques discussed in section 3 and the clustering methods summarized in section 5. Alternatively, these steps can be bypassed by performing density-based clustering, which in principle directly provides the Markov states (see section 4), but without the benefits of the VAC. Although the construction of a MSM requires many hyperparameters related to the choice of featurization, dimensionality reduction, and clustering steps, the VAC enables these hyperparameters to be optimized in an objective fashion.

TICA, the VAC, and MSMs are designed for the dynamics of systems at thermodynamic equilibrium; that is, those that adhere to microscopic reversibility. Recently these approaches were extended to their nonequilibrium analogues—TCCA, VAMP, and Koopman models, respectively (see section 6.4). This suite of methods yield analyses that are less physically interpretable than their reversible counterparts, but can accommodate a greater range of chemical and biophysical systems. An important advance in this class of algorithms is their recent combination with the modern deep learning approaches briefly outlined in section 3.2.6. The key idea is that the VAC or VAMP score is interpreted as a loss function, and a deep neural network incorporating feature representation, dimensionality reduction and clustering can be optimized end-to-end using backpropagation. The combination of state-of-the-art kinetic modeling tools with deep learning is an exciting area for future methods development, and holds the promise of selecting model hyperparameters in a rigorous, automatic, and possibly transferable way.

Altogether, we have given an overview of unsupervised learning methods for data representation, dimensionality reduction, clustering and kinetic modeling in molecular simulation. We have primarily focused on the discussion of classical, so-called “shallow” machine learning methods in which the relevant statistics of the data are often collected in vectors or matrices and then a linear algebra method is solved (e.g., PCA, MSMs, diffusion map) or a simple algorithm is iterated (e.g., *k*-means). The result of the calculation represents the dimensionality reduction, clustering, or kinetic model and often allows us to conduct further analysis in a relatively straightforward way, such as trying to understand which molecular features are most relevant in a learned low-dimensional representation (e.g., PCA, TICA), or compute a variety of kinetic properties (e.g., MSMs). These shallow machine learning methods are robust, efficient, easy to implement and have withstood the test of time.

On the other hand, an overwhelming amount of recent research focuses on deep learning methods, triggered by the renaissance of these methods since the AlexNet paper on image recognition.<sup>415</sup> Deep learning methods are notable in that they can exploit (and require) large amounts of data, and they can learn highly nonlinear transformations and hidden complex patterns that a human designer might not be able to come up with. Their downside is that they are very expensive to train, have a high memory consumption to store their many parameters and their predictions may be unstable and susceptible to noise unless care is taken to prevent that.

Whereas in traditional machine learning applications, such as image processing or game-playing, deep learning methods clearly define the state of the art, this is not always clear in scientific tasks such as the analysis of molecular simulations. As deep learning methods become more and more established in all



areas of science, more emphasis should be placed on their efficiency and reproducibility rather than simply the application of a deep learning idea to a new problem setting. It is our opinion that shallow and deep learning methods both have a role to play in this consideration.

## AUTHOR INFORMATION

### Corresponding Author

**Alessandro Laio** – *International School for Advanced Studies (SISSA), 34014 Trieste, Italy; International Centre for Theoretical Physics (ICTP), Condensed Matter and Statistical Physics Section, 34100 Trieste, Italy; [orcid.org/0000-0001-9164-7907](https://orcid.org/0000-0001-9164-7907); Email: [laio@sissa.it](mailto:laio@sissa.it)*

### Authors

**Aldo Glielmo** – *International School for Advanced Studies (SISSA), 34014 Trieste, Italy; [orcid.org/0000-0002-4737-2878](https://orcid.org/0000-0002-4737-2878)*

**Brooke E. Husic** – *Freie Universität Berlin, Department of Mathematics and Computer Science, 14195 Berlin, Germany; [orcid.org/0000-0002-8020-3750](https://orcid.org/0000-0002-8020-3750)*

**Alex Rodriguez** – *International Centre for Theoretical Physics (ICTP), Condensed Matter and Statistical Physics Section, 34100 Trieste, Italy*

**Cecilia Clementi** – *Freie Universität Berlin, Department for Physics, 14195 Berlin, Germany; Rice University Houston, Department of Chemistry, Houston, Texas 77005, United States; [orcid.org/0000-0001-9221-2358](https://orcid.org/0000-0001-9221-2358)*

**Frank Noé** – *Freie Universität Berlin, Department of Mathematics and Computer Science and Department for Physics, 14195 Berlin, Germany; Rice University Houston, Department of Chemistry, Houston, Texas 77005, United States*

Complete contact information is available at:  
<https://pubs.acs.org/10.1021/acs.chemrev.0c01195>

### Author Contributions

<sup>†</sup>A.G., B.E.H., and A.R. contributed equally.

### Notes

The authors declare no competing financial interest.

### Biographies

Aldo Glielmo is a postdoctoral researcher at the International School for Advanced Studies (SISSA). He obtained his Ph.D. from King's College London, where he worked on the development of machine learning models for interatomic potentials and for many-body wave functions. During his Ph.D., he also worked at the International Centre for Theoretical Physics (ICTP) and at the Alan Turing Institute. His research interests include the developments of computational methods to characterize complex data manifolds, as well as the development of novel data driven models to tackle problems in computational physics. In 2019, Aldo was awarded the IoP thesis prize in computational physics and the "Italy made me" research prize.

Brooke Husic is a Lewis Sigler Scholar and postdoctoral researcher at the Lewis-Sigler Institute for Integrative Genomics at Princeton University with joint affiliations at the Princeton Center for Theoretical Science and the Princeton Center for the Physics of Biological Function. She previously held a postdoctoral research appointment in the Department of Mathematics and Computer Science at the Free University Berlin, earned her Ph.D. in Chemistry at Stanford University, and completed her M.Phil. in Chemistry at the University of Cambridge

through the Gates Cambridge Scholarship program. Her research focuses on the modeling of molecules and their dynamics using statistics and machine learning with applications to protein folding, molecular property prediction, fluid dynamics, and nanomaterials.

Alex Rodriguez obtained his Ph.D. in physical chemistry at the Universitat de Barcelona. He worked as research assistant at the Universitat Politècnica de Catalunya, in the fields of protein folding and drug design. Then, he moved for a postdoctoral position at SISSA, where he started working in the development of unsupervised learning methods. Nowadays, he holds a "Ludwig Boltzmann" Senior Postdoctoral Fellowship at the International Center of Theoretical Physics, where he works in the development and application of Machine Learning methods for the analysis of condensed matter and statistical physics problems.

Cecilia Clementi is Einstein Professor of Physics at Freie Universität (FU) Berlin, Germany. She joined the faculty of FU in June 2020 after 19 years as a Professor of Chemistry at Rice University in Houston, Texas. Cecilia obtained her Ph.D. in Physics at SISSA and was a postdoctoral fellow at the University of California, San Diego, where she was part of the La Jolla Interfaces in Science program. Her research focuses on the development and application of methods for the modeling of complex biophysical processes, by means of molecular dynamics, statistical mechanics, coarse-grained models, experimental data, and machine learning. Cecilia's research has been recognized by a National Science Foundation CAREER Award (2004), and the Robert A. Welch Foundation Norman Hackerman Award in Chemical Research (2009). Since 2016, she is also a co-Director of the National Science Foundation Molecular Sciences Software Institute.

Frank Noé is Full Professor for Artificial Intelligence in the sciences at departments of Mathematics and Physics at FU Berlin. He also holds an Adjunct Professorship for Chemistry at Rice University, Houston, TX. Frank obtained a Ph.D. in Computer Science and Biophysics from University of Heidelberg and started a research group at FU Berlin in 2007. He received an ERC starting grant in 2012 and ERC advanced grant in 2017. In 2019, Frank received the early career award for Theoretical Chemistry of the American Chemical Society (Physical Chemistry division), and he is an ISI highly cited researcher since 2019. Frank's research interest include the development of modern Machine Learning (ML) methods, efficient algorithms and scientific software for fundamental problems in the molecular sciences and biosciences. His main focus is on ML-driven solutions of the sampling problem in statistical mechanics, learning coarse-grained molecular dynamics and variational methods for quantum chemistry.

Alessandro Laio is Full Professor in Statistical and Biological Physics at SISSA, Trieste. He is also Scientific Consultant of the International Center for Theoretical Physics, Trieste. He obtained a Ph.D. in solid-state physics at SISSA and was a postdoctoral fellow at ETH, Switzerland, where he worked first on a QM/MM approach to molecular biochemistry and then on the development of techniques for computer simulations of rare events. Between 2016 and 2019 he has been the president of CECAM, Lausanne. His current research interests are the development of machine learning techniques for analyzing complex data landscapes, and the development of algorithms for enhanced sampling in molecular simulations.

## ACKNOWLEDGMENTS

A.G. and A.L. acknowledge support from the European Union's Horizon 2020 research and innovation program (Grant No. 824143, MaX 'Materials design at the eXascale' Centre of Excellence). B.E.H. and F.N. acknowledge funding from the European Commission (No. ERC CoG 772230 "ScaleCell")

and MATH+, the Berlin Mathematics Center (No. EF1-2). F.N. acknowledges the BMBF (Berlin Institute for the Foundations of Learning and Data – BIFOLD). C.C. acknowledges support by the National Science Foundation (CHE-1738990, CHE-1900374, and PHY-1427654), the Welch Foundation (C-1570), the Deutsche Forschungsgemeinschaft (SFB/TRR 186/A12 and SFB 1078/C7), and the Einstein Foundation Berlin.

