Machine Learning Methods for Molecular Dynamic Simulations

Zineb Belkacemi

École des Ponts ParisTech

THÈSE DE DOCTORAT
de l'École des Ponts ParisTech

Deciphering protein function with artificial intelligence

École doctorale N°532, ÉCOLE DOCTORALE MATHÉMATIQUES ET STIC (MSTIC)

Mathématiques

Thèse préparée au CERMICS en collaboration avec Sanofi, R& D

Thèse soutenue le 06 Juillet 2022, par
Zineb BELKACEMI

Composition du jury:

Chris, CHIPOT Directeur de recherche, Université de l'Illinois	*Président du jury*
Stefan, KLUS Maître de conférences , Université de Surrey	*Rapporteur*
Peter, BOLHUIS Professeur, Université d'Amsterdam	*Rapporteur*
Wei, ZHANG Chargé de recherche, Zuse Institute Berlin	*Examinateur*
Elodie, LAINE Professeure associée , Sorbonne Université	*Examinateur*
Paraskevi, GKEKA Docteur, Sanofi, R& D	*Encadrant Industriel*
Tony, LELIEVRE Professeur, École nationale des ponts et chaussées	*Directeur de thèse*
Gabriel, STOLTZ Professeur , École nationale des ponts et chaussées	*Co-directeur de thèse*

Acknowledgments

First and foremost, I am thankful my PhD supervisors Tony Lelièvre, Gabriel Stoltz and Paraskevi Gkeka, without whom this work would not have been possible. Tony, your technical insight has been invaluable, either through your many papers or the innumerable chalkboard mini courses of the mathematics behind molecular dynamics. Your ever present good spirit and your jokes have helped me keep my confidence and morale through this journey. Evi, thank you for your patience through all of the questions I had about proteins and simulation software (especially VMD !!). Thank you also for your advice on writing and presentation, for your support, for the iced or hot coffees and all the insightful conversations we've had about machine learning molecular dynamics, but also about more personal matters. Gabriel, thank you for your unwavering presence and support, for always pushing me when you knew before I did how well I could do. Thank you for your patience and understanding whenever I needed it and for our many discussions. The three of you have shown me three very different but equally excellent ways of being a good supervisor.

I would like to thank Stefan Klus, and Peter Bolhuis who graciously agreed to be my thesis referees. Thank you as well to the remaining members of the jury, Chris Chipot, Wei Zhang, and Elodie Laine, for their time and insightful comments.

I am also grateful ANRT (Association Nationale de la Recherche et de la Technologie), Sanofi Aventis and the ENPC for the opportunity of this industrial thesis. Being able to do my PhD in collaboration with them has been perfect for me, a student eager to discover both worlds of research and of industry.

I have had the opportunity during this PhD to work with many amazing fellow PhD candidates, colleagues from Sanofi I would like to thank Jean Philippe Rameau, Herve Minoux and Marc Bianciotto for their time and insight. Thank you as well to Francesco, Maxime and all of the Sanofi IDD team. My thanks also go to the CERMICS team including Laura, Inass, Fred, Dylan, Pierre-Loik, Adel, Rafael, Lingling, Thomas, and so many more. I've also had the pleasure of encountering researchers from various institutions. Special thanks to Andrew Ferguson and Wei Chen for answering so many questions about MESA and to Massimiliano Bonomi for his insights on the workings of Plumed.

I am also truly grateful to the amazing administrative/HR staff of both Sanofi and CERMICS who have facilitated all the paperwork necessary for work permits, conferences, registrations, etc. A very special thank you to Isabelle Simunic.

My gratitude also goes out to my friends, all of whom have helped me through this journey, from sharing a drink at Le Gobelet or le Gob, watching a late NBA game together, or spending the afternoon playing board games. Thank you especially to Alex, Artiom, Drago, Flore, Florianne, Marius, Mathilde, William. Above all, I would like to thank you Etienne, for always being there, for your love and support, for always listening every time I had to vent about the code, the writing, the powerpoint presentations... I am beyond lucky to have you in my life.

I thank anyone and everyone whom I have had the pleasure to meet during these three years. I apologize in advance if I do not mention you in these acknowledgments, please know it was an involuntary oversight.

Finally, I thank all of my family, and especially my mother Zohra, father Ali, and brother and sister, Amine and Maria. Thank you for your unconditional love, for the laughs, for taking my mind off things whenever I need it.

Contents

Abstract

English

With the continually improving computational capacity of computers, machine learning methods have provided novel solutions to problems in a variety of fields. In particular, machine learning has been extensively used in the last decade in the field of computational biochemistry and drug discovery in virtually all stages of this field, such as defining new molecules, determining important sites in targeted proteins, designing adequate forcefields based on experimental results, or improving the efficiency of sampling molecular conformations of a given system. This thesis focuses on the latter task of using machine learning methods for enhanced sampling. More precisely, free energy biasing methods have proven to be powerful tools to accelerate the simulation of important conformational changes of molecules by modifying the sampling measure. However, most of these methods rely on the prior knowledge of low-dimensional slow degrees of freedom, i.e. collective variables. Alternatively, such low dimensional mappings can be identified using machine learning and dimensionality reduction algorithms. In addition to being used to accelerate sampling, the learned collective variables can also help acquire valuable insight into the studied system, namely by facilitating the visualization of the different states of the system, as well as its free energy landscape. In this work, important notions and definitions of molecular dynamics are first presented before reviewing state of the art machine learning algorithms which were devised or applied in the recent years for automatic collective variable discovery and enhanced sampling. Then, the method developed during this thesis, coined "free energy biasing and machine learning with autoencoders" (FEBILAE), is introduced. This method uses an iterative scheme to alternately generate new simulations and learn collective variables from these simulations using autoencoders. Finally, we present the application of machine learning methods to a real system of interest. Here, autoencoders are used to learn collective variables to perform biased simulations of the heat shock 90 (HSP90) chaperone protein.

Francais

Avec l'amélioration continue de la capacité de calcul des ordinateurs, les méthodes d'apprentissage automatique ont permis le développement de nouvelles solutions aux problèmes dans divers domaines. En particulier, l'apprentissage automatique a été largement utilisé au cours de la dernière décennie dans le domaine de la biochimie computationnelle et de la découverte et développement de nouveaux médicaments. Cela inclut l'application de méthodes d'apprentissage automatique pour la définition de nouvelles molécules, la détermination de sites impor-

tants dans les protéines ciblées, la conception de champs de force adéquats fondés sur des résultats expérimentaux ou encore l'amélioration de l'efficacité de l'échantillonnage des conformations moléculaires d'un système donné. Cette thèse de doctorat se concentre sur la dernière tâche consistant à utiliser des méthodes d'apprentissage automatique pour améliorer l'échantillonnage en dynamique moléculaire. En effet, les simulations de dynamique moléculaire se sont avérées être un outil très utile en complément des expériences en laboratoire. Malgré leur large utilisation pour capturer les phénomènes rapides, il existe encore de nombreux cas où les échelles de temps accessibles aux simulations de dynamique moléculaire sont bien plus petites que les échelles de temps nécessaires pour l'observation des changements conformationnels importants du système, en raison de la présence de barrières hautes dans le profil énergétique. Les méthodes de biaisage par l'énergie libre se sont avérées être des outils puissants pour accélérer l'observation de tels changements en modifiant la mesure d'échantillonnage. Cependant, la plupart de ces méthodes s'appuient sur la connaissance préalable de variable collective du système, c'est-à-dire des degrés de liberté de faible dimension représentant les directions lentes du système moléculaire. Ces variables collectives peuvent être identifiées à l'aide d'algorithmes d'apprentissage automatique et de réduction de dimensionalité. En plus d'être utilisées pour accélérer l'échantillonnage, les variables collectives construites par apprentissage automatique aident également à acquérir une connaissance précieuse du système étudié, à savoir en facilitant la visualisation de ses différents états, ainsi que de son profil d'énergie libre. Dans ce travail, d'importantes notions et définitions de la dynamique moléculaire sont d'abord présentées avant de passer en revue les algorithmes d'apprentissage automatique de pointe qui ont été conçus ou appliqués ces dernières années pour la construction automatique de variables collectives. Ensuite, la méthode développée au cours de cette thèse, baptisée "Free energy biasing and machine learning with autoencoders" (FEBILAE), est introduite. Cette méthode utilise un schéma itératif pour générer alternativement de nouvelles simulations et apprendre les variables collectives à partir de ces simulations en utilisant des autoencodeurs. Enfin, nous présentons l'application de méthodes d'apprentissage automatique à un véritable système d'intérêt. Ici, des autoencodeurs sont utilisés pour apprendre les variables collectives de la protéine chaperone HSP90, dans le but d'effectuer des simulations biaisées de ce système.

Chapter 1

Introduction

In the last decades, molecular dynamics (MD) simulations have helped gain insight into the microscopic and macroscopic properties of biomolecular processes. However, the time scales accessible to MD simulations are often significantly smaller than the times needed for the observation of slow conformational changes of the systems under study [1, 2]. This is due to the presence of energy or entropy traps in the energy landscape, which causes the system to be stuck within metastable states and thus hinders the full exploration of the configurational space. As a consequence, thermodynamic quantities (obtained from trajectorial averages) can not be accurately estimated.

To cope with this issue, several methods for enhanced sampling have been designed to mitigate the sampling difficulties associated with metastability [3, 4]. Most of these methods can be broadly divided into two categories according to whether or not they use collective variables (CV), also known as reaction coordinates, which are low dimensional or coarse-grained representations of the system. Collective variable free methods alter the canonical distribution by e.g. modifying the system temperature or the system Hamiltonian in order to accelerate crossing energetic or entropic barriers. This category includes, for example, simulated tempering [5], parallel tempering [6], replica exchange MD [7], multicanonical simulation [8], temperature-accelerated dynamics [9] or the Wang-Landau algorithm [10]. Collective variable based methods modify the system's dynamics by adding a bias in order to accelerate the dynamics by flattening the energy barriers along a chosen CV. Most of these methods simultaneously calculate the free energy associated to these CVs. Notable examples include metadynamics [11, 12], umbrella sampling [13, 14] and adaptive biasing force (ABF) methods [15–18]. The efficiency of these methods crucially relies on the prior knowledge of a proper low dimensional CV which contains most of the metastable degrees of freedom of the dynamics, and thus in particular clearly distinguishes between the metastable states. While in the case of simple systems these CVs may be determined from physical and chemical knowledge, and/or by trial and error, in most cases, guessing a good CV can prove challenging. Instead, data driven and machine learning methods can be used to determine suitable CVs by computing (linear or non linear) functions of the molecular configuration which optimize a given criterion based on metastability, state separation, etc.

This introduction is organized as follows:

- Section 1.1 first presents important notions of molecular dynamics, including elements of statistical physics, examples of thermodynamic ensembles, and dynamics equations. The section then moves on to definitions of collective variables and free energy, listing

the different existing methods for free energy computations, and detailing in particular ABF and extended system ABF, the methods used in this work. Finally, a discussion is made on optimality criteria for collective variables.

- Section 1.2 lists the state of the art methods devised for the automatic discovery of collective variables with machine learning algorithms. The methods are categorized according to the type of data they use: input conformations only, input conformations and their time evolution, or input conformations with known states.

- Section 1.3 finally summarizes our contributions in the field of machine learning based CV discovery: first, the method we propose in [19], coined free energy biasing and iterative learning with autoencoders (FEBILAE), presented in Chapter 2; and second, in Chapter 3, the application of machine learning algorithms for collective variables learning and biasing of a complex real life protein, the heat shock protein 90 (HSP90). An additional contribution, namely a review paper entitled "Machine learning force fields and coarse-grained variables in molecular dynamics: application to materials and biological systems" [20] is included in the appendix.

1.1 Molecular dynamics and free energy computations

In this section, we recall important elements and equations of statistical physics and molecular dynamics. We refer the interested reader to [21–25] for a more comprehensive presentation of the elements introduced in this section.

1.1.1 Elements of computational statistical physics

In the context of statistical physics applied to the study of molecular motions, the order of magnitude of the quantities under study are small enough to capture changes and motions at the atomic level. Taking into consideration the values of some key constants of statistical physics, such as the proton mass ($m_\mathrm{p} = 1.67 \times 10^{-27}$ kg), or the Boltzmann constant ($k_\mathrm{B} = 1.381 \times 10^{-23}$ J/K), the average values of some quantities can be inferred. Energies are thus of the order of $k_\mathrm{B}T$ (where T is the room temperature), distances are measured in Angstroms (1 Å $= 10^{-10}$ m), and time is measured in femtoseconds (1 fs $= 10^{-15}$ s). The description of a system at the microscopic level requires following the dynamics of every atom, at every timestep. For practical and computational reasons, this limits both the number of atoms and the time horizon that can be considered. In 1977, one of the first molecular simulations of a protein was reported [26]. It consisted of a 9.2 ps simulation of a small 58-residue protein, the bovine pancreatic trypsin inhibitor in vacuum (~ 500 atoms). Today, notably large simulations include a 500 μs simulation of the Villin headpiece in solvent involving $20,000$ atoms [27]; a 50 ns simulation of the complete satellite tobacco mosaic virus in solvent involving over 1 million atoms [28]; and the folding simulations of various proteins, over time horizons between 100 μs and 1ms involving $\sim 6,000$ to $\sim 35,000$ atoms [29]. All of these simulations, and others reported in recent years, always involve a very small number (often 1) of macromolecules, and a number of particles that is far from the Avogadro number $\mathcal{N}_A = 6.02 \times 10^{23}$. Nevertheless, it is observed that MD can still be used for computing macroscopic averages for a variety of quantities, yielding thermodynamic properties of the system, such as the free energy. This is particularly useful when these quantities cannot easily be computed from experiment, e.g. when a costly experimental setup is required.

Moreover, MD provides a temporal and spatial microscope into the motions of a given system, making it possible to link microscopic observations to the observations and intuitions one could have about the macroscopic behavior of the system.

1.1.1.A Molecular system description

A system composed of N particles is typically described as a vector (q, p), called the configuration of the system, where $q = (q_1, \dots, q_N) \in \mathcal{D}$ are the positions of the particle, and $p = (p_1, \dots, p_N) \in \mathbb{R}^{3N}$ are the associated momenta. The set $\mathcal{D}$ is defined by the boundary conditions applied to the system. Many different choices can be used. Let us describe three popular choices. The first one is to impose no boundary conditions, meaning the system is free to visit the full physical space, i.e. $\mathcal{D} = \mathbb{R}^{3N}$. This is commonly used when considering systems in vacuo. The second possibility is to impose periodic boundary conditions, where the simulated system is put inside a box, which is conceptually surrounded by translated copies of itself. This choice is the most commonly used with solvated macromolecules as it minimizes boundary effects. Each particle can interact with any other particle of the system, or with one of its periodic images. Of course, the cutoff r_c (for short-range interactions) and the smallest translation value L (over all directions) are set so as to ensure that a given particle may only interact with either the original particle or at most one of its periodic images, that is to say, $r_c < L/2$. Under such conditions, $\mathcal{D} = (L\mathbb{T})^{3N}$, where $\mathbb{T} = \mathbb{R}/\mathbb{Z}$ is the one-dimensional torus. The third choice is that of reflective boundary conditions, when the system is confined in a box and reflection rules are set for whenever the particle encounters the boundary. The set $\mathcal{D}$ is in this case defined by the choice of the box size.