## REFERENCES

- (1) Bishop, C. M. *Pattern Recognition and Machine Learning*; Springer, 2006.
- (2) Sidky, H.; Chen, W.; Ferguson, A. L. Machine learning for collective variable discovery and enhanced sampling in biomolecular simulation. *Mol. Phys.* **2020**, *118*, No. e1737742.
- (3) Rohrdanz, M. A.; Zheng, W.; Clementi, C. Discovering mountain passes via torchlight: Methods for the definition of reaction coordinates and pathways in complex macromolecular reactions. *Annu. Rev. Phys. Chem.* **2013**, *64*, 295–316.
- (4) Noé, F.; Tkatchenko, A.; Müller, K.-R.; Clementi, C. Machine learning for molecular simulation. *Annu. Rev. Phys. Chem.* **2020**, *71*, 361–390.
- (5) Abrams, C.; Bussi, G. Enhanced sampling in molecular dynamics using metadynamics, replica-exchange, and temperature-acceleration. *Entropy* **2014**, *16*, 163–199.
- (6) Ferguson, A. L. Machine learning and data science in soft materials engineering. *J. Phys.: Condens. Matter* **2018**, *30*, 043002.
- (7) Butler, K. T.; Davies, D. W.; Cartwright, H.; Isayev, O.; Walsh, A. Machine learning for molecular and materials science. *Nature* **2018**, *559*, 547–555.
- (8) Jackson, N. E.; Webb, M. A.; de Pablo, J. J. Recent advances in machine learning towards multiscale soft materials design. *Curr. Opin. Chem. Eng.* **2019**, *23*, 106–114.
- (9) Häse, F.; Roch, L. M.; Friederich, P.; Aspuru-Guzik, A. Designing and understanding light-harvesting devices with machine learning. *Nat. Commun.* **2020**, *11*, 4587.
- (10) Metropolis, N.; Rosenbluth, A. W.; Rosenbluth, M. N.; Teller, A. H.; Teller, E. Equation of state calculations by fast computing machines. *J. Chem. Phys.* **1953**, *21*, 1087–1092.
- (11) Braun, E.; Gilmer, J.; Mayes, H. B.; Mobley, D. L.; Monroe, J. I.; Prasad, S.; Zuckerman, D. M. Best practices for foundations in molecular simulations [Article v1. 0]. *Live CoMS* **2019**, 1.
- (12) Piana, S.; Laio, A. Advillin folding takes place on a hypersurface of small dimensionality. *Phys. Rev. Lett.* **2008**, *101*, 208101.
- (13) Rodriguez, A.; d'Errico, M.; Facco, E.; Laio, A. Computing the free energy without collective variables. *J. Chem. Theory Comput.* **2018**, *14*, 1206–1215.
- (14) Schütte, C.; Fischer, A.; Huisinga, W.; Deuffhard, P. A direct approach to conformational dynamics based on hybrid Monte Carlo. *J. Comput. Phys.* **1999**, *151*, 146–168.
- (15) Singhal, N.; Snow, C. D.; Pande, V. S. Using path sampling to build better Markovian state models: predicting the folding rate and mechanism of a tryptophan zipper beta hairpin. *J. Chem. Phys.* **2004**, *121*, 415–425.
- (16) Swope, W. C.; Pitera, J. W.; Suits, F. Describing protein folding kinetics by molecular dynamics simulations. 1. Theory. *J. Phys. Chem. B* **2004**, *108*, 6571–6581.
- (17) Chodera, J. D.; Singhal, N.; Pande, V. S.; Dill, K. A.; Swope, W. C. Automatic discovery of metastable states for the construction of Markov models of macromolecular conformational dynamics. *J. Chem. Phys.* **2007**, *126*, 155101.
- (18) Noé, F.; Horenko, I.; Schütte, C.; Smith, J. C. Hierarchical analysis of conformational dynamics in biomolecules: Transition networks of metastable states. *J. Chem. Phys.* **2007**, *126*, 155102.
- (19) Buchete, N.-V.; Hummer, G. Coarse master equations for peptide folding dynamics. *J. Phys. Chem. B* **2008**, *112*, 6057–6069.
- (20) Bowman, G. R.; Beauchamp, K. A.; Boxer, G.; Pande, V. S. Progress and challenges in the automated construction of Markov state models for full protein systems. *J. Chem. Phys.* **2009**, *131*, 124101.
- (21) Prinz, J.-H.; Wu, H.; Sarich, M.; Keller, B.; Senne, M.; Held, M.; Chodera, J. D.; Schütte, C.; Noé, F. Markov models of molecular kinetics: Generation and validation. *J. Chem. Phys.* **2011**, *134*, 174105.
- (22) Husic, B. E.; Pande, V. S. Markov state models: From an art to a science. *J. Am. Chem. Soc.* **2018**, *140*, 2386–2396.
- (23) Cohen, F. E.; Sternberg, M. J. On the prediction of protein structure: the significance of the root-mean-square deviation. *J. Mol. Biol.* **1980**, *138*, 321–333.
- (24) Gordon, M. S.; Pople, J. A. Approximate self-consistent molecular-orbital theory. VI. INDO calculated equilibrium geometries. *J. Chem. Phys.* **1968**, *49*, 4643–4650.
- (25) Bonomi, M.; Branduardi, D.; Bussi, G.; Camilloni, C.; Provasi, D.; Raiteri, P.; Donadio, D.; Marinelli, F.; Pietrucci, F.; Broglia, R. A.; et al. PLUMED: A portable plugin for free-energy calculations with molecular dynamics. *Comput. Phys. Commun.* **2009**, *180*, 1961–1972.
- (26) Cossio, P.; Laio, A.; Pietrucci, F. Which similarity measure is better for analyzing protein structures in a molecular dynamics trajectory? *Phys. Chem. Chem. Phys.* **2011**, *13*, 10421–10425.
- (27) Ramachandran, G.; Ramakrishnan, C.; Sasisekharan, V. Stereochemistry of polypeptide chain configurations. *J. Mol. Biol.* **1963**, *7*, 95–99.
- (28) Buch, I.; Giorgino, T.; De Fabritiis, G. Complete reconstruction of an enzyme-inhibitor binding process by molecular dynamics simulations. *Proc. Natl. Acad. Sci. U. S. A.* **2011**, *108*, 10184–10189.
- (29) Plattner, N.; Doerr, S.; De Fabritiis, G.; Noé, F. Complete protein-protein association kinetics in atomic detail revealed by molecular dynamics simulations and Markov modelling. *Nat. Chem.* **2017**, *9*, 1005.
- (30) Wayment-Steele, H. K.; Hernández, C. X.; Pande, V. S. Modelling intrinsically disordered protein dynamics as networks of transient secondary structure. *bioRxiv* **2018**, 377564.
- (31) Shrake, A.; Rupley, J. A. Environment and exposure to solvent of protein atoms. Lysozyme and insulin. *J. Mol. Biol.* **1973**, *79*, 351–371.
- (32) McKiernan, K. A.; Husic, B. E.; Pande, V. S. Modeling the mechanism of CLN025 beta-hairpin formation. *J. Chem. Phys.* **2017**, *147*, 104107.
- (33) Harrigan, M. P.; Shukla, D.; Pande, V. S. Conserve water: A method for the analysis of solvent in molecular dynamics. *J. Chem. Theory Comput.* **2015**, *11*, 1094–1101.
- (34) Harrigan, M. P.; McKiernan, K. A.; Shanmugasundaram, V.; Denny, R. A.; Pande, V. S. Markov modeling reveals novel intracellular modulation of the human TREK-2 selectivity filter. *Sci. Rep.* **2017**, *7*, 632.
- (35) Paul, F.; Wu, H.; Vossel, M.; de Groot, B. L.; Noé, F. Identification of kinetic order parameters for non-equilibrium dynamics. *J. Chem. Phys.* **2019**, *150*, 164120.
- (36) Briones, R.; Aponte-Santamaría, C.; de Groot, B. L. Localization and ordering of lipids around aquaporin-0: protein and lipid mobility effects. *Front. Physiol.* **2017**, *8*, 124.
- (37) Corradi, V.; Mendez-Villuendas, E.; Ingólfsson, H. I.; Gu, R.-X.; Siuda, I.; Melo, M. N.; Moussatova, A.; DeGagné, L. J.; Sejdiu, B. I.; Singh, G.; et al. Lipid-protein interactions are unique fingerprints for membrane proteins. *ACS Cent. Sci.* **2018**, *4*, 709–717.
- (38) Husic, B. E.; McGibbon, R. T.; Sultan, M. M.; Pande, V. S. Optimized parameter selection reveals trends in Markov state models for protein folding. *J. Chem. Phys.* **2016**, *145*, 194103.
- (39) Harrigan, M. P.; Sultan, M. M.; Hernández, C. X.; Husic, B. E.; Eastman, P.; Schwantes, C. R.; Beauchamp, K. A.; McGibbon, R. T.; Pande, V. S. MSMBuilder: Statistical models for biomolecular dynamics. *Biophys. J.* **2017**, *112*, 10–15.
- (40) Scherer, M. K.; Husic, B. E.; Hoffmann, M.; Paul, F.; Wu, H.; Noé, F. Variational selection of features for molecular kinetics. *J. Chem. Phys.* **2019**, *150*, 194108.
- (41) Kuhn, H. W. The Hungarian method for the assignment problem. *Nav. Res. Logist. Q.* **1955**, *2*, 83–97.
- (42) Reinhard, F.; Grubmüller, H. Estimation of absolute solvent and solvation shell entropies via permutation reduction. *J. Chem. Phys.* **2007**, *126*, 014102.

- (43) Noé, F.; Olsson, S.; Köhler, J.; Wu, H. Boltzmann generators: Sampling equilibrium states of many-body systems with deep learning. *Science* **2019**, *365*, eaaw1147.
- (44) Cisneros, G. A.; Wikfeldt, K. T.; Ojamäe, L.; Lu, J.; Xu, Y.; Torabifard, H.; Bartók, A. P.; Csányi, G.; Molinero, V.; Paesani, F. Modeling molecular interactions in water: From pairwise to many-body potential energy functions. *Chem. Rev.* **2016**, *116*, 7501–7528.
- (45) Glielmo, A.; Zeni, C.; Fekete, A.; De Vita, A. *Machine Learning Meets Quantum Physics*; Springer International Publishing, 2020; pp 67–98.
- (46) Deringer, V. L.; Csányi, G. Machine learning based interatomic potential for amorphous carbon. *Phys. Rev. B: Condens. Matter Mater. Phys.* **2017**, *95*, 094203.
- (47) Behler, J.; Parrinello, M. Generalized neural-network representation of high-dimensional potential-energy surfaces. *Phys. Rev. Lett.* **2007**, *98*, 146401–4.
- (48) Behler, J. Atom-centered symmetry functions for constructing high-dimensional neural network potentials. *J. Chem. Phys.* **2011**, *134*, 074106–14.
- (49) Artrith, N.; Morawietz, T.; Behler, J. High-dimensional neural-network potentials for multicomponent systems: Applications to zinc oxide. *Phys. Rev. B: Condens. Matter Mater. Phys.* **2011**, *83*, 153101–4.
- (50) Artrith, N.; Hiller, B.; Behler, J. Neural network potentials for metals and oxides - First applications to copper clusters at zinc oxide. *Phys. Status Solidi B* **2013**, *250*, 1191–1203.
- (51) Smith, J. S.; Isayev, O.; Roitberg, A. E. ANI-1: an extensible neural network potential with DFT accuracy at force field computational cost. *Chem. Sci.* **2017**, *8*, 3192–3203.
- (52) Cheng, B.; Mazzola, G.; Pickard, C. J.; Ceriotti, M. Evidence for supercritical behaviour of high-pressure liquid hydrogen. *Nature* **2020**, *585*, 217–220.
- (53) Bartók, A. P.; Payne, M. C.; Kondor, R.; Csányi, G. Gaussian approximation potentials: The accuracy of quantum mechanics, without the electrons. *Phys. Rev. Lett.* **2010**, *104*, 136403–4.
- (54) Bartók, A. P.; Kondor, R.; Csányi, G. On representing chemical environments. *Phys. Rev. B: Condens. Matter Mater. Phys.* **2013**, *87*, 184115–16.
- (55) Glielmo, A.; Zeni, C.; De Vita, A. Efficient nonparametric *n*-body force fields from machine learning. *Phys. Rev. B: Condens. Matter Mater. Phys.* **2018**, *97*, 184037.
- (56) Rupp, M.; Tkatchenko, A.; Müller, K. R.; von Lilienfeld, O. A. Fast and accurate modeling of molecular atomization energies with machine learning. *Phys. Rev. Lett.* **2012**, *108*, 058301–5.
- (57) Hansen, K.; Biegler, F.; Ramakrishnan, R.; Pronobis, W.; von Lilienfeld, O. A.; Müller, K. R.; Tkatchenko, A. Machine learning predictions of molecular properties: Accurate many-body potentials and nonlocality in chemical space. *J. Phys. Chem. Lett.* **2015**, *6*, 2326–2331.
- (58) Faber, F. A.; Hutchison, L.; Huang, B.; Gilmer, J.; Schoenholz, S. S.; Dahl, G. E.; Vinyals, O.; Kearnes, S.; Riley, P. F.; Von Lilienfeld, O. A. Prediction errors of molecular machine learning models lower than hybrid DFT error. *J. Chem. Theory Comput.* **2017**, *13*, S255–S264.
- (59) Huo, H.; Rupp, M. Unified representation for machine learning of molecules and crystals. *arXiv*, 2017, 1704.06439. <https://arxiv.org/abs/1704.06439>.
- (60) Geiger, P.; Dellago, C. Neural networks for local structure detection in polymorphic systems. *J. Chem. Phys.* **2013**, *139*, 164105.
- (61) Pietrucci, F.; Martoňák, R. Systematic comparison of crystalline and amorphous phases: Charting the landscape of water structures and transformations. *J. Chem. Phys.* **2015**, *142*, 104704.
- (62) Zeni, C.; Rossi, K.; Glielmo, A.; Baletto, F. On machine learning force fields for metallic nanoparticles. *Adv. Phys. X* **2019**, *4*, 1654919.
- (63) Himanen, L.; Jäger, M. O. J.; Morooka, E. V.; Federici Canova, F.; Ranawat, Y. S.; Gao, D. Z.; Rinke, P.; Foster, A. S. Dscribe: Library of descriptors for machine learning in materials science. *Comput. Phys. Commun.* **2020**, *247*, 106949.
- (64) Langer, M. F.; Goebmann, A.; Rupp, M. Representations of molecules and materials for interpolation of quantum-mechanical simulations via machine learning. *arXiv*, 2020, 2003.12081. <https://arxiv.org/abs/2003.12081>.
- (65) Battaglia, P. W.; Hamrick, J. B.; Bapst, V.; Sanchez-Gonzalez, A.; Zambaldi, V.; Malinowski, M.; Tacchetti, A.; Raposo, D.; Santoro, A.; Faulkner, R. et al. Relational inductive biases, deep learning, and graph networks. *arXiv*, 2018, 1806.01261. <https://arxiv.org/abs/1806.01261>.
- (66) Schütt, K.; Kindermans, P.-J.; Sauceda Felix, H. E.; Chmiela, S.; Tkatchenko, A.; Müller, K.-R. SchNet: A continuous-filter convolutional neural network for modeling quantum interactions. *Adv. Neural. Inf. Process. Syst.* **2017**, *30*, 991–1001.
- (67) Unke, O. T.; Muwly, M. PhysNet: A neural network for predicting energies, forces, dipole moments, and partial charges. *J. Chem. Theory Comput.* **2019**, *15*, 3678–3693.
- (68) Klicpera, J.; Groß, J.; Günnemann, S. Directional message passing for molecular graphs. *ICLR 2020*, 2020.
- (69) Anderson, B.; Hy, T. S.; Kondor, R. Cormorant: Covariant molecular neural networks. *Adv. Neural. Inf. Process. Syst.* **2019**, *32*, 14537–14546.
- (70) Thomas, N.; Smidt, T.; Kearnes, S.; Yang, L.; Li, L.; Kohlhoff, K.; Riley, P. Tensor field networks: Rotation-and translation-equivariant neural networks for 3D point clouds. *arXiv*, 2018, 1802.08219. <https://arxiv.org/abs/1802.08219>.
- (71) Zhang, L.; Han, J.; Wang, H.; Car, R.; E, W. Deep potential molecular dynamics: A scalable model with the accuracy of quantum mechanics. *Phys. Rev. Lett.* **2018**, *120*, 143001.
- (72) Schütt, K. T.; Arbabzadah, F.; Chmiela, S.; Müller, K. R.; Tkatchenko, A. Quantum-chemical insights from deep tensor neural networks. *Nat. Commun.* **2017**, *8*, 13890.
- (73) Ruza, J.; Wang, W.; Schwalbe-Koda, D.; Axelrod, S.; Harris, W. H.; Gomez-Bombarelli, R. Temperature-transferable coarse-graining of ionic liquids with dual graph convolutional neural networks. *arXiv*, 2020, 2007.14144. <https://arxiv.org/abs/2007.14144>.
- (74) Husic, B. E.; Charron, N. E.; Lemm, D.; Wang, J.; Pérez, A.; Majewski, M.; Krämer, A.; Chen, Y.; Olsson, S.; de Fabritiis, G.; et al. Coarse graining molecular dynamics with graph neural networks. *J. Chem. Phys.* **2020**, *153*, 194101.
- (75) von Lilienfeld, O. A.; Müller, K.-R.; Tkatchenko, A. Exploring chemical compound space with quantum-based machine learning. *Nat. Rev. Chem.* **2020**, *4*, 347–358.
- (76) Van Der Maaten, L.; Postma, E.; Van den Herik, J. Dimensionality reduction: a comparative review. *J. Mach. Learn. Res.* **2009**, *10*, 66–71.
- (77) Wang, J.; Ferguson, A. L. Nonlinear machine learning in simulations of soft and biological materials. *Mol. Simul.* **2018**, *44*, 1090–1107.
- (78) Pearson, K. On lines and planes of closest fit to systems of points in space. *Philos. Mag.* **1901**, *2*, 559–572.
- (79) Jolliffe, I. T. *Principal Component Analysis*, Springer Series in Statistics; Springer Science & Business Media: New York, NY, 2013.
- (80) Jolliffe, I. T.; Cadima, J. Principal component analysis: a review and recent developments. *Philos. Trans. R. Soc., A* **2016**, *374*, 20150202.
- (81) Ichiye, T.; Karplus, M. Collective motions in proteins: a covariance analysis of atomic fluctuations in molecular dynamics and normal mode simulations. *Proteins: Struct., Funct., Genet.* **1991**, *11*, 205–217.
- (82) García, A. E. Large-amplitude nonlinear motions in proteins. *Phys. Rev. Lett.* **1992**, *68*, 2696–2699.
- (83) Amadei, A.; Linssen, A.; Berendsen, H. Essential Dynamics of Proteins. *Proteins: Struct., Funct., Genet.* **1993**, *17*, 412–425.
- (84) Karpen, M. E.; Tobias, D. J.; Brooks, C. L., III Statistical clustering techniques for the analysis of long molecular dynamics trajectories: analysis of 2.2-ns trajectories of YPGDV. *Biochemistry* **1993**, *32*, 412–420.
- (85) van Aalten, D. M.; Conn, D. A.; de Groot, B. L.; Berendsen, H. J.; Findlay, J. B.; Amadei, A. Protein dynamics derived from clusters of crystal structures. *Biophys. J.* **1997**, *73*, 2891–2896.
- (86) de Groot, B. L.; Daura, X.; Mark, A. E.; Grubmüller, H. Essential dynamics of reversible peptide folding: memory-free conformational



dynamics governed by internal hydrogen bonds. *J. Mol. Biol.* **2001**, 309, 299–313.

(87) Daidone, I.; Amadei, A.; Roccatano, D.; Di Nola, A. Molecular dynamics simulation of protein folding by essential dynamics sampling: Folding landscape of horse heart cytochrome c. *Biophys. J.* **2003**, 85, 2865–2871.

(88) Tournier, A. L.; Smith, J. C. Principal components of the protein dynamical transition. *Phys. Rev. Lett.* **2003**, 91, 573–574.

(89) Mu, Y.; Nguyen, P. H.; Stock, G. Energy landscape of a small peptide revealed by dihedral angle principal component analysis. *Proteins: Struct., Funct., Genet.* **2005**, 58, 45–52.

(90) Lange, O. F.; Lakomek, N.-A.; Farès, C.; Schröder, G. F.; Walter, K. F. A.; Becker, S.; Meiler, J.; Grubmüller, H.; Griesinger, C.; de Groot, B. L. Recognition dynamics up to microseconds revealed from an RDC-derived ubiquitin ensemble in solution. *Science* **2008**, 320, 1471–1475.

(91) Spiwok, V.; Lipovová, P.; Králová, B. Metadynamics in essential coordinates: Free energy simulation of conformational changes. *J. Phys. Chem. B* **2007**, 111, 3073–3076.

(92) Anelli, A.; Engel, E. A.; Pickard, C. J.; Ceriotti, M. Generalized convex hull construction for materials discovery. *Phys. Rev. Mater.* **2018**, 2, 103804.

(93) Helfrecht, B. A.; Cersonsky, R. K.; Fraux, G.; Ceriotti, M. Structure–property maps with kernel principal covariates regression. *arXiv*, 2020, 2002.05076. <https://arxiv.org/abs/2002.05076>.

(94) Cheng, B.; Griffiths, R.-R.; Wengert, S.; Kunkel, C.; Stenczel, T.; Zhu, B.; Deringer, V. L.; Bernstein, N.; Margraf, J. T.; Reuter, K.; et al. Mapping materials and molecules. *Acc. Chem. Res.* **2020**, 53, 1981–1991.

(95) Clarage, J. B.; Romo, T.; Andrews, B. K.; Pettitt, B. M.; Phillips, G. N. A sampling problem in molecular dynamics simulations of macromolecules. *Proc. Natl. Acad. Sci. U. S. A.* **1995**, 92, 3288–3292.

(96) Balsera, M. A.; Wriggers, W.; Oono, Y.; Schulten, K. Principal component analysis and long time protein dynamics. *J. Phys. Chem.* **1996**, 100, 2567–2572.

(97) Lange, O. F.; Grubmüller, H. Can principal components yield a dimension reduced description of protein dynamics on long time scales? *J. Phys. Chem. B* **2006**, 110, 22842–22852.

(98) Amadei, A.; Ceruso, M. A.; Di Nola, A. On the convergence of the conformational coordinates basis set obtained by the essential dynamics analysis of proteins' molecular dynamics simulations. *Proteins: Struct., Funct., Genet.* **1999**, 36, 419–424.

(99) de Groot, B. L.; van Aalten, D. M.; Amadei, A.; Berendsen, H. J. The consistency of large concerted motions in proteins in molecular dynamics simulations. *Biophys. J.* **1996**, 71, 1707–1713.

(100) Hess, B. Convergence of sampling in protein simulations. *Phys. Rev. E: Stat. Phys., Plasmas, Fluids, Relat. Interdiscip. Top.* **2002**, 65, 031910.

(101) Das, P.; Moll, M.; Stamati, H.; Kaviraki, L. E.; Clementi, C. Low-dimensional, free-energy landscapes of protein-folding reactions by nonlinear dimensionality reduction. *Proc. Natl. Acad. Sci. U. S. A.* **2006**, 103, 9885–9890.

(102) Young, G.; Householder, A. S. Discussion of a set of points in terms of their mutual distances. *Psychometrika* **1938**, 3, 19–22.

(103) Torgerson, W. S. Multidimensional scaling: I. Theory and method. *Psychometrika* **1952**, 17, 401–419.

(104) France, S. L.; Carroll, J. D. Two-way multidimensional scaling: A review. *IEEE Trans. Syst. Man Cybern. Syst.* **2011**, 41, 644–661.

(105) Shawe-Taylor, J.; Cristianini, N. *Kernel methods for pattern analysis* **2004**, DOI: 10.1017/CBO9780511809682.

(106) Tenenbaum, J. B.; de Silva, V.; Langford, J. C. A global geometric framework for nonlinear dimensionality reduction. *Science* **2000**, 290, 2319–2323.

(107) Balasubramanian, M.; Schwartz, E. L. The Isomap algorithm and topological stability. *Science* **2002**, 295, 7.

(108) Plaku, E.; Stamati, H.; Clementi, C.; Kaviraki, L. E. Fast and reliable analysis of molecular motion using proximity relations and dimensionality reduction. *Proteins: Struct., Funct., Genet.* **2007**, 67, 897–907.

(109) Brown, W. M.; Martin, S.; Pollock, S. N.; Coutsiar, E. A.; Watson, J.-P. Algorithmic dimensionality reduction for molecular structure analysis. *J. Chem. Phys.* **2008**, 129, 064118–14.

(110) Stamati, H.; Clementi, C.; Kaviraki, L. E. Application of nonlinear dimensionality reduction to characterize the conformational landscape of small peptides. *Proteins: Struct., Funct., Genet.* **2010**, 78, 223–235.

(111) Duan, M.; Fan, J.; Li, M.; Han, L.; Huo, S. Evaluation of dimensionality-reduction methods from peptide folding-unfolding simulations. *J. Chem. Theory Comput.* **2013**, 9, 2490–2497.

(112) Spiwok, V.; Králová, B. Metadynamics in the conformational space nonlinearly dimensionally reduced by Isomap. *J. Chem. Phys.* **2011**, 135, 224504–7.

(113) Hashemian, B.; Millán, D.; Arroyo, M. Modeling and enhanced sampling of molecular systems with smooth and nonlinear data-driven collective variables. *J. Chem. Phys.* **2013**, 139, 214101.

(114) Campadelli, P.; Casiraghi, E.; Ceruti, C.; Rozza, A. Intrinsic dimension estimation: Relevant techniques and a benchmark framework. *Math. Probl. Eng.* **2015**, 2015, 759567.

(115) Camastra, F.; Staiano, A. Intrinsic dimension estimation: Advances and open problems. *Inf. Sci.* **2016**, 328, 26–41.

(116) Scholkopf, B.; Smola, A.; Müller, K. R. Nonlinear component analysis as a kernel eigenvalue problem. *Neural Comput.* **1998**, 10, 1299–1319.

(117) Choi, H.; Choi, S. Robust kernel isomap. *Pattern Recognit.* **2007**, 40, 853–862.

(118) Antoniou, D.; Schwartz, S. D. Toward identification of the reaction coordinate directly from the transition state ensemble using the kernel PCA method. *J. Phys. Chem. B* **2011**, 115, 2465–2469.

(119) David, C. C.; Jacobs, D. J. In *Protein Dynamics: Methods and Protocols*; Livesay, D. R., Ed.; Humana Press: Totowa, NJ, 2014; pp 193–226.

(120) Thompson, A. P.; Swiler, L. P.; Trott, C. R.; Foiles, S. M.; Tucker, G. J. Spectral neighbor analysis method for automated generation of quantum-accurate interatomic potentials. *J. Comput. Phys.* **2015**, 285, 316–330.

(121) Bartók, A. P.; Kermode, J.; Bernstein, N.; Csányi, G. Machine learning a general purpose interatomic potential for silicon. *Phys. Rev. X* **2018**, 041048.

(122) Rowe, P.; Deringer, V. L.; Gasparotto, P.; Csányi, G.; Michaelides, A. An accurate and transferable machine learning potential for carbon. *arXiv*, 2020, 2006.13655. <https://arxiv.org/abs/2006.13655>.

(123) Helfrecht, B. A.; Semino, R.; Pireddu, G.; Auerbach, S. M.; Ceriotti, M. A new kind of atlas of zeolite building blocks. *J. Chem. Phys.* **2019**, 151, 154112.

(124) Reinhardt, A.; Pickard, C. J.; Cheng, B. Predicting the phase diagram of titanium dioxide with random search and pattern recognition. *Phys. Chem. Chem. Phys.* **2020**, 22, 12697–12705.

(125) Coifman, R. R.; Lafon, S.; Lee, A. B.; Maggioni, M.; Nadler, B.; Warner, F.; Zucker, S. W. Geometric diffusions as a tool for harmonic analysis and structure definition of data: Diffusion maps. *Proc. Natl. Acad. Sci. U. S. A.* **2005**, 102, 7426–7431.