The interactions between the particles are represented by a potential energy function $V : q \mapsto V(q)$. The total energy of the system is then given by the Hamiltonian:

$$H(q, p) = E_{\text{kin}}(p) + V(q), \tag{1.1}$$

where the kinetic energy E_{kin} is defined as:

$$E_{\text{kin}}(p) = \frac{1}{2} p^T M^{-1} p, \qquad M = \begin{pmatrix} m_1 I_3 & & 0 \\ & \ddots & \\ 0 & & m_N I_3 \end{pmatrix},$$

with m_i the mass of the ith particle.The Hamiltonian above is called separable, i.e. the q and p related energy terms can be separated. In the next section, we discuss possible choices for the potential energy function V, which is not as easily modeled as the kinetic energy.

1.1.1.B Interaction potentials

The ideal choice for the potential energy V is to derive it from first principles, by solving the Schrödinger equation for a given atomic configuration to determine its electronic ground-state energy and adding the Coulomb interactions between nuclei. Unfortunately, this so-called *ab initio* approach requires a considerable amount of computations, which can only be considered for very small systems. In the majority of cases, the *ab initio* potential is not an option and empirical potentials or forcefields are used instead. Empirical forcefields are given by an analytical form of the function V and a set of parameters involved in this analytical form [23]. Once the analytical form is defined, a parameterization protocol is set to optimize the parameters using either *ab initio* calculations, or available experimental data, e.g. by

matching computed quantities to their experimental values [30–32]. Usually the analytical function can be written by distinguishing the different interatomic interactions:

$$V = V_{\text{bonds}} + V_{\text{angles}} + V_{\text{torsions}} + V_{\text{improper}} + V_{\text{VdW}} + V_{\text{Coulomb}}. \tag{1.2}$$

Here we give for each component, a brief definition and a typical example of its analytical form. For this, we define for each pair of atoms (i, j), the vector $\mathbf{r}_{i,j} = q_i - q_j$, and the distance $r_{i,j} = |q_i - q_j|$.

- V_{bonds} is the sum over all pairs (i, j) of bonded atoms, of the covalent bond potentials, e.g. $v_{i,j} = \frac{1}{2}k_b(r_{i,i+1} - r_0)^2$.
- V_{angles} is the sum over all 3-tuples of atoms (i, j, k) such that the pairs (i, j) and (j, k) are bonded, of the angle bending potentials, e.g. $v_{i,j,k} = \dfrac{1}{2}k_a(\theta_{i,j,k} - \theta_0)^2$ or $v_{i,j,k} = \dfrac{1}{2}k_a(\cos(\theta_{i,j,k}) - \cos(\theta_0))^2$. Here, the angle $\theta_{i,j,k}$ can be defined from its cosine: $\cos(\theta_{i,j,k}) = \dfrac{\mathbf{r}_{i,j} \cdot \mathbf{r}_{j,k}}{r_{i,j}r_{j,k}}$.
- V_{torsions} is the sum over 4-tuples of consecutively bonded atoms (i, j, k, l) of the dihedral torsion potentials, e.g. $v_{i,j,k,l} = \frac{U_n}{2}(1 + \cos(n\phi_{i,j,k,l}))$. Here, the dihedral angle is defined from its cosine: $\cos(\phi_{i,j,k,l}) = -\dfrac{\mathbf{r}_{i,j} \times \mathbf{r}_{j,k}}{r_{i,j}r_{j,k}} \cdot \dfrac{\mathbf{r}_{j,k} \times \mathbf{r}_{k,l}}{r_{j,k}r_{k,l}}$.
- V_{improper} is the sum over 4-tuples of atoms (i, j, k, l) such that i is bonded to the other three atoms, of the improper torsion potentials , e.g. $v^{\text{imp}}_{i,j,k,l} = \dfrac{k_{\text{imp}}}{2}(1 + \cos(2\phi_{i,j,k,l}))$, or $v^{\text{imp}}_{i,j,k,l} = \dfrac{k_{\text{imp}}}{2}(\phi_{i,j,k,l} - \phi_0)^2$. Here $\cos(\phi_{i,j,k,l})$ is defined using the same equation as for dihedral torsions. This term is added to ensure the planarity of the group of atoms (i, j, k, l), see 23 for further details.
- V_{VdW} represents repulsive and Van der Waals interactions modeled e.g. by a Lennard Jones potential as: $V_{\text{VdW}} = \displaystyle\sum_{1 \le i < j \le N} 4\epsilon_{i,j}\left(\left(\frac{\sigma_{i,j}}{r_{i,j}}\right)^{12} - \left(\frac{\sigma_{i,j}}{r_{i,j}}\right)^{6}\right)$.
- V_{Coulomb} represents Coulomb interactions $V_c = \sum_{1 \le i < j \le N} \frac{Z_i Z_j}{r_{i,j}}$, where Z_i is the charge of particle i.

Some forcefield functions also include additional terms, such as: cross terms, i.e. terms involving two bonds, or a bond and an angle for example; terms that account for hydrogen bonds; or polarization terms [23]. According to the inclusions of these various terms, three classes can be distinguished to categorize most force fields.

- **Class I force fields:** These force fields typically have a classical analytical function of the form (1.2) without including any additional terms. Notable examples include CHARMM [33], AMBER [34], GROMOS [35] and OPLS [36].
- **Class II force fields:** This category of forcefields adds cross terms, i.e. terms that couple at least two intermolecular interactions (for example, bond-bond, bond-angle, or angle-torsion cross terms). These additional terms help improve conformational energy calculations, and observation of vibrational spectra. This class includes MM2 [37], MM3 [38], MM4 [39], MMFF [40], COMPASS [41], etc.

- **Class III force fields: polarizable force fields.** These forcefields include an explicit term for polarization using one of the three following methods: fluctuating charges, induced dipoles, or Drude particles. PIPF [42], DRF90 [43], or AMOEBA [44] are popular examples of this class. Classical software for forcefields such as CHARMM and AMBER have also developed new versions of polarized forcefields. Note that the addition of this term implies that the parameters of the potential function must be re-estimated for the force field to be more accurate.

In recent years, other classes and types of forcefields have been devised, such as reactive forcefields [45, 46], which are able to handle reactions such as bond formation and breaking, or machine learning (in particular deep networks) based forcefields [47–49].

1.1.1.C Thermodynamic ensembles

The macroscopic behaviour of the system, represented by a set of macroscopic quantites, can be described using a probability measure μ over the phase space $\mathcal{D} \times \mathbb{R}^{3N}$. Macroscopic quantities are computed as averages of an observable A over the phase space:

$$\mathbb{E}_\mu(A) = \int_{\mathcal{D}\times\mathbb{R}^{3N}} A(q,p)\mu(dq\ dp).$$

The thermodynamic ensemble of the system is fully defined by the choice of μ. Here, we present three main choices: the microcanonical ensemble NVE (fixed number of atoms, volume and total energy), the canonical ensemble NVT (fixed number of atoms, volume and temperature), and the isobaric-isothermal ensemble NPT (fixed number of atoms, pressure and temperature).

- **The microcanonical ensemble NVE** is the ensemble naturally associated with the Hamiltonian dynamics (see (1.5) below). It describes isolated systems with a fixed total energy E and a fixed volume. The corresponding probability measure is the uniform law over $S(E)$, the set of all $(q,p) \in \mathcal{D} \times \mathbb{R}^{3N}$ such that $H(q,p) = E$. The probability measure can be derived by a limiting procedure, involving a small energy variation ΔE, by introducing the set:

$$\mathcal{N}_{E,\Delta E} = \{(q,p) \in \mathcal{D} \times \mathbb{R}^{3N} \ \mid \ E \leq H(q,p) \leq E + \Delta E\}.$$

  The measure $\delta_{H(q,p)-E}(dq\ dp)$ can then be defined as follows: for any observable A,

$$\int_{S(E)} A(q,p)\delta_{H(q,p)-E}(dq\ dp) = \lim_{\Delta E \to 0} \frac{1}{\Delta E} \int_{\mathcal{N}_{E,\Delta E}} A(q,p) dq\ dp.$$

  Finally, the microcanonical measure is obtained by normalizing $\delta_{H(q,p)-E}(dq\ dp)$ to make it a probability measure (i.e. so that its integral sums to 1).

- **The canonical ensemble NVT** is an ensemble describing systems at a fixed temperature and volume, with energy fluctuating around an average value. We assume that $\mathrm{e}^{-\beta V} \in L^1(\mathcal{D})$, i.e $\mathrm{e}^{-\beta \mathrm{V}}$ is integrable over $\mathcal{D}$. The canonical probability measure μ is defined as:

$$\mu(dq\ dp) = Z_\mu^{-1} \mathrm{e}^{-\beta H(q,p)} dq\ dp = Z_\mu^{-1} \mathrm{e}^{-\beta V(q)} \mathrm{e}^{-\beta E_{\mathrm{kin}}(p)} dq\ dp, \tag{1.3}$$

where $\beta = (k_{\mathrm{B}}T)^{-1}$ and Z_μ is a normalization constant called the partition function. Because the Hamiltonian is separable, the canonical measure μ is a tensor product of two probability measures, in the variables p and q respectively. The momenta and positions are thus independent and can be sampled separately. The momenta p follow a Gaussian distribution, making their sampling straightforward. The positions q follow the more complex probability measure:

$$\nu(dq) = Z_\nu^{-1} \mathrm{e}^{-\beta V(q)} dq, \tag{1.4}$$

where Z_ν is a normalization constant.

- **The isobaric isothermic ensemble NPT** corresponds to simulations at fixed pressure and temperature where the volume and energy thus fluctuate around average values. A notable example is a periodic cubic box where the box size can vary to maintain a fixed pressure. For simplicity, we write the expression of the probability measure when the simulation domain is a rectangular box whose size can only vary in one direction. We denote by x the periodic box length in that direction, and by L the length of the remaining directions. Then,

$$\mathcal{D} = \mathcal{D}_x = (x\mathbb{T} \times (L\mathbb{T})^2)^N.$$

Using a uniform measure over all volumes as a reference *a priori* measure, and under the constraint that the volume and energy are fixed on average, the NPT measure is derived by maximizing the entropy with respect to the reference measure [22]. It reads:

$$\mu_{\mathrm{NPT}}(dq\ dp\ dx) = Z_{\mathrm{NPT}}^{-1}\ \mathrm{e}^{-\beta P L^2 x} \mathrm{e}^{-\beta H(q,p)} \mathbf{1}_{q \in \mathcal{D}_x} dq\ dp\ dx.$$

Other thermodynamic ensembles exist of course, such as the grand canonical ensemble μVT where the number of particles is only fixed on average. The most commonly used ensembles for macromolecules are the NVT and NPT ensembles.

1.1.1.D Dynamics

In this subsection, we briefly define three examples of dynamics used to sample the probability measures presented in Section 1.1.1.C. The considered examples are Hamiltonian dynamics, Langevin dynamics and overdamped Langevin dynamics. For each dynamics, key properties are listed. We refer the interested reader to [22] for additional details and proofs of these properties.

Hamiltonian dynamics. Hamiltonian dynamics are the adequate dynamics for describing isolated systems (NVE). The corresponding equations read:

$$\begin{cases} dq_t &= \nabla_p H(q_t, p_t) dt = M^{-1} p_t dt, \\ dp_t &= -\nabla_q H(q_t, p_t) dt = -\nabla V(q_t) dt, \end{cases} \tag{1.5}$$

with a provided initial state (q^0, p^0). The generator associated with this dynamics is:

$$\mathcal{L}_{\mathrm{ham}} = p^T M^{-1} \nabla_q - \nabla V^T \nabla_p. \tag{1.6}$$

We denote by ϕ_t the flow associated with the dynamics, i.e. $\phi_t(q_0, p_0) = (q_t, p_t)$. The flow ϕ_t satisfies the following properties:

- Symmetry: $\phi_t \circ \phi_{-t} = Id$.
- Energy conservation: $H \circ \phi_t = H$.
- Volume preservation: for any measurable set $B \subset \mathcal{D} \times \mathbb{R}^{3N}$, $\int_{\phi_t(B)} dq\ dp = \int_B dq\ dp$.
- Simplecticity: $(\nabla\phi_t)^T J(\nabla\phi_t) = J$ where $J = \begin{pmatrix} 0 & I_{3N} \\ -I_{3N} & 0 \end{pmatrix}$.
- Reversibility: Considering S the momentum reversal function $S(q,p) = (q,-p)$, it holds that $\phi_{-t} = S \circ \phi_t \circ S$.

Overdamped Langevin dynamics. Overdamped Langevin dynamics are stochastic dynamics which sample the position components of the canonical measure, under the equation:

$$dq_t = -\nabla V(q_t)dt + \sqrt{\frac{2}{\beta}}dW_t, \tag{1.7}$$

where W_t is a standard $3N$-dimensional Brownian process. Again, these dynamics can be formulated using the associated generator defined with maximal domain on $L^2(\nu)$:

$$\mathcal{L}_{\text{od}} = \frac{1}{\beta}\Delta - \nabla V^T \nabla. \tag{1.8}$$

The generator $\mathcal{L}_{\text{od}}$ is self-adjoint with respect to the canonical measure ν, which means that the dynamics are reversible. The canonical measure ν is invariant, i.e. for all smooth functions of compact support ϕ, $\int_{\mathcal{D}} \mathcal{L}_{\text{od}}\phi\ d\nu = 0$. Finally, the overdamped Langevin dynamics are ergodic with respect to the canonical measure, which means that for any observable $A \in L^1(\nu)$:

$$\lim_{T\to+\infty} \frac{1}{T}\int_0^T A(q_t)dt = \mathbb{E}_\nu(A) \quad \text{a.s.}$$

Langevin dynamics. Langevin dynamics are stochastic dynamics used to sample the canonical measure (positions and momenta).The dynamics are described by the equations:

$$\begin{cases} dq_t &= M^{-1}p_t dt, \\ dp_t &= -\nabla V(q_t)dt - \gamma M^{-1}p_t dt + \sqrt{\frac{2\gamma}{\beta}}dW_t, \end{cases} \tag{1.9}$$

where W_t is a standard $3N$-dimensional Brownian process and $\gamma > 0$ is the friction coefficient. Hamiltonian and overdamped Langevin dynamics can both be considered as special cases of Langevin dynamics [21]: Hamiltonian dynamics correspond to the case where the dynamics are underdamped, i.e. $\gamma = 0$; while overdamped Langevin are obtained when $\gamma \to \infty$ (with an appropriate time rescaling), or when the mass matrix $M = mI_{3N}$ and $m \to 0$. The infinitesimal generator of the Langevin dynamics, i.e. the second order differential operator associated with the dynamics, defined with maximal domain on $L^2(\mu)$ reads:

$$\mathcal{L} = \mathcal{L}_{\text{ham}} + \gamma\mathcal{L}_{\text{FD}}. \tag{1.10}$$

Here, $\mathcal{L}_{\text{ham}}$ is the generator of the Hamiltonian dynamics (defined in (1.6)), and $\mathcal{L}_{\text{FD}}$ represents the fluctuation dissipation part and is the generator of the Ornstein-Uhlenbeck process, which is a Gaussian process on the momenta:

$$\mathcal{L}_{\text{ham}} = \nabla_p H \cdot \nabla_q - \nabla_q H \cdot \nabla_p = p^T M^{-1} \nabla_q - \nabla V^T \nabla_p \ ,$$
$$\mathcal{L}_{\text{FD}} = -p^T M^{-1} \nabla_p + \frac{1}{\beta} \Delta_p = \frac{1}{\beta} \mathrm{e}^{\beta H} \operatorname{div}(\mathrm{e}^{-\beta H} \nabla_p).$$

The operator $\mathcal{L}_{\text{ham}}$ is antisymmetric while $\mathcal{L}_{\text{FD}}$ is symmetric with respect to the inner product on $L^2(\mu)$. Thus, the adjoint of $\mathcal{L}$ is:

$$\mathcal{L}^* = -\mathcal{L}_{\text{ham}} + \gamma \mathcal{L}_{\text{FD}}. \tag{1.11}$$