(126) Coifman, R. R.; Lafon, S. Diffusion maps. *Appl. Comput. Harmon. Anal.* **2006**, 21, 5–30.

(127) Trstanova, Z.; Leimkuhler, B.; Lelièvre, T. Local and global perspectives on diffusion maps in the analysis of molecular systems. *Proc. R. Soc. London, Ser. A* **2020**, 476, 20190036.

(128) Nadler, B.; Lafon, S.; Coifman, R. R.; Kevrekidis, I. G. Diffusion maps, spectral clustering and reaction coordinates of dynamical systems. *Appl. Comput. Harmon. Anal.* **2006**, 21, 113–127.

(129) Coifman, R. R.; Lafon, S.; Lee, A. B.; Maggioni, M.; Nadler, B.; Warner, F.; Zucker, S. W. Geometric diffusions as a tool for harmonic analysis and structure definition of data: Diffusion maps. *Proc. Natl. Acad. Sci. U. S. A.* **2005**, 102, 7426–7431.

(130) Zheng, W.; Vargiu, A. V.; Rohrdanz, M. A.; Carloni, P.; Clementi, C. Molecular recognition of DNA by ligands: Roughness and complexity of the free energy profile. *J. Chem. Phys.* **2013**, 139, 145102.

- (131) Noé, F.; Clementi, C. Kinetic distance and kinetic maps from molecular dynamics simulation. *J. Chem. Theory Comput.* **2015**, *11*, 5002–5011.
- (132) Noé, F.; Banisch, R.; Clementi, C. Commute maps: Separating slowly mixing molecular configurations for kinetic modeling. *J. Chem. Theory Comput.* **2016**, *12*, S620–S630.
- (133) Pérez-Hernández, G.; Paul, F.; Giorgino, T.; De Fabritiis, G.; Noé, F. Identification of slow molecular order parameters for Markov model construction. *J. Chem. Phys.* **2013**, *139*, 015102.
- (134) Schwantes, C. R.; Pande, V. S. Improvements in Markov state model construction reveal many non-native interactions in the folding of NTL9. *J. Chem. Theory Comput.* **2013**, *9*, 2000–2009.
- (135) Rohrdanz, M. A.; Zheng, W.; Maggioni, M.; Clementi, C. Determination of reaction coordinates via locally scaled diffusion map. *J. Chem. Phys.* **2011**, *134*, 124116.
- (136) Wang, J.; Gayatri, M. A.; Ferguson, A. L. Mesoscale simulation and machine learning of asphaltene aggregation phase behavior and molecular assembly landscapes. *J. Phys. Chem. B* **2017**, *121*, 4923–4944.
- (137) Zheng, W.; Rohrdanz, M. A.; Maggioni, M.; Clementi, C. Polymer reversal rate calculated via locally scaled diffusion map. *J. Chem. Phys.* **2011**, *134*, 144109.
- (138) Zheng, W.; Qi, B.; Rohrdanz, M. A.; Cafilisch, A.; Dinner, A. R.; Clementi, C. Delineation of folding pathways of a  $\beta$ -sheet miniprotein. *J. Phys. Chem. B* **2011**, *115*, 13065–13074.
- (139) Zheng, W.; Rohrdanz, M. A.; Clementi, C. Rapid exploration of configuration space with diffusion-map-directed molecular dynamics. *J. Phys. Chem. B* **2013**, *117*, 12769–12776.
- (140) Preto, J.; Clementi, C. Fast recovery of free energy landscapes via diffusion-map-directed molecular dynamics. *Phys. Chem. Chem. Phys.* **2014**, *16*, 19181–19191.
- (141) Chiavazzo, E.; Covino, R.; Coifman, R. R.; Gear, C. W.; Georgiou, A. S.; Hummer, G.; Kevrekidis, I. G. Intrinsic map dynamics exploration for uncharted effective free-energy landscapes. *Proc. Natl. Acad. Sci. U. S. A.* **2017**, *114*, E5494–E5503.
- (142) Boninsegna, L.; Gobbo, G.; Noé, F.; Clementi, C. Investigating molecular kinetics by variationally optimized diffusion maps. *J. Chem. Theory Comput.* **2015**, *11*, 5947–5960.
- (143) Ceriotti, M.; Tribello, G. A.; Parrinello, M. Simplifying the representation of complex free-energy landscapes using sketch-map. *Proc. Natl. Acad. Sci. U. S. A.* **2011**, *108*, 13023–13028.
- (144) Tribello, G. A.; Ceriotti, M.; Parrinello, M. Using sketch-map coordinates to analyze and bias molecular dynamics simulations. *Proc. Natl. Acad. Sci. U. S. A.* **2012**, *109*, 5196–5201.
- (145) De, S.; Bartók, A. P.; Csányi, G.; Ceriotti, M. Comparing molecules and solids across structural and alchemical space. *Phys. Chem. Chem. Phys.* **2016**, *18*, 13754–13769.
- (146) Musil, F.; De, S.; Yang, J.; Campbell, J. E.; Day, G. M.; Ceriotti, M. Machine learning for the structure-energy-property landscapes of molecular crystals. *Chem. Sci.* **2018**, *9*, 1289–1300.
- (147) van der Maaten, L.; Hinton, G. Visualizing data using t-SNE. *J. Mach. Learn. Res.* **2008**, *9*, 2579–2605.
- (148) Hinton, G. E.; Roweis, S. T. Stochastic neighbor embedding. *Adv. Neural. Inf. Process. Syst.* **2003**, 857–864.
- (149) Kullback, S.; Leibler, R. A. On information and sufficiency. *Ann. Math. Stat.* **1951**, *22*, 79–86.
- (150) Kobak, D.; Linderman, G. C. Initialization is critical for preserving global data structure in both t-SNE and UMAP. *Nat. Biotechnol.* **2021**, *39*, 156–157.
- (151) Spiwok, V.; Kříž, P. Time-lagged t-distributed stochastic neighbor embedding (t-SNE) of molecular simulation trajectories. *Front. Mol. Biosci.* **2020**, *7*, 132.
- (152) Zhou, H.; Wang, F.; Tao, P. t-Distributed Stochastic Neighbor Embedding Method with the Least Information Loss for Macromolecular Simulations. *J. Chem. Theory Comput.* **2018**, *14*, 5499–5510.
- (153) Zhou, H.; Wang, F.; Bennett, D. I.; Tao, P. Directed kinetic transition network model. *J. Chem. Phys.* **2019**, *151*, 144112.
- (154) Kramer, M. A. Nonlinear principal component analysis using autoassociative neural networks. *AIChE J.* **1991**, *37*, 233–243.
- (155) Lopez, R.; Regier, J.; Jordan, M. I.; Yosef, N. Information Constraints on Auto-Encoding Variational Bayes. *Adv. Neural. Inf. Process. Syst.* **2018**, *31*, 6114–6125.
- (156) Goodfellow, I.; Pouget-Abadie, J.; Mirza, M.; Xu, B.; Warde-Farley, D.; Ozair, S.; Courville, A.; Bengio, Y. Generative Adversarial Nets. *Adv. Neural. Inf. Process. Syst.* **2014**, *27*, 2672–2680.
- (157) Mardt, A.; Pasquali, L.; Wu, H.; Noé, F. VAMPnets for deep learning of molecular kinetics. *Nat. Commun.* **2018**, *9*, 5.
- (158) Hernández, C. X.; Waymunt-Steele, H. K.; Sultan, M. M.; Husic, B. E.; Pande, V. S. Variational encoding of complex dynamics. *Phys. Rev. E: Stat. Phys., Plasmas, Fluids, Relat. Interdiscip. Top.* **2018**, *97*, 062412.
- (159) Wehmeyer, C.; Noé, F. Time-lagged autoencoders: Deep learning of slow collective variables for molecular kinetics. *J. Chem. Phys.* **2018**, *148*, 241703.
- (160) Chen, W.; Ferguson, A. L. Molecular enhanced sampling with autoencoders: On-the-fly collective variable discovery and accelerated free energy landscape exploration. *J. Comput. Chem.* **2018**, *39*, 2079–2102.
- (161) Chen, W.; Tan, A. R.; Ferguson, A. L. Collective variable discovery and enhanced sampling using autoencoders: Innovations in network architecture and error function design. *J. Chem. Phys.* **2018**, *149*, 072312.
- (162) Ribeiro, J. M. L.; Bravo, P.; Wang, Y.; Tiwary, P. Reweighted autoencoded variational Bayes for enhanced sampling (RAVE). *J. Chem. Phys.* **2018**, *149*, 072301.
- (163) Shamsi, Z.; Cheng, K. J.; Shukla, D. Reinforcement learning based adaptive sampling: REAPing rewards by exploring protein conformational landscapes. *J. Phys. Chem. B* **2018**, *122*, 8386–8395.
- (164) Bonati, L.; Zhang, Y.-Y.; Parrinello, M. Neural networks-based variationally enhanced sampling. *Proc. Natl. Acad. Sci. U. S. A.* **2019**, *116*, 17641–17647.
- (165) Zhang, J.; Yang, Y. I.; Noé, F. Targeted adversarial learning optimized sampling. *J. Phys. Chem. Lett.* **2019**, *10*, 5791–5797.
- (166) Jung, H.; Covino, R.; Hummer, G. Artificial intelligence assists discovery of reaction coordinates and mechanisms from molecular dynamics simulations. *arXiv*, 2019, 1901.04595. <https://arxiv.org/abs/1901.04595>.
- (167) Bittracher, A.; Banisch, R.; Schütte, C. Data-driven computation of molecular reaction coordinates. *J. Chem. Phys.* **2018**, *149*, 154103.
- (168) Brandt, S.; Sittel, F.; Ernst, M.; Stock, G. Machine learning of biomolecular reaction coordinates. *J. Phys. Chem. Lett.* **2018**, *9*, 2144–2150.
- (169) Lemke, T.; Peter, C. Encodermap: Dimensionality reduction and generation of molecule conformations. *J. Chem. Theory Comput.* **2019**, *15*, 1209–1215.
- (170) Bozkurt Varolguş, Y.; Bereau, T.; Rudzinski, J. F. Interpretable embeddings from molecular simulations using Gaussian mixture variational autoencoders. *Mach. Learn.: Sci. Technol.* **2020**, *1*, 015012.
- (171) Sultan, M. M.; Waymunt-Steele, H. K.; Pande, V. S. Transferable neural networks for enhanced sampling of protein dynamics. *J. Chem. Theory Comput.* **2018**, *14*, 1887–1894.
- (172) Rezende, D.; Mohamed, S. Variational Inference with Normalizing Flows. *Proc. Mach. Learn. Res.* **2015**, *37*, 1530–1538.
- (173) Li, S.-H.; Wang, L. Neural network renormalization group. *Phys. Rev. Lett.* **2018**, *121*, 260601.
- (174) Nadaraya, E. A. On estimating regression. *Theory Probab. Its Appl.* **1964**, *9*, 141–142.
- (175) Steinwart, I.; Hush, D.; Scovel, C. A classification framework for anomaly detection. *J. Mach. Learn. Res.* **2005**, *6*, 211–232.
- (176) Westerlund, A. M.; Delemotte, L. InfileCS: Clustering free energy landscapes with Gaussian mixtures. *J. Chem. Theory Comput.* **2019**, *15*, 6752–6759.
- (177) Sittel, F.; Stock, G. Robust Density-Based Clustering To Identify Metastable Conformational States of Proteins. *J. Chem. Theory Comput.* **2016**, *12*, 2426–2435.
- (178) Lemke, O.; Keller, B. G. Density-based cluster algorithms for the identification of core sets. *J. Chem. Phys.* **2016**, *145*, 164104.



- (179) Sormani, G.; Rodriguez, A.; Laio, A. An explicit characterization of the free energy landscape of a protein in the space of all its  $\alpha$  carbons. *J. Chem. Theory Comput.* **2020**, *16*, 80–87.
- (180) Wu, H.; Noé, F. Gaussian Markov transition models of molecular kinetics. *J. Chem. Phys.* **2015**, *142*, 084104.
- (181) Silverman, B. W. *Density Estimation for Statistics and Data Analysis*; CRC Press, 1986; Vol. 26.
- (182) Dempster, A. P.; Laird, N. M.; Rubin, D. B. Maximum likelihood from incomplete data via the EM algorithm. *J. R. Stat. Soc. B* **1977**, *39*, 1–22.
- (183) McLachlan, G. J.; Rathnayake, S. On the number of components in a Gaussian mixture model. *WIREs Data Min. Knowl. Discovery* **2014**, *4*, 341–355.
- (184) Smyth, P. Model selection for probabilistic clustering using cross-validated likelihood. *Stat. Comput.* **2000**, *10*, 63–72.
- (185) Blei, D. M.; Jordan, M. I.; et al. Variational inference for Dirichlet process mixtures. *Bayesian Anal.* **2006**, *1*, 121–143.
- (186) Habeck, M. Bayesian estimation of free energies from equilibrium simulations. *Phys. Rev. Lett.* **2012**, *109*, 100601.
- (187) Westerlund, A. M.; Harpole, T. J.; Blau, C.; Delemotte, L. Inference of calmodulin's  $\text{Ca}^{2+}$ -dependent free energy landscapes via Gaussian mixture model validation. *J. Chem. Theory Comput.* **2018**, *14*, 63–71.
- (188) Maragakis, P.; van der Vaart, A.; Karplus, M. Gaussian-mixture umbrella sampling. *J. Phys. Chem. B* **2009**, *113*, 4664–4673.
- (189) Tribello, G. A.; Ceriotti, M.; Parrinello, M. A self-learning algorithm for biased molecular dynamics. *Proc. Natl. Acad. Sci. U. S. A.* **2010**, *107*, 17509–17514.
- (190) Laio, A.; Parrinello, M. Escaping free-energy minima. *Proc. Natl. Acad. Sci. U. S. A.* **2002**, *99*, 12562–12566.
- (191) Valsson, O.; Parrinello, M. Variational approach to enhanced sampling and free energy calculations. *Phys. Rev. Lett.* **2014**, *113*, 090601.
- (192) Moore, B. L.; Kelley, L. A.; Barber, J.; Murray, J. W.; MacDonald, J. T. High-quality protein backbone reconstruction from alpha carbons using Gaussian mixture models. *J. Comput. Chem.* **2013**, *34*, 1881–1889.
- (193) Zhou, R.; Berne, B. J.; Germain, R. The free energy landscape for  $\beta$  hairpin folding in explicit water. *Proc. Natl. Acad. Sci. U. S. A.* **2001**, *98*, 14931–14936.
- (194) Krivov, S. V.; Karplus, M. Hidden complexity of free energy surfaces for peptide (protein) folding. *Proc. Natl. Acad. Sci. U. S. A.* **2004**, *101*, 14766–14770.
- (195) Zhou, R. Trp-cage: Folding free energy landscape in explicit water. *Proc. Natl. Acad. Sci. U. S. A.* **2003**, *100*, 13280–13285.
- (196) Rodriguez, A.; Mokoema, P.; Corcho, F.; Bisetty, K.; Perez, J. J. Computational study of the free energy landscape of the miniprotein CLN025 in explicit and implicit solvent. *J. Phys. Chem. B* **2011**, *115*, 1440–1449.
- (197) Riccardi, L.; Nguyen, P. H.; Stock, G. Free-Energy Landscape of RNA Hairpins Constructed via Dihedral Angle Principal Component Analysis. *J. Phys. Chem. B* **2009**, *113*, 16660–16668.
- (198) Friedman, J. H. On bias, variance, 0/1-loss, and the curse-of-dimensionality. *Data Min. Knowl. Discovery* **1997**, *1*, 55–77.
- (199) Epanechnikov, V. A. Non-parametric estimation of a multivariate probability density. *Theory Probab. Its Appl.* **1969**, *14*, 153–158.
- (200) Li, Q.; Racine, J. S. *Nonparametric Econometrics: Theory and Practice*; Princeton University Press, 2006; Introductory Chapters.
- (201) Chen, Y.-C. A tutorial on kernel density estimation and recent advances. *Biostat. Epidemiol.* **2017**, *1*, 161–187.
- (202) Turlach, B. A. *Bandwidth Selection in Kernel Density Estimation: A Review*; CORE and Institut de Statistique, 1993.
- (203) Zambom, A. Z.; Dias, R. A review of kernel density estimation with applications to econometrics. *arXiv* 2012, 1212.2812. <https://arxiv.org/abs/1212.2812>.
- (204) Heidenreich, N.-B.; Schindler, A.; Sperlich, S. Bandwidth selection for kernel density estimation: a review of fully automatic selectors. *ASSTA Adv. Stat. Anal.* **2013**, *97*, 403–433.
- (205) Jones, M. C.; Marron, J. S.; Sheather, S. J. A brief survey of bandwidth selection for density estimation. *J. Am. Stat. Assoc.* **1996**, *91*, 401–407.
- (206) Scott, D. W. *Multivariate Density Estimation: Theory, Practice, and Visualization*; John Wiley & Sons, 2015.
- (207) Simonoff, J. S. *Smoothing Methods in Statistics*; Springer Science & Business Media, 2012.
- (208) Jones, M.; Marron, J. S.; Sheather, S. Progress in data-based bandwidth selection for kernel density estimation. *Comput. Stat.* **1996**, *11*, 337–381.
- (209) Lepskii, O. V. A problem of adaptive estimation in Gaussian white noise. *Theory Probab. Its Appl.* **1991**, *35*, 454–466.
- (210) Lepski, O. V.; Mammen, E.; Spokoiny, V. G. Optimal spatial adaptation to inhomogeneous smoothness: an approach based on kernel estimates with variable bandwidth selectors. *Ann. Statist.* **1997**, *25*, 929–947.
- (211) Polzehl, J.; Spokoiny, V. Image denoising: pointwise adaptive approach. *Ann. Statist.* **2003**, *31*, 30–57.
- (212) Polzehl, J.; Spokoiny, V. Propagation-separation approach for local likelihood estimation. *Probab. Theory Relat. Fields* **2006**, *135*, 335–362.
- (213) Gach, F.; Nickl, R.; Spokoiny, V. Spatially adaptive density estimation by localised Haar projections. *Ann. Inst. H. Poincaré Probab. Statist.* **2013**, *49*, 900–914.
- (214) Rebelles, G. Pointwise adaptive estimation of a multivariate density under independence hypothesis. *Bernoulli* **2015**, *21*, 1984–2023.
- (215) Becker, S. M. A.; Mathé, P. A different perspective on the Propagation-Separation Approach. *Electron. J. Statist.* **2013**, *7*, 2702–2736.
- (216) Bura, E.; Zhmurov, A.; Barsegov, V. Nonparametric density estimation and optimal bandwidth selection for protein unfolding and unbinding data. *J. Chem. Phys.* **2009**, *130*, 015102.
- (217) Henriques, J.; Cragnell, C.; Skepö, M. Molecular dynamics simulations of intrinsically disordered proteins: Force field evaluation and comparison with experiment. *J. Chem. Theory Comput.* **2015**, *11*, 3420–3431.
- (218) Berg, M. A.; Kaur, H. Nonparametric analysis of nonexponential and multidimensional kinetics. I. Quantifying rate dispersion, rate heterogeneity, and exchange dynamics. *J. Chem. Phys.* **2017**, *146*, 054104.
- (219) Weber, M.; Andrae, K. A simple method for the estimation of entropy differences. *MATCH Commun. Math. Comput. Chem.* **2010**, *63*, 319–332.
- (220) Mack, Y.; Rosenblatt, M. Multivariate  $k$ -nearest neighbor density estimates. *J. Multivar. Anal.* **1979**, *9*, 1–15.
- (221) Izenman, A. J. Recent developments in nonparametric density estimation. *J. Am. Stat. Assoc.* **1991**, *86*, 205–224.
- (222) Beyer, K.; Goldstein, J.; Ramakrishnan, R.; Shaft, U. When is “nearest neighbor” meaningful? *Proc. Int. Conf. Database Theory* **1999**, *1540*, 217–235.
- (223) Nagler, T.; Czado, C. Evading the curse of dimensionality in nonparametric density estimation with simplified vine copulas. *J. Multivar. Anal.* **2016**, *151*, 69–89.
- (224) Ozakin, A.; Gray, A. G. Submanifold density estimation. *Adv. Neural. Inf. Process. Syst.* **2009**, 1375–1382.
- (225) Campadelli, P.; Casiraghi, E.; Ceruti, C.; Rozza, A. Intrinsic dimension estimation: Relevant techniques and a benchmark framework. *Math. Probl. Eng.* **2015**, 759567.
- (226) Granata, D.; Carnevale, V. Accurate estimation of the intrinsic dimension using graph distances: Unraveling the geometric complexity of datasets. *Sci. Rep.* **2016**, *6*, 31377.
- (227) Facco, E.; d'Errico, M.; Rodriguez, A.; Laio, A. Estimating the intrinsic dimension of datasets by a minimal neighborhood information. *Sci. Rep.* **2017**, *7*, 12140.
- (228) Erba, V.; Gherardi, M.; Rotondo, P. Intrinsic dimension estimation for locally undersampled data. *Sci. Rep.* **2019**, *9*, 17133.
- (229) Lemke, O.; Keller, B. G. Common nearest neighbor clustering—a benchmark. *Algorithms* **2018**, *11*, 19.