The canonical measure μ is an invariant probability measure. Indeed, it is easily checked that for all test functions ϕ,

$$\begin{aligned}\int_{\mathcal{D}\times\mathbb{R}^{3N}} \mathcal{L}\phi d\mu &= -\frac{1}{\beta} \int_{\mathcal{D}\times\mathbb{R}^{3N}} (\nabla_p \mathrm{e}^{-\beta H} \cdot \nabla_q \phi - \nabla_q \mathrm{e}^{-\beta H} \cdot \nabla_p \phi) \\ &\quad + \frac{\gamma}{\beta} \int_{\mathcal{D}\times\mathbb{R}^{3N}} \operatorname{div}(\mathrm{e}^{-\beta H} \nabla_p \phi) \\ &= 0.\end{aligned}$$

Additionally, the Langevin dynamics are ergodic for the canonical measure μ: for any $A \in L^1(\mu)$,

$$\lim_{T \to +\infty} \frac{1}{T} \int_0^T A(q_t, p_t) dt = \mathbb{E}_\mu(A) \quad \text{a.s.}$$

To describe the evolution of time averages, we define the semigroup of operators $\mathrm{e}^{\tau\mathcal{L}}$ defined by the generator $\mathcal{L}$: for any $\tau > 0$ and any smooth function φ,

$$\mathrm{e}^{\tau\mathcal{L}} \varphi(q^0, p^0) = \mathbb{E}_{(q^0,p^0)}[\varphi(q_\tau, p_\tau)]$$

Numerical discretization of Langevin dynamics. For all but trivial systems, the solutions to equations such as (1.9) cannot be analytically integrated, and need to be approximated by a numerical discretization in time: for a given small timestep Δt, for each timestep index k, a Markov process equation determining (q^{k+1}, p^{k+1}) from (q^k, p^k) is devised. Here, (q^k, p^k) is a discrete approximation of $(q_{k\Delta t}, p_{k\Delta t})$, the continuous process at time $k\Delta t$. For this, the Langevin operator $\mathcal{L}$ (see (1.10)) can be split into three sub-step operators. Indeed, $\mathcal{L}_{\text{ham}}$ can be decomposed as the sum of the position generator $\mathcal{L}_{\text{A}} = p^T M^{-1} \nabla_q$ and the momenta (or velocity) operator $\mathcal{L}_{\text{B}} = -\nabla V^T \nabla_p$. Both of these operators correspond to deterministic motions. The stochastic substep is represented by $\mathcal{L}_{\text{O}} = \gamma \mathcal{L}_{\text{FD}}$, the generator of the Ornstein-Uhlenbeck process. These three generators define elementary dynamics which are analytically integrable. For a given timestep Δt,

$$\text{A}: \qquad p^{k+1} = p^k, \qquad q^{k+1} = q^k + M^{-1} p^k \Delta t \tag{1.12}$$
$$\text{B}: \qquad p^{k+1} = p^k - \nabla V(q^k) \Delta t, \qquad q^{k+1} = q^k \tag{1.13}$$
$$\text{O}: \qquad p^{k+1} = \alpha_{\Delta t} p^k + \sqrt{\frac{(1 - \alpha_{\Delta t}^2) M}{\beta}} G^k, \qquad q^{k+1} = q^k \tag{1.14}$$

where $\{G^k\}_k$ is a sequence of independent standard normal random vectors and $\alpha_{\Delta t} = \mathrm{e}^{-\gamma M^{-1}\Delta t}$. The one-step evolution operator $\mathrm{e}^{\Delta t\mathcal{L}}$ can then be approximated as a Trotter factorization of the one-step operators $\mathrm{e}^{\Delta t\mathcal{L}_{\mathrm{A}}}$ and $\mathrm{e}^{\Delta t\mathcal{L}_{\mathrm{B}}}$ associated with the deterministic substeps, and the one-step evolution operator $\mathrm{e}^{\Delta t\mathcal{L}_{\mathrm{O}}}$ associated with the stochastic substep. Many discretization schemes can then be defined, depending on the order in which these substeps are performed. Often the order is chosen so as to have a symmetrical factorization. For example, the "BAOAB" scheme [50, 51] can be written using the symmetric Strang splitting [52]:

$$\mathrm{e}^{\Delta t\mathcal{L}} = \mathrm{e}^{\frac{\Delta t}{2}\mathcal{L}_{\mathrm{B}}}\mathrm{e}^{\frac{\Delta t}{2}\mathcal{L}_{\mathrm{A}}}\mathrm{e}^{\Delta t\mathcal{L}_{\mathrm{O}}}\mathrm{e}^{\frac{\Delta t}{2}\mathcal{L}_{\mathrm{A}}}\mathrm{e}^{\frac{\Delta t}{2}\mathcal{L}_{\mathrm{B}}}.$$

and thus corresponds to the equations:

$$\begin{cases} p^{k+1/4} &= p^k - \nabla V(q^k)\Delta t/2, \\ q^{k+1/2} &= q^k + M^{-1}p^{k+1/4}\Delta t/2, \\ p^{k+3/4} &= \alpha_{\Delta t}p^{k+1/4} + \sqrt{\dfrac{(1-\alpha_{\Delta t}^2)M}{\beta}}G^k, \\ q^{k+1} &= q^{k+1/2} + M^{-1}p^{k+3/4}\Delta t/2, \\ p^{k+1} &= p^{k+3/4} - \nabla V(q^{k+1})\Delta t/2. \end{cases}$$

Here, $\alpha_{\Delta t} = \mathrm{e}^{-\gamma M^{-1}\Delta t}$. For the remainder of Section 1.1, we assume that the system follows Langevin dynamics (or overdamped Langevin when considering positions only), and therefore samples the canonical measure.

1.1.2 Collective variables and free energy

The free energy of a system is an essential quantity in MD and thermodynamics. Its estimation for a given system is the central point of many biomolecular studies [53]. We distinguish between the absolute free energy, and the free energy difference between states, where states are defined either by the value of a parameter λ of the Hamiltonian (the alchemical case), or by the value of a given function $\xi(q,p) = z$ of the phase space, called a reaction coordinate or a collective variable (the reaction coordinate case).

1.1.2.A The absolute free energy

We start by recalling the definition of the absolute free energy.

Definition 1. *The absolute free energy of a system is defined as a logarithm of the partition function Z_μ. For the canonical ensemble, the absolute free energy reads:*

$$F = -\frac{1}{\beta}\ln Z_\mu = -\frac{1}{\beta}\ln\int_{\mathcal{D}\times\mathbb{R}^{3N}} e^{-\beta H(q,p)}dq\ dp.$$

The quantity F is also called the Helmholtz free energy. In the isobaric isothermal ensemble, the free energy is instead called the Gibbs free energy. This definition is actually in accordance with the thermodynamics definition of the free energy, $F = U - TS$ where U is the internal energy and S the entropy. This can indeed be checked by replacing each of these quantities by its microscopic equivalent [22], namely the average energy $U_{\mathrm{NVT}} = \mathbb{E}_\mu(H)$ for U, and the entropy $S_{\mathrm{NVT}} = -k_B\int_{\mathcal{D}\times\mathbb{R}^{3N}}\frac{\mathrm{e}^{-\beta H(q,p)}}{Z_\mu}\ln\left(\frac{\mathrm{e}^{-\beta H(q,p)}}{Z_\mu}\right)dq\ dp$ for S. The

absolute free energy is a quantity describing the system as a whole, but in most studies of biomolecular processes, an important goal is to analyse differences, particularly free energy differences, between various states within the same system, so that one is only interested in the free energy up to an additive constant. A simple example is the computation of the free energy difference between folded and unfolded states of a given protein, or the bound versus unbound states of a ligand-protein system.

1.1.2.B The relative alchemical free energy

In the alchemical case, a state is defined as the value of a parameter λ involved in the definition of the Hamiltonian H_λ, and thus the associated canonical measure μ_λ. In practice, λ is a parameter of the forcefield potential function V_λ, hence the term "alchemical": the nature of the particles changes according to the value of λ. This represents a physically impossible transition, but it is feasible in computer simulations.

Definition 2. *The free energy difference between states $\lambda = \lambda_1$ and $\lambda = \lambda_2$ can be computed as:*

$$F(\lambda_2) - F(\lambda_1) = -\frac{1}{\beta} \ln \frac{Z_{\mu_2}}{Z_{\mu_1}}, \tag{1.15}$$

where $Z_{\mu_i} = \int_{\mathcal{D}\times\mathbb{R}^{3N}} \mathrm{e}^{-\beta H_{\lambda_i}}$. Note that a simple computation makes it possible to write the fraction in the above equation as an average quantity with respect to the law μ_1:

$$\begin{aligned} F(\lambda_2) - F(\lambda_1) = -\frac{1}{\beta} \ln \frac{Z_{\mu_2}}{Z_{\mu_1}} &= -\frac{1}{\beta} \ln \left(\frac{\displaystyle\int_{\mathcal{D}\times\mathbb{R}^{3N}} \mathrm{e}^{-\beta H_{\lambda_2}(q,p)} dq\ dp}{\displaystyle\int_{\mathcal{D}\times\mathbb{R}^{3N}} \mathrm{e}^{-\beta H_{\lambda_1}(q,p)} dq\ dp} \right) \\ &= -\frac{1}{\beta} \ln \left[\mathbb{E}_{\mu_1} \left(\mathrm{e}^{-\beta (H_{\lambda_2}(q,p) - H_{\lambda_1}(q,p))} \right) \right]. \end{aligned} \tag{1.16}$$

The last equality is easily obtained by rewriting the integrand of the integral in the numerator as $\mathrm{e}^{-\beta(H_{\lambda_2}(q,p)-H_{\lambda_1}(q,p))}\mathrm{e}^{-\beta(H_{\lambda_1}(q,p)}$. The free energy difference can thus be computed as the μ_1-average of the quantity $\mathrm{e}^{-\beta(H_{\lambda_2}(q,p)-H_{\lambda_1}(q,p))}$. This is the basis for free energy perturbation methods, which we define in Section 1.1.3.A. Alternatively, the free energy difference can be computed by integrating the derivative $F'(\lambda)$, which reads:

$$F'(\lambda) = \mathbb{E}_{\mu_\lambda} \left(\frac{\partial H_\lambda}{\partial \lambda} \right). \tag{1.17}$$

We defined in Section 1.1.3.B thermodynamic integration methods, which integrate the derivative $F'(\lambda)$ of the free energy.

1.1.2.C The relative free energy of a reaction coordinate

In the reaction coordinate case, a state is defined using the value of a reaction coordinate (RC), also called a collective variable (CV). A reaction coordinate is a function ξ of the phase space which contains information on the overall structure of the system. In most cases, it actually only depends on the positions q, so that $\xi : \mathcal{D} \to \mathcal{A} \subset \mathbb{R}^d$ where in practice $d \ll 3N$. Classical examples are combinations of well defined functions of the molecule, such as distances between residues, or dihedral angles. The coordinate ξ is always assumed to satisfy $\mathrm{rank}(\nabla\xi) = d$ unless specified otherwise. A state z is then the collection of all configurations q with the same value of $\xi(q) = z$.

Definition 3. *The free energy of state z is:*

$$F(z) = -\frac{1}{\beta} \ln \left(\frac{\int_{\Sigma(z)\times\mathbb{R}^{3N}} e^{-\beta H(q,p)} \delta_{\xi(q)-z}(dq)dp}{Z_\mu} \right),$$

where

$$\Sigma(z) = \{q \in \mathcal{D} \mid \xi(q) = z\}.$$

The measure $\delta_{\xi(q)-z}(dq)$, supported by $\Sigma(z)$ satisfies $\delta_{\xi(q)-z}(dq)dz = dq$. It can also be defined using the co-area formula (Equation (3.14) in Ref. 22) as:

$$\delta_{\xi(q)-z}(dq) = \frac{\sigma_{\Sigma(z)}(dq)}{\sqrt{\det(G)}},$$

where $\sigma_{\Sigma(z)}(dq)$ is the surface measure over the manifold $\Sigma(z)$ when $\mathcal{D}$ is equipped with the standard Euclidean scalar product, and

$$G = (\nabla\xi)^T \nabla\xi \tag{1.18}$$

is the so-called Gram matrix. Note that the free energy F is related to the marginal distribution μ^ξ of the canonical measure μ along the coordinate ξ:

$$\mu^\xi(dz) = \left(\frac{\int_{\Sigma(z)\times\mathbb{R}^{3N}} \mathrm{e}^{-\beta H(q,p)} \delta_{\xi(q)-z}(dq)dp}{Z_\mu} \right) dz = \mathrm{e}^{-\beta F(z)} dz.$$

The free energy can thus be viewed as a coarse grained potential function over the coordinate $\xi(q)$. Note also that, because the considered Hamiltonian is separable, the kinetic energy term can be taken out of the free energy definition:

$$F(z) = -\frac{1}{\beta} \ln \left(\frac{\int_{\Sigma(z)} \mathrm{e}^{-\beta V(q)} \delta_{\xi(q)-z}(dq)}{Z_\nu} \right), \qquad Z_\nu = \int_{\mathcal{D}} \mathrm{e}^{-\beta V(q)} dq. \tag{1.19}$$

The free energy difference between states $z = z_1$ and $z = z_2$ is thus:

$$F(z_2) - F(z_1) = -\frac{1}{\beta} \ln \left(\frac{\int_{\Sigma(z_2)} \mathrm{e}^{-\beta V(q)} \delta_{\xi(q)-z_2}(dq)}{\int_{\Sigma(z_1)} \mathrm{e}^{-\beta V(q)} \delta_{\xi(q)-z_1}(dq)} \right).$$

This free energy difference allows to quantify the relative likelihoods of the configurations of state z_1 to those of state z_2. The derivative of the free energy, called the mean force, reads:

$$\nabla F(z) = \frac{\int_{\Sigma(z)} f(q) \mathrm{e}^{-\beta V(q)} \delta_{\xi(q)-z}(dq)}{\int_{\Sigma(z)} \mathrm{e}^{-\beta V(q)} \delta_{\xi(q)-z}(dq)} = \int_{\Sigma(z)} f(q) \nu^\xi(dq|z), \tag{1.20}$$

where $\nu^\xi(.|z)$ is the measure ν conditioned to $\xi(q) = z$. The d-dimensional vector function $f = (f_1, \dots, f_d)$ is called the local mean force. Its analytical expression reads:

$$f_i = \sum_{j=1}^{d} (G^{-1})_{i,j} \nabla \xi_j \cdot \nabla V - \frac{1}{\beta} \operatorname{div} \left(\sum_{j=1}^{d} (G^{-1})_{i,j} \nabla \xi_j \right), \tag{1.21}$$

where, we recall, G is defined in (1.18).

1.1.2.D Reaction coordinates, metastability and free energy biasing

The choice of the reaction coordinate ξ greatly impacts the relevance of the free energy differences we want to compute. As mentioned above, the states we want to define represent the gradual folding of a protein, or its binding to a ligand for example. Thus the coordinate ξ should be chosen so as to describe these motions. In most cases, the dynamics of the system under study are metastable, and the motions of interest are in fact rarely occurring transitions between different metastable states (e.g. folded and misfolded states of a protein). This is caused by the shape of the energy function, which contains local minima or entropic traps (metastable states) separated by free energy barriers (transition states). This implies that the coordinate ξ we are looking for is a slowly varying degree of freedom whose values distinguish between the different states. The process $(\xi(q_t))_{t \geq 0}$ is thus also metastable, i.e. its values may stay trapped inside some region of the space $\mathbb{R}^d$ before crossing to another region, marking a transition of the system from one metastable state to another. Classical examples of reaction coordinates are combinations of well defined simple functions of the positions q, such as distances, dihedrals or contacts.