- (230) Liu, S.; Zhu, L.; Sheong, F. K.; Wang, W.; Huang, X. Adaptive partitioning by local density-peaks: An efficient density-based clustering algorithm for analyzing molecular dynamics trajectories. *J. Comput. Chem.* **2017**, *38*, 152–160.
- (231) Hnizdo, V.; Darian, E.; Fedorowicz, A.; Demchuk, E.; Li, S.; Singh, H. Nearest-neighbor nonparametric method for estimating the configurational entropy of complex molecules. *J. Comput. Chem.* **2007**, *28*, 655–668.
- (232) Hnizdo, V.; Tan, J.; Killian, B. J.; Gilson, M. K. Efficient calculation of configurational entropy from molecular simulations by combining the mutual-information expansion and nearest-neighbor methods. *J. Comput. Chem.* **2008**, *29*, 1605–1614.
- (233) Galvelis, R.; Sugita, Y. Neural network and nearest neighbor algorithms for enhancing sampling of molecular dynamics. *J. Chem. Theory Comput.* **2017**, *13*, 2489–2500.
- (234) Fogolari, F.; Corazza, A.; Esposito, G. Free energy, enthalpy and entropy from implicit solvent end-point simulations. *Front. Mol. Biosci.* **2018**, *5*, 11.
- (235) Friedman, J.; Hastie, T.; Tibshirani, R. *The Elements of Statistical Learning*; Springer, 2001; Vol. 1.
- (236) Lloyd, S. Least squares quantization in PCM. *IEEE Trans. Inf. Theory* **1982**, *28*, 129–137.
- (237) Kaufman, L.; Rousseeuw, P. J. *Finding Groups in Data: An Introduction to Cluster Analysis*; John Wiley & Sons, 1991; Vol. 344.
- (238) Drineas, P.; Frieze, A.; Kannan, R.; Vempala, S.; Vinay, V. Clustering large graphs via the singular value decomposition. *Mach. Learn.* **2004**, *56*, 9–33.
- (239) Arthur, D.; Vassilvitskii, S. K-Means++: The advantages of careful seeding. *Proceedings of the Eighteenth Annual ACM-SIAM Symposium on Discrete Algorithms* **2007**, 1027–1035.
- (240) Sculley, D. Web-scale *k*-means clustering. *Proceedings of the 19th International Conference on World Wide Web* **2010**, 1177–1178.
- (241) Dunn, J. C. A fuzzy relative of the ISODATA process and its use in detecting compact well-separated clusters. *J. Cybernetics* **1973**, *3*, 32–57.
- (242) Bezdek, J. C. *Pattern Recognition with Fuzzy Objective Function Algorithms*; Springer Science & Business Media, 2013.
- (243) Schubert, E.; Rousseeuw, P. J. Faster *k*-medoids clustering: improving the PAM, CLARA, and CLARANS algorithms. *International Conference on Similarity Search and Applications* **2019**, 11807, 171–187.
- (244) Makles, A. Stata tip 110: How to get the optimal *k*-means cluster solution. *Stata J.* **2012**, *12*, 347–351.
- (245) Rousseeuw, P. J. Silhouettes: a graphical aid to the interpretation and validation of cluster analysis. *J. Comput. Appl. Math.* **1987**, *20*, 53–65.
- (246) Hartigan, J. A. *Clustering Algorithms*; John Wiley & Sons, Inc., 1975.
- (247) Gonzalez, T. F. Clustering to minimize the maximum intercluster distance. *Theor. Comput. Sci.* **1985**, *38*, 293–306.
- (248) Beauchamp, K. A.; Bowman, G. R.; Lane, T. J.; Maibaum, L.; Haque, I. S.; Pande, V. S. MSMBuilder2: Modeling conformational dynamics on the picosecond to millisecond scale. *J. Chem. Theory Comput.* **2011**, *7*, 3412–3419.
- (249) Daura, X.; Gademann, K.; Jaun, B.; Seebach, D.; van Gunsteren, W. F.; Mark, A. E. Peptide folding: When simulation meets experiment. *Angew. Chem., Int. Ed.* **1999**, *38*, 236–240.
- (250) Beauchamp, K. A.; Ensign, D. L.; Das, R.; Pande, V. S. Quantitative comparison of villin headpiece subdomain simulations and triplet-triplet energy transfer experiments. *Proc. Natl. Acad. Sci. U. S. A.* **2011**, *108*, 12734–12739.
- (251) Schwantes, C. R.; Pande, V. S. Improvements in Markov state model construction reveal many non-native interactions in the folding of NTL9. *J. Chem. Theory Comput.* **2013**, *9*, 2000–2009.
- (252) Lapidus, L. J.; Acharya, S.; Schwantes, C. R.; Wu, L.; Shukla, D.; King, M.; DeCamp, S. J.; Pande, V. S. Complex pathways in folding of protein G explored by simulation and experiment. *Biophys. J.* **2014**, *107*, 947–955.
- (253) Baiz, C. R.; Lin, Y.-S.; Peng, C. S.; Beauchamp, K. A.; Voelz, V. A.; Pande, V. S.; Tokmakoff, A. A molecular interpretation of 2D IR protein folding experiments with Markov state models. *Biophys. J.* **2014**, *106*, 1359–1370.
- (254) Noé, F.; Schütte, C.; Vanden-Eijnden, E.; Reich, L.; Weikl, T. R. Constructing the equilibrium ensemble of folding pathways from short off-equilibrium simulations. *Proc. Natl. Acad. Sci. U. S. A.* **2009**, *106*, 19011–19016.
- (255) Paul, F.; Wehmeyer, C.; Abualrous, E. T.; Wu, H.; Crabtree, M. D.; Schöneberg, J.; Clarke, J.; Freund, C.; Weikl, T. R.; Noé, F. Protein-peptide association kinetics beyond the seconds timescale from atomistic simulations. *Nat. Commun.* **2017**, *8*, 1095.
- (256) Paul, F.; Noé, F.; Weikl, T. R. Identifying conformational-selection and induced-fit aspects in the binding-induced folding of PMI from Markov state modeling of atomistic simulations. *J. Phys. Chem. B* **2018**, *122*, 5649–5656.
- (257) Sultan, M. M.; Kiss, G.; Pande, V. S. Towards simple kinetic models of functional dynamics for a kinase subfamily. *Nat. Chem.* **2018**, *10*, 903–909.
- (258) Husic, B. E.; Noé, F. Deflation reveals dynamical structure in nondominant reaction coordinates. *J. Chem. Phys.* **2019**, *151*, 054103.
- (259) Vanatta, D. K.; Shukla, D.; Lawrenz, M.; Pande, V. S. A network of molecular switches controls the activation of the two-component response regulator NtrC. *Nat. Commun.* **2015**, *6*, 7283.
- (260) Ford, L. R., Jr.; Fulkerson, D. R. *Flows in Networks*; Princeton University Press, 2015; Vol. 54.
- (261) Meila, M. . Spectral Clustering: A Tutorial for the 2010's *Handbook of Cluster Analysis*; 2016; pp 1–23.
- (262) Hagen, L.; Kahng, A. B. New spectral methods for ratio cut partitioning and clustering. *IEEE Trans. Comput.-Aided Des. Integr. Circuits Syst.* **1992**, *11*, 1074–1085.
- (263) Jianbo Shi; Malik, J. Normalized cuts and image segmentation. *IEEE Trans. Pattern Anal. Mach. Intell.* **2000**, *22*, 888–905.
- (264) Raykar, V. C. *Spectral Clustering and Kernel Principal Component Analysis Are Pursuing Good Projections*, Project Report; 2004.
- (265) Dhillon, I. S.; Guan, Y.; Kulis, B. Kernel *k*-means: spectral clustering and normalized cuts. *Proceedings of the Tenth ACM SIGKDD International Conference on Knowledge Discovery and Data Mining* **2004**, 551–556.
- (266) Von Luxburg, U. A tutorial on spectral clustering. *Stat. Comput.* **2007**, *17*, 395–416.
- (267) Phillips, J. L.; Colvin, M. E.; Newsam, S. Validating clustering of molecular dynamics simulations using polymer models. *BMC Bioinf.* **2011**, *12*, 445.
- (268) Sgourakis, N. G.; Merced-Serrano, M.; Boutsidis, C.; Drineas, P.; Du, Z.; Wang, C.; Garcia, A. E. Atomic-level characterization of the ensemble of the A $\beta$  (1–42) monomer in water using unbiased molecular dynamics simulations and spectral algorithms. *J. Mol. Biol.* **2011**, *405*, 570–583.
- (269) Phillips, J. L.; Colvin, M. E.; Lau, E. Y.; Newsam, S. Analyzing dynamical simulations of intrinsically disordered proteins using spectral clustering. *2008 IEEE International Conference on Bioinformatics and Biomedicine Workshops* **2008**, 17–24.
- (270) Sneath, P. H. The application of computers to taxonomy. *Microbiology* **1957**, *17*, 201–226.
- (271) Sorensen, T. A method of establishing groups of equal amplitude in plant sociology based on similarity of species content and its application to analyses of the vegetation on Danish commons. *Biol. Skr.* **1948**, *5*, 1–34.
- (272) Sokal, R. R.; Michener, C. D. A statistical method for evaluating systematic relationships. *Univ. Kans. Sci. Bull.* **1958**, *38*, 1409–1438.
- (273) Ward, J. H., Jr Hierarchical grouping to optimize an objective function. *J. Am. Stat. Assoc.* **1963**, *58*, 236–244.
- (274) Mirkin, B. Choosing the number of clusters. *WIREs Data Min. Knowl. Discovery* **2011**, *1*, 252–260.
- (275) Everitt, B. S.; Landau, S.; Leese, M.; Stahl, D. Hierarchical clustering. *Cluster Analysis* **2011**, *5*, 71–110.
- (276) Beauchamp, K. A.; McGibbon, R.; Lin, Y.-S.; Pande, V. S. Simple few-state models reveal hidden complexity in protein folding. *Proc. Natl. Acad. Sci. U. S. A.* **2012**, *109*, 17807–17813.

- (277) Husic, B. E.; Pande, V. S. Ward clustering improves cross-validated Markov state models of protein folding. *J. Chem. Theory Comput.* **2017**, *13*, 963–967.
- (278) Kamvar, S. D.; Klein, D.; Manning, C. D. Interpreting and extending classical agglomerative clustering algorithms using a model-based approach. *Int. Conf. Mach. Learn.* **2002**, *19*, 283–290.
- (279) Comaniciu, D.; Meer, P. Mean shift: a robust approach toward feature space analysis. *IEEE Trans. Pattern Anal. Mach. Intell.* **2002**, *24*, 603–619.
- (280) Lemmin, T.; Soto, C.; Stuckey, J.; Kwong, P. D. Microsecond dynamics and network analysis of the HIV-1 SOSIP Env trimer reveal collective behavior and conserved microdomains of the glycan shield. *Structure* **2017**, *25*, 1631–1639.
- (281) de Souza, V. C.; Goliatt, L.; Capriles Goliatt, P. V. Z. Clustering algorithms applied on analysis of protein molecular dynamics. *Latin American Conference on Computational Intelligence* **2017**, 1–6.
- (282) Ester, M.; Kriegel, H.-P.; Sander, J.; Xu, X.; et al. A density-based algorithm for discovering clusters in large spatial databases with noise. *KDD-96 Proc.* **1996**, *96*, 226–231.
- (283) Kriegel, H.-P.; Kröger, P.; Sander, J.; Zimek, A. Density-based clustering. *WIREs Data Min. Knowl. Discovery* **2011**, *1*, 231–240.
- (284) Singh, P.; Meshram, P. A. Survey of density based clustering algorithms and its variants. *International Conference on Inventive Computing and Informatics* **2017**, 920–926.
- (285) Ankerst, M.; Breunig, M. M.; Kriegel, H.-P.; Sander, J. OPTICS: ordering points to identify the clustering structure. *ACM Sigmod record* **1999**, *28*, 49–60.
- (286) Campello, R. J.; Moulavi, D.; Sander, J. Density-based clustering based on hierarchical density estimates. *Pacific-Asia Conference on Knowledge Discovery and Data Mining* **2013**, 7819, 160–172.
- (287) Campello, R. J.; Moulavi, D.; Zimek, A.; Sander, J. Hierarchical density estimates for data clustering, visualization, and outlier detection. *ACM Trans. Knowl. Discovery Data.* **2015**, *10*, 1–51.
- (288) Bergonzo, C.; Henriksen, N. M.; Roe, D. R.; Swails, J. M.; Roitberg, A. E.; Cheatham, T. E. Multidimensional replica exchange molecular dynamics yields a converged ensemble of an RNA tetranucleotide. *J. Chem. Theory Comput.* **2014**, *10*, 492–499.
- (289) Wang, K.; Chodera, J. D.; Yang, Y.; Shirts, M. R. Identifying ligand binding sites and poses using GPU-accelerated Hamiltonian replica exchange molecular dynamics. *J. Comput.-Aided Mol. Des.* **2013**, *27*, 989–1007.
- (290) Idrissi, A.; Vyalov, I.; Georgi, N.; Kiselev, M. On the characterization of inhomogeneity of the density distribution in supercritical fluids via molecular dynamics simulation and data mining analysis. *J. Phys. Chem. B* **2013**, *117*, 12184–12188.
- (291) Hong, C.; Tieleman, D. P.; Wang, Y. Microsecond molecular dynamics simulations of lipid mixing. *Langmuir* **2014**, *30*, 11993–12001.
- (292) Berg, A.; Franke, L.; Scheffner, M.; Peter, C. Machine learning driven analysis of large scale simulations reveals conformational characteristics of ubiquitin chains. *J. Chem. Theory Comput.* **2020**, *16*, 3205–3220.
- (293) Melvin, R. L.; Godwin, R. C.; Xiao, J.; Thompson, W. G.; Berenhaut, K. S.; Salsbury, F. R. Uncovering large-scale conformational change in molecular dynamics without prior knowledge. *J. Chem. Theory Comput.* **2016**, *12*, 6130–6146.
- (294) Chandramouli, B.; Melino, G.; Chillemi, G. Smyd2 conformational changes in response to p53 binding: role of the C-terminal domain. *Mol. Oncol.* **2019**, *13*, 1450–1461.
- (295) Melvin, R. L.; Xiao, J.; Godwin, R. C.; Berenhaut, K. S.; Salsbury, F. R., Jr Visualizing correlated motion with HDBSCAN clustering. *Protein Sci.* **2018**, *27*, 62–75.
- (296) Rodriguez, A.; Laio, A. Clustering by fast search and find of density peaks. *Science* **2014**, *344*, 1492–1496.
- (297) Mehta, P.; Bukov, M.; Wang, C.-H.; Day, A. G. R.; Richardson, C.; Fisher, C. K.; Schwab, D. J. A high-bias, low-variance introduction to Machine Learning for physicists. *Phys. Rep.* **2019**, *810*, 1–124.
- (298) Bai, L.; Cheng, X.; Liang, J.; Shen, H.; Guo, Y. Fast density clustering strategies based on the *k*-means algorithm. *Pattern Recognit.* **2017**, *71*, 375–386.
- (299) Liang, Z.; Chen, P. Delta-density based clustering with a divide-and-conquer strategy: 3DC clustering. *Pattern Recognit. Lett.* **2016**, *73*, 52–59.
- (300) Mehmood, R.; Zhang, G.; Bie, R.; Dawood, H.; Ahmad, H. Clustering by fast search and find of density peaks via heat diffusion. *Neurocomputing* **2016**, *208*, 210–217.
- (301) Du, M.; Ding, S.; Jia, H. Study on density peaks clustering based on *k*-nearest neighbors and principal component analysis. *Knowl.-Based Syst.* **2016**, *99*, 135–145.
- (302) Xie, J.; Gao, H.; Xie, W.; Liu, X.; Grant, P. W. Robust clustering by detecting density peaks and assigning points based on fuzzy weighted *k*-nearest neighbors. *Inf. Sci.* **2016**, *354*, 19–40.
- (303) Yaohui, L.; Zhengming, M.; Fang, Y. Adaptive density peak clustering based on *K*-nearest neighbors with aggregating strategy. *Knowl.-Based Syst.* **2017**, *133*, 208–220.
- (304) d'Errico, M.; Facco, E.; Laio, A.; Rodriguez, A. Automatic topography of high-dimensional data sets by non-parametric density peak clustering. *Inf. Sci.* **2021**, *560*, 476–492.
- (305) Liu, S.; Zhu, L.; Sheong, F. K.; Wang, W.; Huang, X. Adaptive partitioning by local density-peaks: An efficient density-based clustering algorithm for analyzing molecular dynamics trajectories. *J. Comput. Chem.* **2017**, *38*, 152–160.
- (306) Giorgino, T.; Laio, A.; Rodriguez, A. METAGUI 3: A graphical user interface for choosing the collective variables in molecular dynamics simulations. *Comput. Phys. Commun.* **2017**, *217*, 204–209.
- (307) Morrone, J. A.; Perez, A.; MacCallum, J.; Dill, K. A. Computed binding of peptides to proteins with MELD-accelerated molecular dynamics. *J. Chem. Theory Comput.* **2017**, *13*, 870–876.
- (308) Kahle, L.; Musaelian, A.; Marzari, N.; Kozinsky, B. Unsupervised landmark analysis for jump detection in molecular dynamics simulations. *Phys. Rev. Mater.* **2019**, *3*, 055404.
- (309) Pinamonti, G.; Paul, F.; Noé, F.; Rodriguez, A.; Bussi, G. The mechanism of RNA base fraying: Molecular dynamics simulations analyzed with core-set Markov state models. *J. Chem. Phys.* **2019**, *150*, 154123.
- (310) Molgedey, L.; Schuster, H. G. Separation of a mixture of independent signals using time delayed correlations. *Phys. Rev. Lett.* **1994**, *72*, 3634.
- (311) Meyer, C. D. *Matrix Analysis and Applied Linear Algebra*; Siam, 2000; Vol. 71.
- (312) Wu, H.; Nüske, F.; Paul, F.; Klus, S.; Koltai, P.; Noé, F. Variational Koopman models: slow collective variables and molecular kinetics from short off-equilibrium simulations. *J. Chem. Phys.* **2017**, *146*, 154104.
- (313) Harmeling, S.; Ziehe, A.; Kawanabe, M.; Müller, K.-R. Kernel-based nonlinear blind source separation. *Neural Comput.* **2003**, *15*, 1089–1124.
- (314) Schwantes, C. R.; Pande, V. S. Modeling molecular kinetics with tICA and the kernel trick. *J. Chem. Theory Comput.* **2015**, *11*, 600–608.
- (315) Naritomi, Y.; Fuchigami, S. Slow dynamics in protein fluctuations revealed by time-structure based independent component analysis: The case of domain motions. *J. Chem. Phys.* **2011**, *134*, 065101.
- (316) Pérez-Hernández, G.; Paul, F.; Giorgino, T.; De Fabritiis, G.; Noé, F. Identification of slow molecular order parameters for Markov model construction. *J. Chem. Phys.* **2013**, *139*, 015102.
- (317) Stanley, N.; Esteban-Martín, S.; De Fabritiis, G. Kinetic modulation of a disordered protein domain by phosphorylation. *Nat. Commun.* **2014**, *5*, 5272.
- (318) Litzinger, F.; Boninsegna, L.; Wu, H.; Nüske, F.; Patel, R.; Baraniuk, R.; Noé, F.; Clementi, C. Rapid calculation of molecular kinetics using compressed sensing. *J. Chem. Theory Comput.* **2018**, *14*, 2771–2783.
- (319) McCarty, J.; Parrinello, M. A variational conformational dynamics approach to the selection of collective variables in metadynamics. *J. Chem. Phys.* **2017**, *147*, 204109.



- (320) Sultan, M. M.; Wayment-Steele, H. K.; Pande, V. S. Transferable neural networks for enhanced sampling of protein dynamics. *J. Chem. Theory Comput.* **2018**, *14*, 1887–1894.
- (321) Noé, F.; Nuske, F. A variational approach to modeling slow processes in stochastic dynamical systems. *Multiscale Model. Simul.* **2013**, *11*, 635–655.
- (322) Noé, F.; Doose, S.; Daidone, I.; Löllmann, M.; Sauer, M.; Chodera, J. D.; Smith, J. C. Dynamical fingerprints for probing individual relaxation processes in biomolecular dynamics with simulations and kinetic experiments. *Proc. Natl. Acad. Sci. U. S. A.* **2011**, *108*, 4822–4827.
- (323) Gross, E. K.; Oliveira, L. N.; Kohn, W. Rayleigh-Ritz variational principle for ensembles of fractionally occupied states. *Phys. Rev. A: At, Mol., Opt. Phys.* **1988**, *37*, 2805.
- (324) Mezić, I. Spectral properties of dynamical systems, model reduction and decompositions. *Nonlinear Dyn.* **2005**, *41*, 309–325.
- (325) Fan, K. On a theorem of Weyl concerning eigenvalues of linear transformations I. *Proc. Natl. Acad. Sci. U. S. A.* **1949**, *35*, 652.
- (326) Husic, B. E.; Pande, V. S. Note: MSM lag time cannot be used for variational model selection. *J. Chem. Phys.* **2017**, *147*, 176101.
- (327) Sarich, M.; Noé, F.; Schütte, C. On the approximation quality of Markov state models. *Multiscale Model. Simul.* **2010**, *8*, 1154–1177.
- (328) McGibbon, R. T.; Pande, V. S. Variational cross-validation of slow dynamical modes in molecular kinetics. *J. Chem. Phys.* **2015**, *142*, 124105.
- (329) Wu, H.; Noé, F. Variational approach for learning Markov processes from time series data. *J. Nonlinear Sci.* **2020**, *30*, 23–66.
- (330) Nuske, F.; Keller, B. G.; Pérez-Hernández, G.; Mey, A. S.; Noé, F. Variational approach to molecular kinetics. *J. Chem. Theory Comput.* **2014**, *10*, 1739–1752.
- (331) Vitalini, F.; Noé, F.; Keller, B. A basis set for peptides for the variational approach to conformational kinetics. *J. Chem. Theory Comput.* **2015**, *11*, 3992–4004.
- (332) Chen, W.; Sidky, H.; Ferguson, A. L. Nonlinear discovery of slow molecular modes using state-free reversible VAMPnets. *J. Chem. Phys.* **2019**, *150*, 214114.
- (333) Zwanzig, R. From classical dynamics to continuous time random walks. *J. Stat. Phys.* **1983**, *30*, 255–262.
- (334) McGibbon, R. T.; Pande, V. S. Efficient maximum likelihood parameterization of continuous-time Markov processes. *J. Chem. Phys.* **2015**, *143*, 034109.
- (335) Noé, F. Probability distributions of molecular observables computed from Markov models. *J. Chem. Phys.* **2008**, *128*, 244103.
- (336) Wu, H.; Mey, A. S.; Rosta, E.; Noé, F. Statistically optimal analysis of state-discretized trajectory data from multiple thermodynamic states. *J. Chem. Phys.* **2014**, *141*, 214106.
- (337) Djurdjevac, N.; Sarich, M.; Schütte, C. On Markov state models for metastable processes. *International Congress of Mathematicians* **2010**, 3105–3131.
- (338) Bowman, G. R.; Pande, V. S.; Noé, F. *An Introduction to Markov State Models and their Application to Long Timescale Molecular Simulation*; Springer Science & Business Media, 2013; Vol. 797.
- (339) Bowman, G. R.; Voelz, V. A.; Pande, V. S. Atomistic folding simulations of the five-helix bundle protein  $\lambda$ 6-85. *J. Am. Chem. Soc.* **2011**, *133*, 664–667.
- (340) Voelz, V. A.; Bowman, G. R.; Beauchamp, K.; Pande, V. S. Molecular simulation of ab initio protein folding for a millisecond folder NTL9 (1–39). *J. Am. Chem. Soc.* **2010**, *132*, 1526–1528.
- (341) Deuffhard, P.; Huisinga, W.; Fischer, A.; Schütte, C. Identification of almost invariant aggregates in reversible nearly uncoupled Markov chains. *Linear Algebra Appl.* **2000**, *315*, 39–59.
- (342) Zhuang, W.; Cui, R. Z.; Silva, D. A.; Huang, X. Simulating the T-jump-triggered unfolding dynamics of trpzip2 peptide and its time-resolved IR and two-dimensional IR signals using the Markov state model approach. *J. Phys. Chem. B* **2011**, *115*, 5415–5424.
- (343) Andrec, M.; Felts, A. K.; Gallicchio, E.; Levy, R. M. Protein folding pathways from replica exchange simulations and a kinetic network model. *Proc. Natl. Acad. Sci. U. S. A.* **2005**, *102*, 6801–6806.
- (344) Schultheis, V.; Hirschberger, T.; Carstens, H.; Tavan, P. Extracting Markov models of peptide conformational dynamics from simulation data. *J. Chem. Theory Comput.* **2005**, *1*, 515–526.
- (345) Horenko, I.; Dittmer, E.; Fischer, A.; Schütte, C. Automated model reduction for complex systems exhibiting metastability. *Multiscale Model. Simul.* **2006**, *5*, 802–827.
- (346) Jensen, C. H.; Nerukh, D.; Glen, R. C. Sensitivity of peptide conformational dynamics on clustering of a classical molecular dynamics trajectory. *J. Chem. Phys.* **2008**, *128*, 115107.
- (347) Buchete, N.-V.; Hummer, G. Peptide folding kinetics from replica exchange molecular dynamics. *Phys. Rev. E* **2008**, *77*, 030902.
- (348) Sakuraba, S.; Kitao, A. Multiple Markov transition matrix method: Obtaining the stationary probability distribution from multiple simulations. *J. Comput. Chem.* **2009**, *30*, 1850–1858.
- (349) Schütte, C.; Noe, F.; Meerbach, E.; Metzner, P.; Hartmann, C. Conformation dynamics. *Proc. Int. Congr. ICIAM* **2009**, 297–336.
- (350) Metzner, P.; Schütte, C.; Vanden-Eijnden, E. Transition path theory for Markov jump processes. *Multiscale Model. Simul.* **2009**, *7*, 1192–1219.
- (351) Bacallado, S.; Chodera, J. D.; Pande, V. Bayesian comparison of Markov models of molecular dynamics with detailed balance constraint. *J. Chem. Phys.* **2009**, *131*, 045106.
- (352) Hummer, G. Position-dependent diffusion coefficients and free energies from Bayesian analysis of equilibrium and replica molecular dynamics simulations. *New J. Phys.* **2005**, *7*, 34.
- (353) Sorin, E. J.; Pande, V. S. Exploring the helix-coil transition via all-atom equilibrium ensemble simulations. *Biophys. J.* **2005**, *88*, 2472–2493.
- (354) Chodera, J. D.; Swope, W. C.; Pitera, J. W.; Dill, K. A. Long-time protein folding dynamics from short-time molecular dynamics simulations. *Multiscale Model. Simul.* **2006**, *5*, 1214–1226.
- (355) Park, S.; Klein, T. E.; Pande, V. S. Folding and misfolding of the collagen triple helix: Markov analysis of molecular dynamics simulations. *Biophys. J.* **2007**, *93*, 4108–4115.
- (356) Kelley, N. W.; Vishal, V.; Krafft, G. A.; Pande, V. S. Simulating oligomerization at experimental concentrations and long timescales: A Markov state model approach. *J. Chem. Phys.* **2008**, *129*, 214707.
- (357) Jayachandran, G.; Vishal, V.; Pande, V. S. Using massively parallel simulation and Markovian models to study protein folding: examining the dynamics of the villin headpiece. *J. Chem. Phys.* **2006**, *124*, 164902.
- (358) Voelz, V. A.; Luttmann, E.; Bowman, G. R.; Pande, V. S. Probing the nanosecond dynamics of a designed three-stranded beta-sheet with a massively parallel molecular dynamics simulation. *Int. J. Mol. Sci.* **2009**, *10*, 1013–1030.
- (359) Muff, S.; Caflisch, A. ETNA: equilibrium transitions network and Arrhenius equation for extracting folding kinetics from REMD simulations. *J. Phys. Chem. B* **2009**, *113*, 3218–3226.
- (360) Marinelli, F.; Pietrucci, F.; Laio, A.; Piana, S. A kinetic model of trp-cage folding from multiple biased molecular dynamics simulations. *PLoS Comput. Biol.* **2009**, *5*, No. e1000452.
- (361) Bowman, G. R.; Pande, V. S. Protein folded states are kinetic hubs. *Proc. Natl. Acad. Sci. U. S. A.* **2010**, *107*, 10890–10895.
- (362) Voelz, V. A.; Jäger, M.; Yao, S.; Chen, Y.; Zhu, L.; Waldauer, S. A.; Bowman, G. R.; Friedrichs, M.; Bakajin, O.; Lapidus, L. J.; et al. Slow unfolded-state structuring in Acyl-CoA binding protein folding revealed by simulation and experiment. *J. Am. Chem. Soc.* **2012**, *134*, 12565–12577.
- (363) Weber, J. K.; Jack, R. L.; Pande, V. S. Emergence of glass-like behavior in Markov state models of protein folding dynamics. *J. Am. Chem. Soc.* **2013**, *135*, 5501–5504.
- (364) Lane, T. J.; Schwantes, C. R.; Beauchamp, K. A.; Pande, V. S. Probing the origins of two-state folding. *J. Chem. Phys.* **2013**, *139*, 145104.
- (365) Dickson, A.; Brooks, C. L., III Native states of fast-folding proteins are kinetic traps. *J. Am. Chem. Soc.* **2013**, *135*, 4729–4734.
- (366) Guarnera, E.; Pellarin, R.; Caflisch, A. How does a simplified-sequence protein fold? *Biophys. J.* **2009**, *97*, 1737–1746.