Note that the metastable character of the dynamics implies that the full exploration of the configurational space takes a lot of time, so that sampling the canonical measure using the dynamics defined in (1.7) or (1.9) raises computational difficulties. In this context comes an important utilization of the relative free energy: biased sampling. Indeed, the free energy along a coordinate ξ can be used to bias the potential of the system so as to make the process $(\xi(q_t))_{t \geq 0}$ no longer metastable. More precisely, we consider a system evolving under the potential $V - F \circ \xi$ instead of V. The Langevin dynamics under this potential become:

$$\begin{cases} dq_t &= M^{-1} p_t dt, \\ dp_t &= -\nabla (V(q_t) - F \circ \xi(q_t)) dt - \gamma M^{-1} p_t dt + \sqrt{\frac{2\gamma}{\beta}} dW_t. \end{cases} \tag{1.22}$$

The marginal distribution $\tilde{\mu}^\xi$ of the new canonical measure $\tilde{\mu}$ along ξ is then, up to a multiplicative constant:

$$\begin{aligned} \tilde{\mu}^\xi(dz) &= \left(\int_{\Sigma(z)} \mathrm{e}^{-\beta(V(q) - F \circ \xi(q))} \delta_{\xi(q) - z}(dq) \right) dz \\ &= \left(\int_{\Sigma(z)} \mathrm{e}^{-\beta(V(q) - F(z))} \delta_{\xi(q) - z}(dq) \right) dz = Z_\nu dz. \end{aligned}$$

The marginal distribution along ξ under the potential $V - F \circ \xi$ is the uniform measure. Under the new dynamics, the coordinate ξ is no longer metastable. This motivates the importance of the choice of the reaction coordinate: The biased potential $V - F \circ \xi$ is only as effective at sampling metastable motions of interest as ξ is at describing them. Figure 1.1 shows

a simple illustration of biased sampling: The example is a 2 dimensional 3-well potential system $q = (x_1, x_2)$, for which the first coordinate x_1 is enough to distinguish the 3 states, and is thus used as a CV to bias the dynamics. A general discussion on optimality criteria for reaction coordinates is given below in Section 1.1.5.

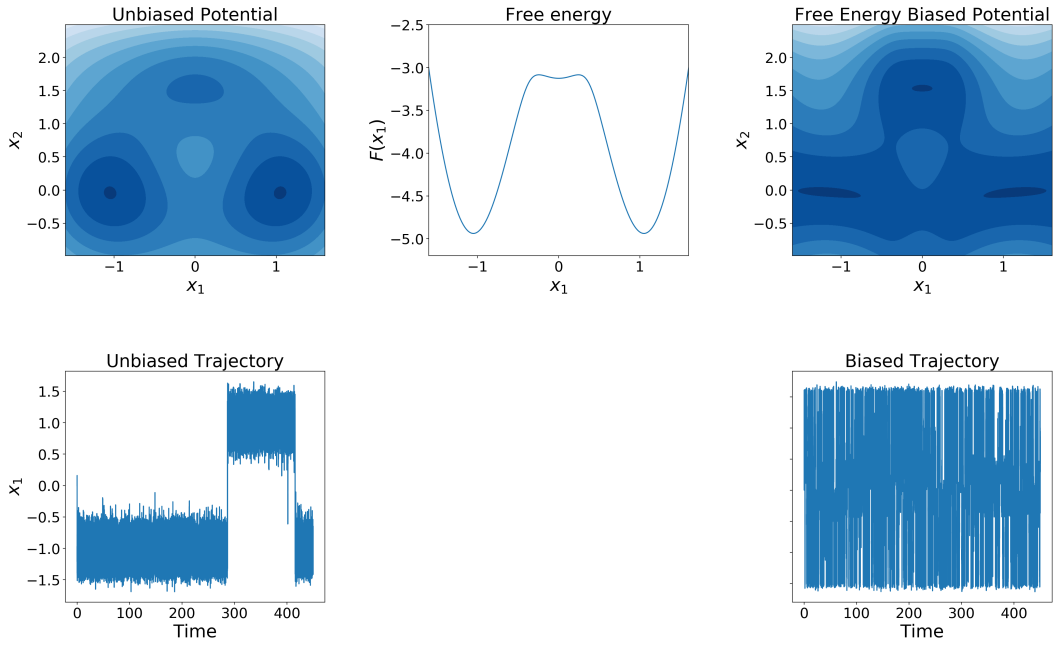


Figure 1.1: Free energy biasing example. The biased potential enables more frequent transitions between the two deep metastable states.

1.1.3 Computing free energy differences

The free energy differences defined in Section 1.1.2 cannot in most cases be analytically computed and instead needs to be estimated using averages over sampled trajectories. In this section, we list some of the methods used for computing free energy differences. We distinguish between four classes of methods which we describe in each of the following sections: Free energy perturbation and histogram methods, thermodynamic integration, non-equilibrium methods and finally adaptive methods.

1.1.3.A Straightforward sampling methods

Free energy perturbation. Free energy perturbation methods allow to compute free energy differences in the alchemical case. The method was first introduced by Zwanzig in [54]. It relies on the expression of the free energy difference given in Equation (1.16):

$$F(\lambda_2) - F(\lambda_1) = -\frac{1}{\beta} \ln \left(\mathbb{E}_{\mu_1}(\mathrm{e}^{-\beta(H_{\lambda_2} - H_{\lambda_1})}) \right).$$

The average $\mathbb{E}_{\mu_1}(\mathrm{e}^{-\beta(H_{\lambda_2} - H_{\lambda_1})})$ can then be approximated by sampling configurations from the canonical measure μ_1 and computing an empirical average of the observable $\mathrm{e}^{-\beta(H_{\lambda_2} - H_{\lambda_1})}$

over these samples. When the states λ_1 and λ_2 are too far apart, i.e. their respective distributions do not overlap, intermediate states can be considered and the difference $F(\lambda_2) - F(\lambda_1)$ can be computed as the sum of free energy differences between consecutive states. This simple solution is called "staging".

Histogram methods. Histogram methods can be used with a reaction coordinate. Here, the RC is assumed one-dimensional for simplicity. The method relies on the following observation: for each z, the observable

$$\chi_{z,\delta z}(q) = \frac{1}{\delta z}\chi\left(\frac{\xi(q) - z}{\delta z}\right)$$

satisfies:

$$\mathbb{E}_\mu(\chi_{z,\delta z}) \xrightarrow[\delta z \to 0]{} \mathrm{e}^{-\beta F(z)},$$

up to a multiplicative constant. Here χ must be a nonnegative function such that $\int_{\mathbb{R}} \chi = 1$, e.g. a Gaussian function or the indicator function of on interval such as $[-1, 1]$. This motivates computing values of $\mathbb{E}_\mu(\chi_{z_i,\delta z})$ for a discretization $z_1, \ldots, z_k$ of the domain of interest I of ξ. This can be done by sampling trajectories and computing ergodic (and discrete time) averages of the quantities $\chi_{z_i,\delta z}$. However, metastability can complicate the computation of these averages. Instead, histogram methods rely on sampling multiple shorter trajectories centered around each bin by adding a bias to the potential. The trajectory centered around bin i is then biased by a so called restrained potential $V_i(q) = V(q) + \frac{1}{2\epsilon_i}(\xi(q) - z_i)^2$. The parameter $\epsilon_i > 0$ should be small enough so the dynamics are effectively restrained around low values of $\xi(q) - z_i$, but large enough to still allow some exploration around the center z_i, making the canonical probability distributions associated with V_i overlap, so all values in I are sampled. The averages $\mathbb{E}_\mu(\chi_{z,\delta z})$ are then computed using the concatenation of the samples from all trajectories, with some reweighting procedure to take into account the restraining potential. Methods to perform this concatenation include the weighted histogram analysis method [55,56] (WHAM) and extended bridge sampling [57–59] (multistate Bennett acceptance ratio).

1.1.3.B Thermodynamic integration

Thermodynamic integration [60] relies on writing the free energy as the integral of the free energy derivative. In the alchemical setting, the free energy difference writes

$$F(\lambda_{\max}) - F(\lambda_{\min}) = \int_{\lambda_{\min}}^{\lambda_{\max}} F'(\lambda) d\lambda, \tag{1.23}$$

where $F'(\lambda)$ is defined in (1.17). The idea is then to discretize the $[\lambda_{\min}, \lambda_{\max}]$ interval into k bins centered around $\lambda_1, \ldots, \lambda_k$, and sample μ_{λ_i} for each $i \in 1, \ldots, k$. The averages $\mathbb{E}_{\mu_{\lambda_i}}\left(\frac{\partial H_{\lambda_i}}{\partial \lambda}\right)$ are then estimated. The free energy difference can be approximated from (1.23) using a quadrature method (e.g. a Riemann sum).

In the reaction coordinate case, the free energy is also computed as the integral of its derivative:

$$F(z_2) - F(z_1) = \int_0^1 \nabla F(\phi(\theta))\phi'(\theta) d\theta,$$

where ϕ is a smooth path in $\mathbb{R}^d$, and $\phi(0) = z_1$ and $\phi(1) = z_2$. In practice, when the reaction coordinate is one-dimensional, the free energy is computed using a simple quadrature method. When the RC is multi-dimensional, the Poisson equation is used instead:

$$\Delta F(z) = -\operatorname{div}(\nabla F). \tag{1.24}$$

Estimating the derivative $\nabla F(z)$ for $z = \phi(\theta)$ requires to sample conditional probabilities $\mu_\xi(dq|z)$. This can be done by projecting the usual dynamics (Langevin or overdamped Langevin) on the submanifold $\Sigma(z)$. We refer to Chapter 3 of 22 for more details.

1.1.3.C Non equilibrium methods

Nonequilibrium methods [61] compute the free energy difference between states as a nonlinear average over nonequilibrium trajectories started at equilibrium from the first state. In the alchemical case, the method starts from initial conditions $(q_0, p_0) \sim \mu_{\lambda_{\min}}$. A schedule for the evolution of the alchemical parameter is imposed by a function $\Lambda \in C^1([0,T],\mathbb{R})$ such that $\Lambda(0) = \lambda_{\min}$ and $\Lambda(T) = \lambda_{\max}$. The process (q_t, p_t) then follows a non-autonomous SDE. The free energy is estimated using the Jarzynski nonequilibrium equality [62] derived using a Feynman-Kac formula. This formula uses a definition of the work $\mathcal{W}_t$ induced on the system under the nonequilibrium dynamics and computes importance weights of nonequilibrium simulations with respect to the equilibrium distribution as $\mathrm{e}^{-\beta \mathcal{W}_t}$. The quantity $\mathrm{e}^{-\beta(F(\lambda_{\max})-F(\lambda_{\min}))}$ is then computed as an average over multiple realizations and multiple initial conditions of the exponential weights $\mathrm{e}^{-\beta \mathcal{W}_T}$. In practice, depending on the switching schedule Λ, the distribution of the computed weights can be spread out, and only very small values of $\mathcal{W}_t$ really count in the computed average. This can be solved by reducing the variance of the work distribution [63, 64].

In the reaction coordinate case, the method starts from initial conditions $(q_0, p_0) \sim \mu(\cdot|\xi(q) = z_{\min})$, and imposes a schedule $z \in C^1([0,T],\mathbb{R}^d)$ such that $z(0) = z_{\min}$ and $z(T) = z_{\max}$ by adding a constraining force to the dynamics to ensure $\xi(q_t) = z(t)$. A generalization of the Jarzynski equality is then used to compute the free energy difference as a nonlinear average over importance weights. It can be proven that thermodynamic integration methods are recovered from nonequilibrium methods in the limit $T \to \infty$, whereas free energy perturbation methods are recovered in the limit $T \to 0$. For more details on nonequilibrium free energy computations for both the alchemical and reaction coordinate cases, we refer the interested reader to Chapter 4 of [22].

1.1.3.D Adaptive methods

Adaptive biasing methods are importance sampling methods where the free energy is simultaneously estimated and used to bias the potential. More precisely, we mentioned in Section 1.1.2.D that, with the use of a reaction coordinate ξ which effectively describes the metastability of the dynamics, one can apply a modified potential $V - F \circ \xi$ to the dynamics (as shown in (1.22)) in order to eliminate the metastability along ξ and thus help accelerate the sampling of transitions between metastable states (see Figure 1.1). Unfortunately, the value of F is unknown and cannot be derived directly from its analytical form. Adaptive methods replace the free energy F by its estimation F_t at time t in the biased dynamics. The potential becomes $V - F_t \circ \xi$ where the estimate F_t is updated on-the-fly in a way that it gradually converges to F as the sampling proceeds. Two categories of adaptive biasing techniques can be distinguished, depending on whether the free energy itself, or its derivative, the mean force, is approximated. We give below a general definition and an example

equation for each category. For simplicity, we restrain ourselves to overdamped Langevin dynamics. The extension to the general case of phase space dynamics such as Langevin is straightforward.

Adaptive Biasing Potential. We first describe adaptive biasing potential (ABP) methods, where the free energy F_t is estimated and its gradient, the so-called mean force, is then derived and used in the dynamics. A general framework for ABP in the overdamped Langevin setting follows the equation:

$$\begin{cases} dq_t = -\nabla(V(q_t) - F_t \circ \xi(q_t))dt + \sqrt{\dfrac{2}{\beta}}dW_t, \\ \dfrac{dF_t(z)}{dt} = \mathcal{F}_t\left(-\dfrac{1}{\beta}\ln\left(\psi^\xi(t,z)\right)\right) , \end{cases}$$

where $\psi^\xi(t,z)$ is the image by ξ of the law $\psi(t,q)$ of the process q_t:

$$\psi^\xi(t,z) = \int_{\Sigma(z)} \psi(t,q)\delta_{\xi(q)-z}(dq),$$

and $\mathcal{F}_t : \mathbb{R} \to \mathbb{R}$ is a family of continuous and strictly increasing functions (e.g. $\mathcal{F}_t(x) = x$). The general idea behind ABP methods is to penalize the regions (i.e. values z) of the coordinate ξ that are visited by increasing their potential (through $F_t(z)$), thus favoring transitions to unexplored regions. Notable examples of ABP methods include the Wang-Landau algorithm [10, 65] (where the reaction coordinate is the potential energy) and metadynamics [11, 12].

Adaptive Biasing Force. Next, we discuss adaptive biasing force (ABF) methods [15–18] where the free energy derivative, the mean force, is estimated directly as a vector field Γ_t, and the free energy is subsequently obtained by numerical integration [66] of the mean force (using, e.g., a simple quadrature method for a one dimensional RC and the Poisson equation in (1.24) for a multi-dimensional RC). The equation of the dynamics of adaptive biasing force in the overdamped Langevin setting is:

$$\begin{cases} dq_t = \left(-\nabla V(q_t) + \displaystyle\sum_{j=1}^{d} \left[\Gamma_t(\xi(q_t))\right]_j \cdot \nabla\xi_j(q_t)\right) dt + \sqrt{\dfrac{2}{\beta}}dW_t, \\ \dfrac{d\Gamma_t(z)}{dt} = \mathcal{G}_t\left(\displaystyle\int_{\Sigma(z)} f(q)\nu^\xi(t, dq|z) - \Gamma_t(z)\right) , \end{cases} \tag{1.25}$$

for a family of vectors $\mathcal{G}_t(x) = \{\mathcal{G}_t^1(x_1), \ldots, \mathcal{G}_t^d(x_d)\}$ of continuous and strictly increasing functions such that $\mathcal{G}_t^i(0) = 0$. Here, $\nu^\xi(t, dq|z)$ is the distribution of q_t conditional to a fixed value of $\xi = z$:

$$\nu^\xi(t, dq|z) = \frac{\psi(t,q)\delta_{\xi(q)-z}(dq)}{\int_{\Sigma(z)} \psi(t,q)\delta_{\xi(q)-z}(dq)}, \tag{1.26}$$

and $f = (f_1, \ldots, f_d)$ is the local mean force defined in (1.21). Note that, apart from the case $d = 1$, the vector field Γ_t is not in general (i.e. for a general instantaneous distribution $\psi(t,q)$) a gradient. In the following section, more details on the adaptive biasing force methods are provided.