- (367) Pietrucci, F.; Marinelli, F.; Carloni, P.; Laio, A. Substrate binding mechanism of HIV-1 protease from explicit-solvent atomistic simulations. *J. Am. Chem. Soc.* **2009**, *131*, 11811–11818.
- (368) Bujotzek, A.; Weber, M. Efficient simulation of ligand-receptor binding processes using the conformation dynamics approach. *J. Bioinf. Comput. Biol.* **2009**, *7*, 811–831.
- (369) Kohlhoff, K. J.; Shukla, D.; Lawrenz, M.; Bowman, G. R.; Konerding, D. E.; Belov, D.; Altman, R. B.; Pande, V. S. Cloud-based simulations on Google Exacycle reveal ligand modulation of GPCR activation pathways. *Nat. Chem.* **2014**, *6*, 15.
- (370) Malmstrom, R. D.; Kornev, A. P.; Taylor, S. S.; Amaro, R. E. Allostery through the computational microscope: cAMP activation of a canonical signalling domain. *Nat. Commun.* **2015**, *6*, 7588.
- (371) Barati Farimani, A.; Feinberg, E.; Pande, V. Binding pathway of opiates to  $\mu$ -opioid receptors revealed by machine learning. *Biophys. J.* **2018**, *114*, 62a–63a.
- (372) Bernetti, M.; Masetti, M.; Recanatini, M.; Amaro, R. E.; Cavalli, A. An integrated Markov state model and path metadynamics approach to characterize drug binding processes. *J. Chem. Theory Comput.* **2019**, *15*, 5689–5702.
- (373) Jagger, B. R.; Ojha, A. A.; Amaro, R. E. Predicting ligand binding kinetics using a Markovian milestoning with voronoi tessellations multiscale approach. *J. Chem. Theory Comput.* **2020**, *16*, 5348–5357.
- (374) Yang, S.; Roux, B. Src kinase conformational activation: thermodynamics, pathways, and mechanisms. *PLoS Comput. Biol.* **2008**, *4*, No. e1000047.
- (375) Yang, S.; Banavali, N. K.; Roux, B. Mapping the conformational transition in Src activation by cumulating the information from multiple molecular dynamics trajectories. *Proc. Natl. Acad. Sci. U. S. A.* **2009**, *106*, 3776–3781.
- (376) Liao, Q.; Kulkarni, Y.; Sengupta, U.; Petrović, D.; Mulholland, A. J.; van der Kamp, M. W.; Strodel, B.; Kamerlin, S. C. L. Loop motion in triosephosphate isomerase is not a simple open and shut case. *J. Am. Chem. Soc.* **2018**, *140*, 15889–15903.
- (377) Sriraman, S.; Kevrekidis, I. G.; Hummer, G. Coarse master equation from Bayesian analysis of replica molecular dynamics simulations. *J. Phys. Chem. B* **2005**, *109*, 6479–6484.
- (378) Voter, A. F. *Radiation Effects in Solids*; Springer, 2007; pp 1–23.
- (379) Weinan, E.; Vanden-Eijnden, E. Towards a theory of transition paths. *J. Stat. Phys.* **2006**, *123*, 503.
- (380) Silva, D.-A.; Bowman, G. R.; Sosa-Peinado, A.; Huang, X. A role for both conformational selection and induced fit in ligand binding by the LAO protein. *PLoS Comput. Biol.* **2011**, *7*, No. e1002054.
- (381) Sadiq, S. K.; Noé, F.; De Fabritiis, G. Kinetic characterization of the critical step in HIV-1 protease maturation. *Proc. Natl. Acad. Sci. U. S. A.* **2012**, *109*, 20449–20454.
- (382) Meng, Y.; Shukla, D.; Pande, V. S.; Roux, B. Transition path theory analysis of c-Src kinase activation. *Proc. Natl. Acad. Sci. U. S. A.* **2016**, *113*, 9193–9198.
- (383) Koltai, P.; Wu, H.; Noé, F.; Schütte, C. Optimal data-driven estimation of generalized Markov state models for non-equilibrium dynamics. *Computation* **2018**, *6*, 22.
- (384) Scherer, M. K.; Trendelkamp-Schroer, B.; Paul, F.; Pérez-Hernández, G.; Hoffmann, M.; Plattner, N.; Wehmeyer, C.; Prinz, J.-H.; Noé, F. PyEMMA 2: A software package for estimation, validation, and analysis of Markov models. *J. Chem. Theory Comput.* **2015**, *11*, 5525–5542.
- (385) Xie, T.; France-Lanord, A.; Wang, Y.; Shao-Horn, Y.; Grossman, J. C. Graph dynamical networks for unsupervised learning of atomic scale dynamics in materials. *Nat. Commun.* **2019**, *10*, 2667.
- (386) Lusch, B.; Kutz, J. N.; Brunton, S. L. Deep learning for universal linear embeddings of nonlinear dynamics. *Nat. Commun.* **2018**, *9*, 4950.
- (387) Otto, S. E.; Rowley, C. W. Linearly recurrent autoencoder networks for learning dynamics. *SIAM J. Appl. Dyn. Syst.* **2019**, *18*, 558–593.
- (388) Wehmeyer, C.; Scherer, M. K.; Hempel, T.; Husic, B. E.; Olsson, S.; Noé, F. Introduction to Markov state modeling with the PyEMMA software [Article v1.0]. *Living J. Comp. Mol. Sci.* **2019**, *1*, 5965.
- (389) McGibbon, R. T.; Beauchamp, K. A.; Harrigan, M. P.; Klein, C.; Swails, J. M.; Hernández, C. X.; Schwantes, C. R.; Wang, L.-P.; Lane, T. J.; Pande, V. S. MDTraj: a modern open library for the analysis of molecular dynamics trajectories. *Biophys. J.* **2015**, *109*, 1528–1532.
- (390) Lot, R.; Pellegrini, F.; Shaidu, Y.; Küçükbenli, E. Panna: Properties from artificial neural network architectures. *Comput. Phys. Commun.* **2020**, *256*, 107402.
- (391) Musil, F.; Veit, M.; Goscinski, A.; Fraux, G.; Willatt, M. J.; Stricker, M.; Junge, T.; Ceriotti, M. Efficient implementation of atom-density representations. *J. Chem. Phys.* **2021**, *154*, 114109.
- (392) Pedregosa, F.; Varoquaux, G.; Gramfort, A.; Michel, V.; Thirion, B.; Grisel, O.; Blondel, M.; Prettenhofer, P.; Weiss, R.; Dubourg, V.; et al. Scikit-learn: Machine learning in Python. *J. Mach. Learn. Res.* **2011**, *12*, 2825–2830.
- (393) Van Der Walt, S.; Colbert, S. C.; Varoquaux, G. The NumPy array: a structure for efficient numerical computation. *Comput. Sci. Eng.* **2011**, *13*, 22.
- (394) Jones, E.; Oliphant, T.; Peterson, P., et al. SciPy: Open source scientific tools for Python. <http://www.scipy.org/> (accessed 2019-04-19).
- (395) Hunter, J. D. Matplotlib: A 2D graphics environment. *Comput. Sci. Eng.* **2007**, *9*, 90–95.
- (396) Banisch, R.; Thiede, E. H.; Trstanova, Z. PyDiffMap Documentation. <https://pydiffmap.readthedocs.io/en/master/> (accessed 2019-04-19).
- (397) Ceriotti, M.; De, S.; Gasparo, P.; Meißner, R.; Tribello, G. SketchMap. *GitHub*. <https://github.com/cosmo-epfl/sketchmap> (accessed 2019-04-19).
- (398) Paszke, A.; Gross, S.; Massa, F.; Lerer, A.; Bradbury, J.; Chanan, G.; Killeen, T.; Lin, Z.; Gimelshein, N.; Antiga, L.; et al. PyTorch: An Imperative Style, High-Performance Deep Learning Library. *Adv. Neural Infor. Process. Syst.* **2019**, *32*, 8024–8035.
- (399) Abadi, M.; Agarwal, A.; Barham, P.; Brevdo, E.; Chen, Z.; Citro, C.; Corrado, G. S.; Davis, A.; Dean, J.; Devin, M. et al. TensorFlow: Large-scale machine learning on heterogeneous systems. <https://www.tensorflow.org/> (accessed 2019-04-19).
- (400) Chollet, F., et al. Keras. <https://github.com/fchollet/keras> (accessed 2019-04-19).
- (401) Wehmeyer, C., et al. Deeptime. *GitHub*. <https://github.com/markovmodel/deeptime> (accessed 2019-04-19).
- (402) Rodriguez, A. Advanced Density Peaks. *GitHub*. <https://github.com/alexdeprenia/Advanced-Density-Peaks> (accessed 2019-04-19).
- (403) D'errico, M.; Rodriguez, A.; Doni, G. DPA. *GitHub*. <https://github.com/mariaderrico/DPA> (accessed 2019-04-19).
- (404) Hoffmann, M., et al. Deeptime documentation. <https://deeptime-ml.github.io/> (accessed 2019-04-19).
- (405) Porter, J. R.; Zimmerman, M. I.; Bowman, G. R. Enspara: Modeling molecular ensembles with scalable data structures and parallel computing. *J. Chem. Phys.* **2019**, *150*, 044108.
- (406) Jung, J.; Nishima, W.; Daniels, M.; Bascom, G.; Kobayashi, C.; Adedoyin, A.; Wall, M.; Lappala, A.; Phillips, D.; Fischer, W.; et al. Scaling molecular dynamics beyond 100,000 processor cores for large-scale biophysical simulations. *J. Comput. Chem.* **2019**, *40*, 1919–1930.
- (407) Lindorff-Larsen, K.; Maragakis, P.; Piana, S.; Shaw, D. E. Picosecond to Millisecond Structural Dynamics in Human Ubiquitin. *J. Phys. Chem. B* **2016**, *120*, 8313–8320.
- (408) Ansari, N.; Laio, A.; Hassanali, A. Spontaneously forming dendritic voids in liquid water can host small polymers. *J. Phys. Chem. Lett.* **2019**, *10*, 5585–5591.
- (409) Hassanali, A. A.; Zhong, D.; Singer, S. J. An AIMD study of CPD repair mechanism in water: role of solvent in ring splitting. *J. Phys. Chem. B* **2011**, *115*, 3860–3871.
- (410) Pietrucci, F.; Saitta, A. M. Formamide reaction network in gas phase and solution via a unified theoretical approach: Toward a reconciliation of different prebiotic scenarios. *Proc. Natl. Acad. Sci. U. S. A.* **2015**, *112*, 15030–15035.
- (411) Bolhuis, P. G.; Dellago, C.; Chandler, D. Reaction coordinates of biomolecular isomerization. *Proc. Natl. Acad. Sci. U. S. A.* **2000**, *97*, 5877–5882.

(412) Mullen, R. G.; Shea, J.-E.; Peters, B. Transmission coefficients, committors, and solvent coordinates in ion-pair dissociation. *J. Chem. Theory Comput.* **2014**, *10*, 659–667.

(413) Schneider, E.; Dai, L.; Topper, R. Q.; Drechsel-Grau, C.; Tuckerman, M. E. Stochastic neural network approach for learning high-dimensional free energy surfaces. *Phys. Rev. Lett.* **2017**, *119*, 150601.

(414) Brünger, A. T.; Brooks, C. L.; Karplus, M. Active site dynamics of ribonuclease. *Proc. Natl. Acad. Sci. U. S. A.* **1985**, *82*, 8458–8462.

(415) Krizhevsky, A.; Sutskever, I.; Hinton, G. E. ImageNet classification with deep convolutional neural networks. *Adv. Neural. Inf. Process. Syst.* **2012**, *25*, 1097–1105.

#### NOTE ADDED AFTER ASAP PUBLICATION

This paper was published ASAP on May 4, 2021, with errors in equations 5 and 52. Equation 52 was corrected on May 5, 2021. Equation 5 was corrected on May 24, 2021.