1.1.4 A focus on adaptive biasing force for free energy computations

The adaptive biasing force method and its variations are the methods of choice used in the work presented in this thesis. This section further details the ABF method as well as its extended version, the eABF method. For a more detailed presentation of ABF and eABF, including the mathematical foundation of the methods and proofs of convergence, we refer the interested reader to [15, 21, 22, 67].

1.1.4.A Adaptive biasing force

For simplicity and ease of notation, we assume in this section and the following section that the reaction coordinate is one-dimensional $\xi : \mathcal{D} \mapsto \mathcal{A} \in \mathbb{R}$, but the generalization to a higher dimensional CV is straightforward.

Dynamics and updating formula. As mentioned in the previous section, the adaptive biasing force method estimates the mean force associated with the collective variable ξ. For a one-dimensional function ξ, (1.20) becomes:

$$F'(z) = \frac{\displaystyle\int_{\Sigma(z)} f(q)\mathrm{e}^{-\beta V(q)}\delta_{\xi(q)-z}(dq)}{\displaystyle\int_{\Sigma(z)} \mathrm{e}^{-\beta V(q)}\delta_{\xi(q)-z}(dq)} = \mathbb{E}_\nu\left(f(q)|\xi(q)=z\right) \ , \tag{1.27}$$

where ν is the canonical measure defined in (1.4) and f is the local mean force introduced in (1.21), which, for a one-dimensional RC, simplifies as:

$$f = \nabla V \cdot \frac{\nabla\xi}{|\nabla\xi|^2} - \frac{1}{\beta}\,\mathrm{div}\left(\frac{\nabla\xi}{|\nabla\xi|^2}\right) \ . \tag{1.28}$$

Equation (1.27) shows that the derivative of the free energy $F'(z)$ is related to the conditional average of the local mean force for a process following the measure ν. However, conformations of the biased dynamics follow a different measure denoted by $\psi(t,\cdot)$ at time t. Note that by multiplying the numerator and denominator of (1.27) by the value $\mathrm{e}^{\beta F_t(z)}$, one obtains:

$$F'(z) = -\frac{1}{\beta}\int_{\Sigma(z)} f(q)\frac{\mathrm{e}^{-\beta(V(q)-F_t\circ\xi(q))}\delta_{\xi(q)-z}(dq)}{\int_{\Sigma(z)}\mathrm{e}^{-\beta(V(q)-F_t\circ\xi(q))}\delta_{\xi(q)-z}(dq)} = \mathbb{E}_{\nu_t}\left(f(q)|\xi(q)=z\right) \ ,$$

where $\nu_t \propto \mathrm{e}^{-\beta(V-F_t\circ\xi)}$. Indeed, $F_t\circ\xi(q) = F_t(z)$ for all $q \in \Sigma(z)$. Under the assumption of instantaneous equilibrium of the process at each time t with respect to $V - F_t \circ \xi$, i.e. $\psi(t,\cdot) = \nu_t$, this means that the mean force F' is directly recovered. Of course, the assumption of instantaneous equilibrium is false, but the above equation can still be used to approximate the mean force as

$$\Gamma_t(z) = \int_{\Sigma(z)} f(q)\nu^\xi(t, dq|z),$$

where $\nu^\xi(t, dq|z)$ is defined in (1.26). The dynamics, with the updating formula for Γ_t then read:

$$\begin{cases} dq_t = -\nabla V(q_t)dt + \Gamma_t(\xi(q_t))\nabla\xi(q_t)dt + \sqrt{\dfrac{2}{\beta}}dW_t, \\ \Gamma_t(z) = \displaystyle\int_{\Sigma(z)} f(q)\nu^\xi(t, dq|z) \ . \end{cases} \tag{1.29}$$

Remark 1. *Note that the above dynamics and updating formula correspond to the special case* $\mathcal{G}_t(x) = \frac{x}{\tau}$ *in the limit* $\tau \to 0$ *of the updating formula of equation* (1.25) *with a one-dimensional reaction coordinate.*

For completeness, we also write here the equation of ABF used in conjunction with Langevin dynamics:

$$\begin{cases} dq_t = M^{-1}p_t dt, \\ dp_t = -\nabla V(q_t)dt + \Gamma_t(\xi(q_t))\nabla\xi(q_t)dt - \gamma M^{-1}p_t dt + \sqrt{\dfrac{2\gamma}{\beta}}dW_t, \\ \Gamma_t(z) = \displaystyle\int_{\Sigma(z)} f(q)\nu^\xi(t, dq|z). \end{cases} \tag{1.30}$$

Under time continuous overdamped Langevin or Langevin dynamics with potential $V - F_t \circ \xi$, an empirical approximation $\Gamma_t^\varepsilon(z)$ at time t of the mean force can then be computed as:

$$\Gamma_t^\varepsilon(z) = \frac{\int_0^t f(q_s)\ \delta^\varepsilon\left(\xi(q_s) - z\right) ds}{\int_0^t \delta^\varepsilon\left(\xi(q_s) - z\right) ds} \ ,$$

where $\varepsilon > 0$ and δ^ε is a C^∞ approximation of the Dirac mass, $\delta^\varepsilon(z) = \frac{1}{\varepsilon}\chi\left(\frac{z}{\varepsilon}\right)$ with χ a smooth nonnegative function and $\int_{\mathcal{A}} \chi = 1$. In practice, the biasing dynamics are applied only for a domain of interest of the collective variable $z \in [z_{\min}, z_{\max}]$. This domain is discretized into K bins and the value of Γ_t at each bin center z_k for $k \in \{1, \dots, K\}$ is approximated as

$$\Gamma_t^k = \frac{\int_0^t f(q_s)\mathbb{1}_{\{z_k - \varepsilon \le \xi(q_s) < z_k + \varepsilon\}}(q_s)ds}{\int_0^t \mathbb{1}_{\{z_k - \varepsilon \le \xi(q_s) < z_k + \varepsilon\}}(q_s)ds},$$

where $\varepsilon = \frac{z_{\max} - z_{\min}}{2K}$.

Convergence of the biasing force. The convergence of Γ_t to F' can be proven under some assumptions on the potential V and the collective variable ξ. First we recall the definition of logarithmic Sobolev inequalities.

Definition 4. *A measure* π_2 *is said to satisfy a logarithmic Sobolev inequality with constant* r *if for all* π_1 *absolutely continuous with respect to* π_2*:*

$$H(\pi_1|\pi_2) \le \frac{1}{2r} I(\pi_1|\pi_2).$$

Here, $H(\pi_1|\pi_2) = \int_{\mathbb{R}} \ln\left(\frac{d\pi_1}{d\pi_2}\right) d\pi_1$ *is the entropy of* π_1 *with respect to* π_2 *and* $I(\pi_1|\pi_2) = \int_{\mathbb{R}} \left|\nabla \ln\left(\frac{d\pi_1}{d\pi_2}\right)\right|^2 d\pi_1$ *is the Fisher information of* π_1 *with respect to* π_2*.*

We now state a proposition on the convergence of ABF.

Proposition 1. *We make the following assumptions on ξ and V:*

$$\begin{cases} \xi \text{ is a smooth function such that } 0 < |\nabla \xi(q)| \leq m < \infty; \\ \sup_{q \in \mathcal{D}} |\nabla_{\Sigma(\xi(q))} f(q)| \leq M < \infty; \\ \exists\ \rho > 0 \text{ such that } \nu^{\xi}(\cdot|z) \text{ satisfies a logarithmic Sobolev inequality with constant } \rho \text{ for all } z \in \mathcal{A}; \\ \exists\ R > 0 \text{ such that } \psi_\infty \text{ satisfies a logarithmic Sobolev inequality with constant } R; \\ \text{The constants } m, M, \rho \text{ satisfy } \dfrac{mM\beta}{2\sqrt{\rho}} < 1. \end{cases}$$

When all the assumptions above are satisfied, under the biased Langevin (or overdamped Langevin) dynamics (1.30) *(or* (1.29)*), the biasing force Γ_t is proven to converge exponentially fast to the mean force F'.*

We refer the interested reader to [68] or Section 5.2 of [22] for the proof of Proposition 1, as well as quantitative estimates of the rate of convergence.

Proof of consistency. In general, under the assumption of convergence, a proof of consistency of the ABF method can easily be written. We briefly give this proof in the general case of ξ with values in $\mathcal{A} \subset \mathbb{R}^d$.

Theorem 1. *Under the assumption of convergence for ABF, i.e. when*

$$\psi(t,\cdot) \xrightarrow[t\to\infty]{} \psi_\infty, \qquad \partial_t \psi(t,\cdot)) \xrightarrow[t\to\infty]{} 0$$

and

$$\Gamma_t \xrightarrow[t\to\infty]{} \Gamma_\infty \quad \text{where } \Gamma_\infty \text{ is a gradient,}$$

then the limit Γ_∞ is the mean force ∇F.

Proof. Under these assumptions, we have $\psi_\infty = Z_\infty^{-1} \mathrm{e}^{-\beta(V - F_\infty \circ \xi)}$ such that $\Gamma_\infty = \nabla F_\infty$, and Z_∞ is a normalizing constant. In the limit $t \to \infty$, we then have:

$$\mathcal{G}_\infty \left(\frac{\int_{\Sigma(z)} f(q) \mathrm{e}^{-\beta(V(q) - F_\infty \circ \xi(q))} \delta_{\xi(q)-z}(dq)}{\int_{\Sigma(z)} \mathrm{e}^{-\beta(V(q) - F_\infty \circ \xi(q))} \delta_{\xi(q)-z}(dq)} - \Gamma_\infty(z) \right) = 0.$$

As $\mathcal{G}_t(0) = 0$ and $\mathcal{G}_t^i$ are continuous and strictly increasing, we thus have:

$$\Gamma_\infty(z) = \frac{\int_{\Sigma(z)} f(q) \mathrm{e}^{-\beta(V(q) - F_\infty(z))} \delta_{\xi(q)-z}(dq)}{\int_{\Sigma(z)} \mathrm{e}^{-\beta(V(q) - F_\infty(z))} \delta_{\xi(q)-z}(dq)} = \nabla F(z),$$

which allows to conclude $F_\infty = F$. □

1.1.4.B Extended system adaptive biasing force

Equation (1.28) shows that regular ABF requires the knowledge of second order derivatives of the CV ξ to compute the local mean force f. The analytical expression of this quantity is quite cumbersome for various choices of reaction coordinates, especially when ξ is vector

valued. In particular, as the CVs considered in this work are based on neural networks which may involve complex non-linear activation functions, extracting the second order derivatives is computationally too expensive.

To overcome the limitations in computing the second term of the right hand side of (1.28), a method coined extended system ABF (eABF) was devised [15, 67]. A fictitious degree of freedom $\lambda \in \mathbb{R}$ is added to the configurational space. The potential of the extended system becomes:

$$V^{\text{ext}}(q, \lambda) = V(q) + \frac{\kappa}{2}|\xi(q) - \lambda|^2 \ , \tag{1.31}$$

where $\kappa > 0$ is a force constant. In the case of Langevin dynamics, a fictitious mass m_λ for the degree of freedom λ must also be defined. To bias the dynamics of this extended system, the collective variable that is used is

$$\xi^{\text{ext}}(x, \lambda) = \lambda \ ,$$

instead of the original collective variable ξ. The new extended local mean force does not depend on the second (or any) derivatives of the collective variable ξ:

$$f^{\text{ext}}(q) = \frac{\partial V^{\text{ext}}}{\partial \lambda}(q) = \kappa(\lambda - \xi(q)).$$

Only the gradient of ξ is needed for computing the gradient of V^{ext}. ABF can therefore easily be applied to the new extended system. Denoting by ρ the momentum of λ, and by M^{ext} the extended mass matrix (which includes m_λ), the Langevin dynamics of the eABF trajectory are:

$$\begin{cases} dq_t^{\text{ext}} = (M^{\text{ext}})^{-1} p_t^{\text{ext}} \ dt, \\ dp_t^{\text{ext}} = \left(-\nabla V^{\text{ext}}(q_t, \lambda_t) + \Gamma_t^{\text{ext}}(\lambda_t)\mathbf{u}\right) dt - \gamma (M^{\text{ext}})^{-1} p_t^{\text{ext}} dt + \sqrt{\dfrac{2\gamma}{\beta}} dW_t, \end{cases} \tag{1.32}$$

where $q_t^{\text{ext}} = (q_t, \lambda_t)$, $p_t^{\text{ext}} = (p_t, \rho_t)$, $\mathbf{u}^T = (0, \ldots, 0, 1)$ and Γ_t^{ext} is the estimate at time t of the mean force associated with the RC λ in the extended system:

$$\Gamma_t^{\text{ext}}(\lambda) = \frac{\displaystyle\int_0^t f^{\text{ext}}(q_s) \ \delta^\varepsilon \left(\lambda_s - \lambda\right) ds}{\displaystyle\int_0^t \delta^\varepsilon \left(\lambda_s - \lambda\right) ds} \ .$$

Convergence of the extended free energy. The free energy F_κ^{ext} associated with ξ^{ext} in the extended system is related to the free energy associated with ξ through a convolultion

with a Gaussian kernel. Indeed, using (1.19),

$$\begin{aligned}
\mathrm{e}^{-\beta F_\kappa^{\mathrm{ext}}(\lambda)} &= \frac{\displaystyle\int_{\mathcal{D}} \mathrm{e}^{-\beta V^{\mathrm{ext}}(q,\lambda)} dq}{\displaystyle\int_{\mathcal{D}}\int_{\mathcal{A}} \mathrm{e}^{-\beta V^{\mathrm{ext}}(q,z)} dq dz} \\
&= \frac{\displaystyle\int_{\mathcal{D}} \mathrm{e}^{-\beta\left(V(q)+\frac{\kappa}{2}|\xi(q)-\lambda|^2\right)} dq}{\displaystyle\int_{\mathcal{D}} \mathrm{e}^{-\beta V(q)} \left(\int_{\mathcal{A}} \mathrm{e}^{-\beta\frac{\kappa}{2}|\xi(q)-z|^2} dz\right) dq} \\
&= \frac{1}{Z_\nu}\sqrt{\frac{\beta\kappa}{2\pi}} \int_{\mathcal{D}} \mathrm{e}^{-\beta V(q)} \mathrm{e}^{-\beta\frac{\kappa}{2}|\xi(q)-\lambda|^2} dq \\
&= \frac{1}{Z_\nu}\sqrt{\frac{\beta\kappa}{2\pi}} \int_{\mathcal{A}}\int_{\Sigma(z)} \mathrm{e}^{-\beta V(q)} \mathrm{e}^{-\beta\frac{\kappa}{2}|\xi(q)-\lambda|^2} \delta_{\xi(q)-z}(dq) dz \\
&= \int_{\mathcal{A}} \chi_\kappa(\lambda - z) \mathrm{e}^{-\beta F(z)} dz
\end{aligned}$$

where $\chi_\kappa(z) = \sqrt{\frac{\beta\kappa}{2\pi}}\mathrm{e}^{-\beta\frac{\kappa}{2}|z|^2}$ is a Gaussian kernel with variance $1/(\kappa\beta)$. Since $\chi_\kappa(z)dz \xrightarrow[\kappa\to\infty]{} \delta_0(dz)$ in the sense of distributions on $\mathcal{A}$, it can be easily proven from the last equality that the free energy F_κ^{ext} converges to F when $\kappa \to \infty$. In practice, a compromise must be made in choosing the value of the spring force constant κ which should be large enough to ensure that F_κ^{ext} is close to F, but small enough so that the dynamics can still be discretized in practice with a timestep Δt which is not too small.

1.1.5 Collective variable optimality

As mentioned in Section 1.1.2.D, the choice of the reaction coordinate ξ greatly impacts the sampling efficiency as well as the relevance of the computed free energy. In this section, we discuss different physical or mathematical criteria for the choice of a reaction coordinate. In general, and as mentioned above, ξ is a low dimensional function of the positions q. We denote by d its dimension.

1.1.5.A Physical and chemical requirement: Interpretability

From a physical and biochemical standpoint, the RC should be interpretable. More precisely, the RC should have a somewhat direct biophysical meaning linked to the structure and physics of the system. We recall that one of the goals behind the definition of a RC (namely within the scope of this work) is the computation of a free energy landscape. This free energy landscape describes the system dynamically, i.e. determines the metastable states (minima), and the free energy barriers to overcome in order to achieve transitions. Using a physically interpretable RC makes it more straightforward to understand the obtained free energy, or even to compare it to an experimentally measurable quantity. Simple examples of an interpretable RC include interresidue distances, dihedral angles, root mean square deviation from a reference conformation, or the radius of gyration.

1.1.5.B Static requirement: Efficient sampling of the canonical measure

Another requirement on the RC comes from a static or thermodynamic perspective on the system, i.e. when only the distributions of the system and of the RC at equilibrium are examined. In general, an intuitive way to describe the RC is that ξ should be constructed such that for a set of points $(q^i)_{1\leq i\leq n}$ drawn from ν, ν is well sampled by $(q^i)_{1\leq i\leq n}$ if ν^ξ is well sampled by $\xi(q^i)$. This sampling requirement can be stated from a biased sampling point of view. Indeed, a natural optimality criterion for an RC is the efficiency of using it in a free energy biased sampling procedure such as ABF or metadynamics. As mentioned in Section 1.1.2.D, biasing along the RC ξ works best if it is metastable, i.e. if for a dynamics q_t sampling ν, the metastability of q_t is encoded in the metastability of $\xi(q_t)$. We formally defined this metastability as the RC having to accurately describe the slow motions of interest. We explore in this section this statement from a mathematical and from a practical point of view.

We use a mathematical definition of metastability in relation to the distributions ν^ξ and $\nu(dq|\xi(q)=z)$ as done in [69].

Definition 5. *Assuming that the conditional means $\nu(dq|\xi(q)=z)$ (respectively the marginal ν^ξ) follow a logarithmic Sobolev inequality with coefficient ρ (respectively r), then*

$$\xi \text{ is metastable if } \quad r \ll \rho. \tag{1.33}$$

This criterion is roughly included in the assumptions listed in Section 1.1.4.A and which are required for the convergence of ABF: the logarithmic Sobolev inequality satisfied by $\nu(dq|\xi(q)=z)$ can be interpreted as a property ensuring that when sampling the law $\nu^\xi(\cdot|z)$, e.g. through constrained dynamics [70], convergence to equilibrium is exponential with rate ρ (as made precise in Chapter 3 of [22]).

From a practical point of view, in order for the biasing along ξ to be efficient, the free energy landscape of the RC should simply contain and adequately describe all of the energy barriers and basins corresponding to these slow motions. Consequently, the RC itself should distinguish between the different states of the system, metastable and transient. Indeed, if ξ resolves the metastable states but confuses a metastable state with a transient one, full sampling could still be impeded in the biasing simulation [71]. If all states are well distinguished by ξ, the conditional distribution $\nu(dq|\xi(q)=z)$ is not supported inside more than one state (metastable or transient), which again means that $\nu(dq|\xi(q)=z)$ is essentially unimodal and thus easy to sample.

Finally, the static requirement can be formally linked to an assumption on which many dimensionality reduction methods rely: That the support of the probability measure of the data (here ν) actually stays approximately within a lower dimensional manifold (here of dimension d), making it possible to represent the high dimensional data (here the positions q) using a lower dimensional set of functions (i.e. the RC ξ). If this assumption is true, then for a well chosen low dimensional embedding ξ, it can be assumed that fully sampling the values of ξ should provide an efficient sampling of the original distribution ν, again implying that the conditional probabilities $\nu(dq|\xi(q)=z)$ are easy to sample.

1.1.5.C Dynamical requirement: Markovianity and effective dynamics

The dynamical relevance of RCs is linked to the coarse grained dynamics of the collective variable [72]. The RC is considered optimal if the full dynamics can be projected on it without losing too much information on the important processes. More precisely, the RC

is Markovian and it can accurately determine dynamical quantities of interest, such as transition mechanisms and transition rates between states, or time scales of the dynamics. We consider here dynamics on positions only, e.g. overdamped Langevin. We also assume $\mathcal{D} = \mathbb{R}^{3N}$, and $\mathcal{A} = \mathbb{R}^d$. We denote by $\mathcal{L}$ the infinitesimal generator of the considered dynamics.

Markovianity. For a given RC ξ, one can define the so called effective dynamics

$$dz_t = b(z_t)dt + \sqrt{\frac{2}{\beta}}\sigma(z_t)dB_t,$$

where B_t is a d-dimensional Brownian process, and the functions b and σ can be equivalently defined using different approaches such as Itô's formula, the Mori-Zwanzig method and the Galerkin method. We refer to [72, 73] for mathematical descriptions of these methods as well as general formulas for the functions b and σ. Here, we provide the formula for b and σ in the case of overdamped Langevin dynamics and a real valued RC:

$$\begin{cases} b(z) & = \int_{\Sigma(z)} \left(-\nabla V \cdot \nabla \xi + \beta^{-1}\Delta\xi\right)(q)\nu^\xi(dq|z) \\ \sigma^2(z) & = \int_{\Sigma(z)} \|\nabla\xi(q)\|^2 \nu^\xi(dq|z). \end{cases}$$

We denote by $\tilde{\mathcal{L}}$ the generator of the effective dynamics. The invariant measure corresponding to these dynamics is ν^ξ. The effective dynamics represent an approximation of a closed form dynamics for the RC. The Markovianity criterion states that the RC is good if the law of $\xi(q_t)$ and that of the effective dynamics variable z_t are close. The Markovianity criterion is closely linked to the static requirement introduced in the previous section. Formally, if ξ is metastable, the evolution of the directions orthogonal to ξ, which represent fast motions, can be averaged out for intermediate time scales, i.e. time scales longer than the relaxation times of these fast directions but shorter than the relaxation times of the slow processes described by ξ. A more mathematically rigorous link lies in that the same assumptions on ξ stated for the sampling requirement (or for the convergence of ABF) can be used to establish an error bound between the laws of $\xi(q_t)$ and z_t. More precisely, if $\nu(dq|\xi(q) = z)$ satisfies a logarithmic Sobolev inequality with coefficient ρ uniformly in z, then the relative entropy of the law of $\xi(q_t)$ with respect to the law of z_t is bound by a constant proportional to $\frac{1}{\rho^2}$. We refer to [69] and references therein for further details and proofs.

Properties of the effective dynamics and preservation of time scales. In [72], it is proven that under some natural assumptions on the system and the RC, many structural properties of the full dynamics q_t, namely ergodicity and reversibility, are preserved by the effective dynamics. The optimality of ξ is then measured in terms of the preservation of dynamical properties of the dynamics. The dynamical properties to use for assessing the optimality of the RC can differ according to the system under study and the process or processes of interest. Examples of these dynamical quantities include inherent time scales, reaction or transition rates, as well as transition probabilities. Here, we focus on the preservation of the time scales linked to the slow processes of the dynamics, also called relaxation rates, as an optimality criterion. The inherent time scales are related to the eigenvalues of the infinitesimal generator $\mathcal{L}$. We denote by the pairs $\{(\lambda_i, \phi_i)\}_{i\in\mathbb{N}}$ (resp. $\{(\tilde{\lambda}_i, \tilde{\phi}_i)\}_{i\in\mathbb{N}}$) the eigenvalues and normalized eigenfunctions of the generator $\mathcal{L}$ (resp. $\tilde{\mathcal{L}}$), ordered so that $0 = \lambda_0 > \lambda_1 \geq \lambda_2 \geq ...$ (resp. $0 = \tilde{\lambda}_0 > \tilde{\lambda}_1 \geq \tilde{\lambda}_2 \geq ...$). We recall here

that $\phi_0 = 1$ and that for $i > 0$, the ith implied timescale, i.e. the relaxation rate of the ith slowest process of the system dynamics is $t_i = \frac{1}{\lambda_i}$. A slow process of the system indicates for example a transition between two metastable states. We assume here that the system possesses a number d of slow processes of interest. Consequently, a spectral gap separates the first $d+1$ eigenvalues from the rest: $\lambda_d \gg \lambda_{d+1}$. This means that we are only interested in the preservation of the first $d+1$ timescales. Note that we look here at the eigenvalues of the generators, but it is also possible (and equivalent) to examine the eigenvalues of the evolution semi-group $\mathrm{e}^{\tau\mathcal{L}}$ for some $\tau > 0$. For a general RC ξ, the relationship between the pairs (λ_i, ϕ_i) and $(\tilde{\lambda}_i, \tilde{\phi}_i)$ is given by the following theorem quoted from [72].

Theorem 2. *For all $i \in \mathbb{N}$ (and thus in particular for all $i < d+1$), we have:*

$$\lambda_i \leq \tilde{\lambda}_i \leq \lambda_i + \frac{1}{\beta}\|\nabla(\phi_i - \tilde{\phi}_i \circ \xi)\|^2_{L^2(\nu)} \ . \tag{1.34}$$

Theorem 2 indicates first that the eigenvalues are always overestimated by the effective dynamics, and second that they are well approximated by the effective dynamics if the eigenfunctions of the generator associated with the projected dynamics approximate the original eigenfunctions well:

$$\forall i < d+1 \ , \quad \phi_i \approx \tilde{\phi}_i \circ \xi. \tag{1.35}$$

In [74,75], this criterion is proven to be equivalent to a property on the transition probability $p^\tau(x, \cdot) = p(\cdot, \tau | x, 0)$ of the dynamics, i.e. the probability distribution of x_τ conditionally to $x_0 = x$. More precisely, ξ is considered an optimal reaction coordinates (from a timescales preservation standpoint) if the transition probability can be approximated by a function of $\xi(x)$ alone:

$$p^\tau(x, \cdot) \approx q^\tau(\xi(x), \cdot). \tag{1.36}$$

This criterion is no longer linked to the eigenvalues or eigenfunctions themselves. In the same work, a numerical computation of an RC which satisfies this requirement is introduced using the so-called transition manifold $\mathbb{M} = \{p^\tau(q, .) | q \in \mathcal{D}\}$.

1.1.5.D Natural candidates

Energy. The potential energy itself as a collective variable

$$\xi(x) = V(x)$$

is an intuitive choice for fulfilling both the chemical and static requirements: The energy is of course directly interpretable from a biophysical standpoint, and because the metastable and transient states are by definition determined by the value of the potential energy, it provides an ideal candidate for biased sampling. This choice of RC is in fact at the heart of the Wang-Landau adaptive biasing potential algorithm [10, 65]. It also has the advantage of being one dimensional, which is convenient for biased sampling. Note however that an important practical problem arises with the use of the potential energy as a biasing CV, namely the biasing range: It is not easy to set the maximal value of the potential energy in order to sample (only) relevant configurations of the phase space. Setting it too high pushes the system to visit unrealistic high energy regions, while setting it too low keeps the system trapped in the starting state.

Eigenfunctions of the generator. A natural candidate for the dynamical requirement of preservation of the ($d+1$ dominant) eigenvalues of the generator $\mathcal{L}$ is to use the ($d+1$ dominant) eigenfunctions as the RC:

$$\xi(x) = (\phi_1(x) \dots, \phi_d(x))^T.$$

Indeed, this choice is proven to insure that the inequalities in Theorem 2 become equalities for all $1 \leq i \leq d$ [72]. As the eigenfunctions of the generator (or transfer operator) of the system describe the slow sub-processes, the resulting RC, while not containing a structural description of the system, still contains some dynamical interpretation. In fact, it has been observed that the eigenfunctions have (approximately) constant values on metastable states [76]. This choice of RC does however present some practical drawbacks. First, the infinitesimal generator of the dynamics is a (often) high dimensional differential operator, its eigenvalue problem (or that of the transfer operator) is computationally expensive or even impossible to solve numerically. Second, depending on the system under study, some or all of the d sub-processes of interest (i.e. transitions between metastable states) can be described by the same reaction coordinate if they lie within the same transition path, making the corresponding eigenfunctions redundant. In this case, a lower dimensional RC can be sufficient for describing all of the subprocesses, and its optimal dimension is the dimension of the transition manifold [75, 77]. The eigenfunctions can then still be used to assess the optimality of the lower dimensional RC ξ using, e.g. Theorem 2.

Committor function. When considering a transition process between two metastable states A and B, the committor function $p_A(q)$ is the probability for the system to visit state A before state B starting from the position q. Of course, we have $p_A(q) = 1$ for all $q \in A$ and $p_A(q) = 0$ for all $q \in B$. An extension of this definition to a d-state system is a $d-1$-dimensional function $p = (p_{A_1}, \dots, p_{A_{d-1}})$ where $p_{A_i}(q)$ is the probability for the system to reach state A_i before any other state. The last probability p_{A_d} can be computed as $p_{A_d} = 1 - (p_{A_1} + \dots + p_{A_{d-1}})$, it is therefore unnecessary to include it. The choice

$$\xi(q) = (p_{A_1}(q), \dots, p_{A_{d-1}}(q))$$

is often considered the optimal reaction coordinate, and satisfies all three requirements listed above:

- While it does not provide a straightforward physical or chemical insight on the structure of the system, the committor function still is an interpretable quantity as it is defined by the metastable states.
- By definition, the committor describes and distinguishes the metastable and transient states: Its value is constant on the metastable states, and accurately traces transition paths on transient states. This makes it a good candidate for biased sampling.
- The committor is in proven [74] to satisfy (1.36) thus preserving the time scales of the full dynamics. Additionally, in [72], the authors also prove that the committor function between states A and B is the reaction coordinate which perfectly preserves the reaction rate between these states.

Because the committor is considered to be an optimal reaction coordinate in many cases [78–80], it can be used to assess the quality of a collective variable ξ using e.g. the committor

histogram test [81]: for each value z of the reaction coordinate (or rather for a discretization $z_1 \dots z_k$ of the coordinate), the committor is computed for multiple positions $q \in \Sigma(z)$. The reaction coordinate is then considered optimal for the description of the reaction process if approximately similar values of the committor are obtained for all $q \in \Sigma(z)$, indicating that the RC and the committor have approximately the same isosurfaces. Unfortunately, this choice of RC shares the same practical drawback as the eigenfunctions: the committor is computationally expensive to compute for all positions q, and its dimensionality is equal to the number of states, which can be higher than the actual number of coordinates needed to describe the dynamics.

1.2 Machine Learning and data driven collective variable discovery

We briefly discussed at the end of Section 1.1.2 the importance of the choice of the reaction coordinate ξ, which greatly impacts the efficiency of the sampling and/or free energy computation methods. Indeed, with a poor choice of CV, the free energy cannot: **(i)** provide an efficient biasing of the dynamics, nor **(ii)** be used for analyzing motions of interest between different states. It is thus important to use a reaction coordinate that allows to describe the metastability of the system. In general, this choice can be made somewhat intuitively for small and/or extensively studied systems. However, the more complex and/or larger the system, the less trivial it is to manually select one or more collective variables. The idea of automatically selecting or constructing the collective variable becomes an attractive solution. For this purpose, many methods have been devised to construct CVs using sampled configurations of a given system. In particular, as the recent years have witnessed a surge in popularity for machine learning (ML) techniques in various fields, the discovery of collective variables using machine learning has gained growing interest. In this section, we provide concise presentations of some of the state of the art methods for data driven collective variable discovery. For a more complete overview on optimization and learning methods for CV discovery and in molecular dynamics in general, we refer the reader to [20, 82, 83]. First, we present in Section 1.2.1 some necessary preprocessing steps for transforming the sampled trajectories into suitable training data. In particular, the choice between Cartesian and internal coordinates is discussed. We then proceed to our overview of some CV learning methods by categorizing them according to the type of data they use. More precisely, we distinguish three categories of CV discovery.

- Section 1.2.2 presents the first category of methods, which use only sampled conformations, with no other knowledge but the sampling distribution. It includes classic unsupervised learning algorithms.

- Section 1.2.3 then discusses the second class of methods. Here, the algorithms rely on time series data, meaning that the data samples are obtained from one or more MD trajectories. Many algorithms of this category aim at approximating the transfer operator of the dynamics.

- Finally, the third category of methods, presented in Section 1.2.4, uses discrete state information to apply supervised learning models. This means that a separation of the space into a small number of states is known and used to classify the data samples.

1.2.1 Choice of input features

Before applying a model to construct a collective variable from sampled configurations, it is first necessary to pre-process the dataset of these sampled configurations in order to select the type of input features that will be fed to the model. Often, only the representation of the molecule, without the surrounding solvent, is considered relevant to the search of a CV. The first step is thus to remove the solvent molecules. Moreover, some atoms are often dismissed as they represent irrelevant fast motions that would only add noise to the data. This famously includes hydrogen atoms, but can also be extended to more atoms depending on the system under study. The description of the remaining atoms may be achieved by a variety of features. Here, we briefly discuss three general choices: Cartesian coordinates, internal coordinates and candidate collective variables.

Cartesian coordinates. Cartesian coordinates are the straightforward representation of the data as they are the typical output of an MD simulation. This choice of coordinates requires eliminating the overall rotational and translational motion of the molecule. Translational motion is simply eliminated by recentering each sampled conformation to the origin of the Cartesian $3D$ space. Rotational motions can be eliminated by aligning the sampled conformations to a pre-selected reference structure q_{ref}. For each sample, a rotation matrix is computed to minimize the root-mean square deviation between the reference structure and the rotated conformation. A variety of algorithms were devised for the efficient and accurate computation of the rotation matrix, including notably the Kabsch algorithm [84], quaternion-based methods [85–87], and other methods [88, 89]. It is important to note that these algorithms for removing the overall rotational motions are less efficient when applied to flexible systems, such as folding proteins [90, 91]. This is due to the fact that internal and overall motions for these systems are mixed in the Cartesian coordinates, and that the alignment is dependent on the selection of the reference structure. New methods for improving the separation between external and internal motions were devised [92–94], but the most efficient method for fully eliminating rotational invariances is by using internal instead of Cartesian coordinates.

Internal coordinates. Internal coordinates are, by definition, invariant with respect to rotational as well as translational motions of the system. Several types of internal coordinates can be used, such as interatomic distances, angles, or dihedral angles. Two problems arise with the use of interatomic distances. First, the data dimensionality increases quadratically with respect to the number of atoms considered. Second, the obtained features are usually highly correlated or even redundant. The same issues appear with angular coordinates. This can be avoided by considering only a subset of atom pairs (or 3-tuples) which are selected using a condition of some sort. In [95] for example, the authors use only distances between inter-residue contacts in the native state of the studied protein (and define a threshold for what is considered a contact). Dihedral angles are defined by sets of 4 consecutively bonded atoms, therefore, using them as internal coordinates would not (or not considerably) increase the data dimensionality. They represent a direct insight into the intrinsic torsions within each residue of the molecule. When using dihedral angles (or bond angles), their periodic nature must be taken into consideration. Indeed, most machine learning or dimensionality reduction methods are not tailored to handle periodic data. Considering for example the two angles $-140°$ and $120°$, their arithmetic average ($-10°$) and absolute difference ($260°$) do not correspond respectively to the circular mean $170°$ and the actual

distance 100°. These issues can be circumvented by defining a suitable metrics [96, 97], or by using the sine and cosine transforms of the angles [98, 99].

Candidate collective variables. When some preliminary information or expert knowledge of the system is available beforehand, a more efficient representation can be obtained by selecting a set of candidate physical or chemical CVs, which typically do not necessarily describe the whole structure, but rather a specific part of the system. These candidate CVs can include specific internal or Cartesian coordinates describing a small part of the system that is identified as of interest. Other physical properties can also be used, for example the root-mean square deviation with respect to a structure of interest (e.g. to the folded state for a protein), the radius of gyration, the solvation state, etc. Some methods for collective variables discovery rely on the availability of such inputs [78, 100, 101]. These methods have the advantage of being much more interpretable as they output (combinations of) known physical or chemical CVs.

For any choice of input representation and any other preprocessing procedures, we denote by $X \in \mathbb{R}^{n\times D}$ the obtained processed dataset of n elements $x^i \in \mathbb{R}^D$, and in general by x a point in the space $\mathcal{X} \subseteq \mathbb{R}^D$ of the processed conformations. We now seek to construct a collective variable $\xi : \mathcal{X} \to \mathcal{A} \subseteq \mathbb{R}^d$ with $d < D$ which represents the data X in a low dimensional space.

1.2.2 Using input features only

In this section, we present methods of CV discovery which only rely on the input dataset X. We divide these methods into three subcategories: linear algorithms presented in Section 1.2.2.A, nonlinear algorithms which are not based on neural networks or deep learning methods in Section 1.2.2.B, and neural network based methods in Section 1.2.2.C. We note that some of the methods below do not always output a clear mapping/function ξ from the inputs x to the CV, some methods instead only providing a matrix $Z \in \mathbb{R}^{n\times d}$, which CV values z^i corresponding to each data sample x^i in the original dataset X. This constitues an issue as we often want to use the computed CV to apply a free energy biasing method, most of which, including ABF and eABF, require a globally defined differentiable CV function ξ. For each of the methods presented, we clarify whether such a mapping is provided (rather than the matrix Z alone). Some methods devised to generate the mapping ξ using the data X and Z are briefly discussed in Section 1.2.2.D.

1.2.2.A Linear dimensionality reduction

Principal Component Analysis. Principal component analysis is perhaps the most commonly used linear dimensionality reduction method for the construction of collective variables of biomolecular processes [98, 102, 103]. The method seeks to construct orthogonal directions which maximize the variance of the dataset X. The first direction, or principal component, is the direction of maximum variance. The ith principal component maximizes the variance among all directions orthogonal to the components 1 through $i-1$. The components are constructed by diagonalizing the positive symmetric covariance matrix

$$C = \frac{1}{n}\sum_{i=1}^{n}(x^i - \bar{x})^T(x^i - \bar{x}), \quad \text{where} \qquad \bar{x} = \frac{1}{n}\sum_{i=1}^{n} x^i.$$

The eigenvectors $v_1, \dots, v_D$ of C associated with decreasing eigenvalues $\lambda_1 \geq \cdots \geq \lambda_D$ are such that each v_k, for $1 < k \leq D$, is the direction of maximal variance over the subspace orthogonal to $\text{span}(v_1, \dots, v_{k-1})$ (and v_1 is the direction of maximal variance). The variance explained by each eigenvector $v_i \in \mathbb{R}^D$ is determined by its associated eigenvalue λ_i, so that the choice of d (which should be as small as possible while covering as much of the data variance as possible) can be made by searching for a spectral gap in the eigenvalues $\lambda_d \gg \lambda_{d+1}$. The CV ξ is then defined as the projection over the first d eigenvectors:

$$\xi(x) = (x \cdot v_1, \dots, x \cdot v_d).$$

Multidimensional scaling. Multidimensional scaling (MDS) is a class of dimensionality reduction methods which construct a low dimensional representation Z of the dataset X with the aim of preserving pairwise distances between data samples $d(x^j, x^k)$. The representation $Z \in \mathbb{R}^{n \times d}$ minimizes a loss of the form:

$$c(X, Z) = \sum_{j,k=1}^{n} \left(d(x^j, x^k) - d(z^j, z^k) \right)^2 .$$

In general, the solution to this optimization problem is obtained using an iterative algorithm. However, another version of the algorithm, called classical MDS, obtains a closed form solution by changing the loss function. Here we consider the case where $d(\cdot, \cdot)$ is the Euclidean distance. In classical MDS, the optimization problem is replaced by:

$$\underset{Z \in \mathbb{R}^{n \times d}}{\text{argmin}}\, c(X, Z), \qquad c(X, Z) = \|J^T (D_X^2 - D_Z^2) J\|_2^2 \tag{1.37}$$

where $D_X^2(j,k) = \|x^j - x^k\|^2$, $D_Z^2(j,k) = \|z^j - z^k\|^2$ and $J = I_n - \frac{1}{n}\mathbf{1}_n$ is the so-called centering matrix (here $\mathbf{1}_n$ is a $n \times n$ matrix of ones). It is easily proven that a solution to this new minimization problem is obtained by computing the eigenvalue decomposition of the centered matrix K of pairwise inner products: $K_{j,k} = (x^j - \bar{x})^T (x^k - \bar{x}) = -\frac{1}{2} J^T D_X^2 J$. Given the eigenvectors $v_i \in \mathbb{R}^n$ and corresponding eigenvalues λ_i of the matrix K, the embedding $Z = (z_1^T, \dots, z_d^T)$ which minimizes (1.37) satisfies: $z_i = \sqrt{\lambda_i} v_i$. Importantly, computing K and thus Z requires knowledge of the distance matrix D_X^2 alone (without having access to the actual dataset X). The collective variable is then only given at the data samples x^i:

$$\forall i \in \{1, \dots, n\}, \qquad \xi(x^i) = z^i,$$

where z^i is the ith line of Z.

Remark 2. *The MDS algorithm is actually only linear if the distance d is the Euclidean distance. In this case, the representation Z recovered by classical MDS is in fact the same as the PCA projection, i.e $Z_{MDS} = Z_{PCA}$. The main difference is in that PCA is based on the $D \times D$ matrix C while MDS uses the $n \times n$ matrix K. MDS also does not need the actual dataset X, but only the distance matrix D_X. However, MDS does not compute projection vectors, but rather directly computes the projected matrix of the dataset X.*

1.2.2.B Nonlinear methods based on kernels or distances

In this section, we present methods for collective variables discovery and dimensionality reduction for which nonlinearity is introduced by the use of distance functions other than the Euclidean distance between data samples, and/or nonlinear kernel functions.

The kernel trick. The kernel trick [104] is a method which allows to introduce nonlinearity into linear machine learning and dimensionality reduction techniques such as PCA. The kernel trick represents the data X in a new higher dimensional space using a nonlinear transformation ϕ. PCA, or more precisely MDS, can then be applied in this new space. The idea is that, with a good choice of the nonlinear transformation ϕ, projecting the data into the higher dimensional space will make it approximately linear. The "trick" consists in the fact that, in practice, the transformation ϕ is never computed explicitely, but is represented by a kernel function h, such that the scalar product between pairs of transformed samples can be computed as:

$$\phi(x^j)^T \phi(x^k) = h(x^j, x^k).$$

The matrix $K = (h(x^j, x^k))_{1 \leq j,k \leq n}$ of pairwise inner products in the high dimensional space is then used to apply MDS. As mentioned in the previous section, the obtained directions are the principal components of PCA. Importantly, because the transformation ϕ is never directly used, its dimensionality can be high, or even infinite. The dimension is solely defined by the choice of the kernel h, which highly impacts the efficiency of the kernel method. Of course choosing a linear kernel $h(x^j, x^k) = (x^j)^T x^k$ enables recovering regular (linear) PCA. Other kernels include polynomial kernels or the popular Gaussian kernel $h(x^j, x^k) = \frac{1}{\sqrt{2\pi}\sigma} \exp(-\frac{\|x^j - x^k\|^2}{2\sigma^2})$, where σ, the standard deviation of the kernel, is a parameter to be adjusted.

Isomap. Isomap [105, 106] is a nonlinear version of MDS, where the distance $d(\cdot, \cdot)$ is the geodesic distance between pair of samples and is computed using a graph representation of the data X. More precisely, using a k-nearest neighbor approach or a threshold value for assigning neighbors, Isomap constructs a graph of the data where vertices are the data samples x^i. If two vertices are considered as neighbors, they are connected by edges whose lengths are the Euclidean distances between the sample pair. The distance $d(x^j, x^k)$ is then computed as the shortest path between corresponding vertices in the constructed graph, using e.g. Djikstra's algorithm. Finally, MDS is applied using this definition of pairwise distance, to obtain a representation Z of the original dataset.

Diffusion Maps. The following definition of the diffusion maps method is based on [107]. We refer to this paper for more details and proofs of the method. Diffusion maps [107–109] compute approximations for the eigenfunctions ϕ_i and eigenvalues λ_i, $1 \leq i \leq d$ of the infinitesimal generator $\mathcal{L}_{\text{od}}$ of the overdamped Langevin dynamics defined on $L^2(\nu)$ (see (1.8)). As discussed in Section 1.1.5.D, the dominant eigenfunctions of the generator represent good collective variables, as they satisfy most RC optimality requirements listed in Section 1.1.5.

The d-dimensional diffusion map is defined as the function:

$$\Psi_{d,t}(x) = (\mathrm{e}^{-\lambda_1 t}\phi_1(x), \ldots, \mathrm{e}^{-\lambda_d t}\phi_d(x)).$$

The approximations of the eigenvalues and eigenfunctions are computed using simulated data. More precisely, given the dataset $X = x^1, \ldots x^n$, the diffusion map is approximated with the use of a kernel function $h_\epsilon(x, x') = \exp\left(-|x - x'|^2/2\epsilon^2\right)$. Here ϵ is a predefined parameter to measure the local neighborhood of each data point. The ϵ-density of x^k is:

$$p_\epsilon(x^k) = \sum_{j=1}^{n} h_\epsilon(x^k, x^j) \ .$$

The pairwise kernel matrix is then computed as

$$K = \left(\frac{h_\epsilon(x^k, x^j)}{\sqrt{p_\epsilon(x^k) p_\epsilon(x^j)}} \right)_{1 \leq j,k \leq n},$$

and rescaled by a row normalization using the diagonal matrix with entries $D_k = \sum_{j=1}^{n} K_{k,j}$, to form the Markov matrix $M = D^{-1}K$. It is proven that, if the data points $x^1 \dots x^n$ are sampled from the probability density ν of the overdamped Langevin dynamics, then when $n \to \infty$ and $\epsilon \to 0$, the operator $(M - I)/\epsilon$ converges in probability to the generator $\mathcal{L}_{\text{od}}$. The eigenvectors v_i and eigenvalues ζ_i of $(M - I)/\epsilon$ are thus computed as an approximation of the values of the eigenfunctions ϕ_i over the dataset X, and their corresponding eigenvalues λ_i. The d-dimensional RC is then:

$$Z = (v_1, \dots, v_d).$$

Stochastic neighbour embedding. Stochastic neighbour embedding [110] (SNE) also uses pairwise distances between data samples. Here, the pairwise distances are used to compute a neighborhood probability function p over pairs of samples, where the probability for a sample x^k to be in the neighborhood of a sample x^j is computed as

$$p(x^k|x^j) = \frac{\mathrm{e}^{-\|x^j - x^k\|^2/2\sigma_j^2}}{\sum_{l \neq k} \mathrm{e}^{-\|x^j - x^l\|^2/2\sigma_j^2}}.$$

This probability function is non-symmetric in its arguments x^j, x^k, thus the actual neighborhood probability between x^j and x^k is given by

$$P_{jk} = \frac{p(x^k|x^j) + p(x^j|x^k)}{2n}.$$

Given the desired low dimension d, a second probability density Q over the d-dimensional representation $Z \in \mathbb{R}^{n \times d}$ is then constructed as a Gaussian distribution in the original SNE; or in the more popular t-SNE [111–113], as a Student's distribution with parameter $k = 1$:

$$Q_{jk} = \frac{(1 + \|z^j - z^k\|^2)^{-1}}{\sum_{l \neq k} (1 + \|z^j - z^l\|^2)^{-1}}.$$

The representation Z is then computed so as to minimize a distance between the distributions P and Q as measured, e.g. by the Kullback-Leibler divergence. The minimization problem reads:

$$\underset{Z \in \mathbb{R}^{n \times d}}{\operatorname{argmin}} KL(P||Q), \qquad KL(P||Q) = \sum_{j=1}^{n} \sum_{k=1, k \neq j}^{n} P_{jk} \log\left(\frac{P_{jk}}{Q_{jk}}\right).$$

The CV is then obtained for each data sample x^j as the low dimensional representation z^j.

1.2.2.C Nonlinear deep learning methods

Artificial neural networks (ANN) [114] are a subset of machine learning methods. Their design is roughly based on the workings of the human brain, in that artificial neurons are made to send signals to one another. More precisely, the neural network is composed of several layers, each of which contains a number of neurons. The input layer contains as many neurons as the dimensionality of the data we wish to apply the neural network to, and the output layer neurons are meant to contain the information we wish to learn using the network. Intermediate layers are used to increase the complexity of the network. Each neuron in each layer is assigned a weight vector which connects it to the neurons of the next layer. The value in each neuron is then computed as a weighted combination of the values of the previous layer neurons (using each neuron's assigned weight), passed through a (often nonlinear) differentiable transformation called an activation function. The neural network is trained by modifying the weights assigned to each neuron, so as to optimize a target distance between the network's predicted output (contained in the output layer) and the real output.

Autoencoders. Autoencoders (AE) [115] are a type of neural network designed for unsupervised learning tasks. The aim is usually to learn a low dimensional representation of the data, called an encoding. The AE is composed of two parts: the *encoder* learns the new representation and the *decoder* simultaneously learns to reconstruct the original data from this representation. The AE thus seeks to approximate the identity function. When the encoder is composed of one fully connected layer which reduces the dimension, together with a *linear* activation function, its learned representation is essentially the same as that of a PCA projection of the same dimensionality [116, 117]: more precisely, the two models project on the same bottleneck linear space, but not using the same generating vectors. In general, however, AEs are used with *nonlinear* activation functions. This allows for nonlinear encoding functions, and thus potentially better encoders than those restricted to stay within the smaller class of linear functions. AEs can have different topologies depending on the learning task, the data size and dimensionality, etc. Here, we describe the general topology for a symmetrical, fully connected, dimensionality reducing (bottleneck) autoencoder. We denote by $\mathcal{X} \subseteq \mathbb{R}^D$ the data space, and by $\mathcal{A} \subseteq \mathbb{R}^d$ a lower dimensional space ($d < D$). The autoencoder can be represented by a mapping $f = f_{\text{dec}} \circ f_{\text{enc}}$ where $f_{\text{enc}} : \mathcal{X} \to \mathcal{A}$, $f_{\text{dec}} : \mathcal{A} \to \mathcal{X}$. It is symmetrical in structure, fully connected, and contains $2L$ layers. Each hidden layer is of dimension $d_\ell = d_{2L-\ell}$ for $\ell = 1 \dots L$, and the output layer is of dimension $d_{2L} = D$ (note that, by convention, the input layer does not count as a layer of the network). Each layer $\ell \in 1, \dots, 2L$ has an activation function g_ℓ and is connected to the previous layer by a projection matrix $W_\ell \in \mathbb{R}^{d_\ell \times d_{\ell-1}}$, and a bias vector $b_\ell \in \mathbb{R}^{d_\ell}$. There are thus $K = \sum_{\ell=1}^{2L} (d_\ell d_{\ell-1} + d_\ell)$ learnable real parameters denoted by $(\theta_1, \dots, \theta_K) \in \mathbb{R}^K$. As the activation functions are predefined and do not change during learning, the autoencoder function is fully described by its parameters $\theta = (\theta_k)_{k=1,\dots,K}$. We indicate this dependence as f_θ. The general formula for f_θ is:

$$\forall x \in \mathcal{X}, \qquad f_\theta(x) = g_{2L}\left[b_{2L} + W_{2L}\, g_{2L-1}\left(b_{2L-1} + W_{2L-1} \dots g_1(b_1 + W_1 x)\right)\right],$$

where the activation functions are by convention applied element-wise: for $z = (z_1, \dots, z_{d_\ell})$, it holds that $g_\ell(z) = (g_\ell(z_1), \dots, g_\ell(z_{d_\ell}))$. Note that f_θ, and thus f_{enc}, are differentiable

functions when all the activation functions $(g_\ell)_{1\leq\ell\leq 2L}$ are. The CV learned by this method is represented by the encoder part of the model:

$$\xi(x) = f_{\text{enc}}(x).$$

Figure 1.2 illustrates the typical structure of a symmetric autoencoder.

To make the model more robust to overfitting, and/or to make it learn useful information about the data distribution, variations of the original AE model were devised. Regularized autoencoders add regularization to the model, e.g. by introducing sparsity to the model using ℓ_1 or ℓ_2 regularization, or white noise to the input data. Variational autoencoders (VAE) [118] combine the autoencoder structure with variational Bayesian inference to produce a probabilistic encoder and decoder, which learn the latent space as a distribution from which the encoding is sampled, rather than modeling it as a deterministic vector. A variety of methods make use of autoencoders or variational autoencoders in their CV learning framework [93, 94, 100, 119–123]. Here, we present two methods which use AEs (or VAEs) to iteratively learn CVs and perform enhanced sampling to generate additional data.

Autoencoders are a central part of FEBILAE, the iterative algorithm for CV learning and biasing developed during this thesis and presented in Chapter 2 in the form of the article published in [19]. In the rest of this section, we present two other iterative algorithms which also use autoencoders: Molecular enhanced sampling with autoencoders (MESA) and reweighted autoencoded variational Bayes for enhanced sampling (RAVE).

Molecular enhanced sampling with autoencoders. MESA [93] is an iterative method for free energy biasing and CV discovery using bottleneck autoencoders. Starting from an unbiased simulation, an autoencoder is trained to extract a first low dimensional CV ξ_0. This CV is then used to perform enhanced sampling (precisely umbrella sampling in the MESA work) and thus sample additional data, exploring a larger region of the configurational space. A new autoencoder is then trained on the new available biased data, resulting in a new CV ξ_1, and so on, until CV convergence is reached. This convergence is assessed by applying a linear regression between consecutive CVs and computing a precision score for this model. A final round of umbrella sampling is then performed to compute the free energy surface of the final CV.

MESA also addresses the issues of rotational invariances between configurations with the use of data augmentation as an alternative to the more straightforward rotational alignment to a reference structure. By applying $p \in \mathbb{N}$ random rotations to each configuration, MESA optimizes the autoencoder using the loss function:

$$L = \sum_{k=1}^{n} \sum_{j=1}^{p} \| f\left(R_j(x^k)\right) - A(x^k, x^{\text{ref}}) \|^2,$$

where $A(x^k, x^{\text{ref}})$ is the configuration x^k aligned to the predefined reference structure q^{ref}, $R_j(x^k)$ is the jth rotation applied to x^k, and f is the autoencoder function. Through this loss, the autoencoder learns to apply an alignment of all configurations to the reference structure x^{ref}, and to ignore rotational differences between otherwise similar configurations. The authors also suggest using more than one reference configuration to obtain an autoencoder function that is independent of the orientation of the molecules.

In a subsequent work [94], innovations in autoencoder structure and loss function design are explored to improve the CVs learned by MESA. More precisely, multiple types of

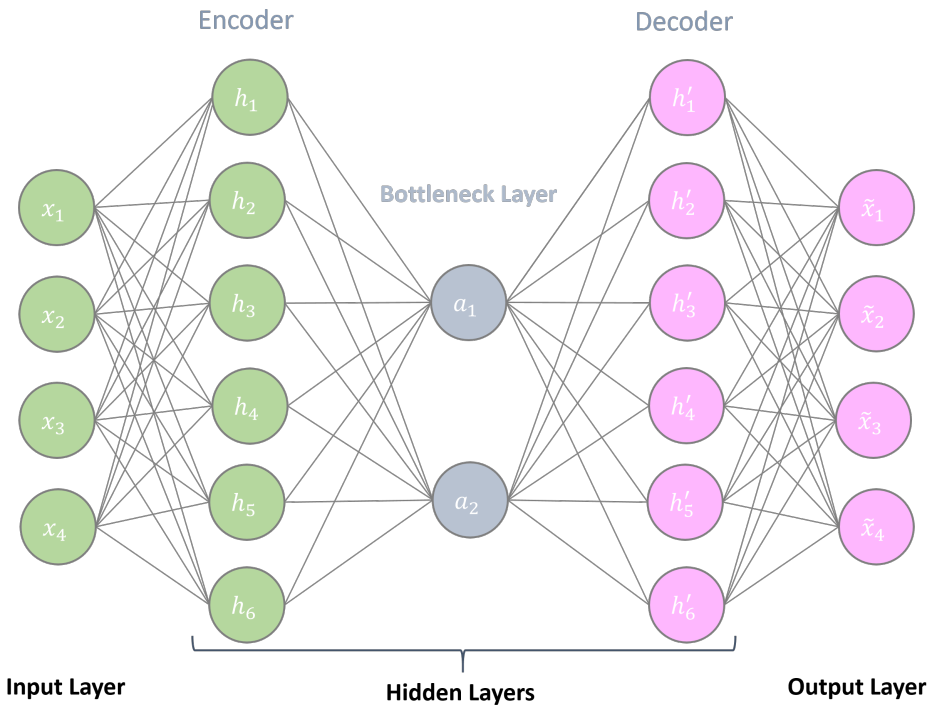


Figure 1.2: Example of an autoencoder topology. Here, the dimension of the data is $D = 4$ and there are $2L = 4$ layers: 3 hidden layers (including the bottleneck) and the output layer, of respective dimensions $d_1 = 6$, $d_2 = 2$, $d_3 = 6$ and $d_4 = 4$. The parameters can be represented by four matrices W_1, W_2 (for the encoder) and W_3, W_4 (for the decoder), such that: $\mathbf{a} = (a_1, a_2) = f_{\text{enc}}(\mathbf{x}) = g_2(b_2 + W_2 g_1(b_1 + W_1\mathbf{x}))$ and $\tilde{\mathbf{x}} = f_{\text{dec}}(\mathbf{a}) = g_4(b_4 + W_4 g_3(b_3 + W_3\mathbf{a}))$ where $g_1, \ldots, g_4$ are activation functions, and $b_1, \ldots, b_4$ are biases.

rotational alignments to the reference structure are tested, such as backbone atoms alignments, helix alignments, etc. As for the autoencoder structure, hierarchical layers are used to obtain a ranking of the CV dimensions. Indeed, a regular encoder projects on a low dimensional subspace whose dimensionality is predefined before training as the dimension of the bottleneck layer, and whose coordinates are a priori equally important, unlike e.g. PCA, where the directions are ranked by their corresponding eigenvalues. Hierarchical autoencoders modify layer to layer connections to ensure CV ranking. For example, if the bottleneck layer contains d nodes (neurons), the first node is fully connected to the previous layer, the second node is connected to e.g. 70% of the nodes from the previous layer and so on. Decreasing levels of information are thus contained in each node.

Reweighted autoencoded variational Bayes for enhanced sampling. RAVE [100] is also an iterative procedure for on-the-fly CV learning and biasing. It makes use of variational autoencoders as the learning model, which enables learning the distribution of the latent variable $p(z)$. In the original algorithm, the method selects a CV χ from a predefined set of M candidate variables $\chi_1, \ldots, \chi_M$, by minimizing the Kullback-Leibler distance

between the (empirical) distribution $p(\chi_i(x))$ of the candidate CVs and the learned distribution of the encoder latent variable $p(z)$. The distribution of the selected CV is then used to bias the dynamics by changing the original potential V into $\tilde{V}(x) = V(x) + \frac{1}{\beta}\ln(p(\chi(x)))$. The CV learned in applications of RAVE is always 1-dimensional, but the method can in theory be used with multi-dimensional CVs. The CV can be computed for any point x as:

$$\xi(x) = \chi(x).$$

Alternatively, the VAE can be used to generate the CV

$$\xi(x) = f_{\text{enc}}(x).$$

Note that the above equation is a simplification of the actual CV which is actually obtained as a point drawn from the distribution $p(z|x)$ learned by the model. If a deterministic function is preferred, the mean of this distribution can for example be used instead as the CV.

1.2.2.D Learning a differentiable CV mapping

Some of the methods presented above, e.g. diffusion maps and Isomap, share a major limitation: the lack of a clear differentiable mapping from the data space to the coordinate space. Indeed, the values $\xi(x^i)$ of the CV are only computed for the samples x^i belonging to the dataset X, making the method incompatible with most collective variable biasing algorithms, namely ABF and eABF. Here, we list some methods devised to generate a CV function ξ by interpolation when the CV discovery method does not.

A simple interpolation method is the kernel PCA projection, or equivalently the Nyström extension formula [124]. This method computes the interpolation for kernel based methods (e.g. Isomap or diffusion maps) using the kernel function itself. More precisely, the projection of the eigenvector v_i over a new point x is computed as:

$$v_i(x) = \frac{1}{\lambda_i}\sum_{j=1}^{n}\frac{h(x^j, x)}{H(x)}v_i^j,$$

where h is the kernel function, v_i^j is the eigenvector projection on x^j and $H(x) = \sum_{j=1}^{n} h(x^j, x)$.

Smooth and nonlinear data driven CV [125] (SandCV) is another interpolation method initially devised for Isomap, but which can be used with other CV learning methods. Given a new configuration x that is not part of the initial dataset used to compute, e.g. the Isomap, the SandCV mapping is defined as:

$$C(x) = (\mathcal{M}^{-1} \circ \mathcal{P})(x).$$

The function $\mathcal{P}$ is a projection of the sample x to its nearest neighbor in the initial dataset X, and $\mathcal{M}$ is a parametrization of a function which maps the CVs to the samples, computed as a linear combination of a number L of smooth basis functions associated to L landmark points η_i selected from the CV space.

Methods for constructing differentiable mappings of a previously computed CV have also been devised with the use of neural networks. For example, diffusion nets [126] use neural networks to derive diffusion map CVs, where the input are samples x^i from the dataset X and the corresponding outputs are their projections using the computed diffusion map eigenvectors. Similarly, the ANNcolvar [127] uses neural networks to map the input coordinates X to a precomputed collective variable space. The trained neural network is then used to predict the value of the CV for new samples.

1.2.3 Using input features and time information

The methods presented in this section make use of time series data, i.e. sampled MD trajectories. Most of these methods are more or less related in their formulation to the Koopman transfer operator $e^{\tau\mathcal{L}}$, where $\tau > 0$ is a preselected hyperparameter called the lag-time and $\mathcal{L}$ is the generator of the dynamics under consideration. We begin this section by briefly recalling the definition and properties of this operator. We then move on to presenting popular state of the art methods whose aim is often to approximate (explicitly or implicitly) the Koopman operator, or more precisely its eigenfunctions.

1.2.3.A Koopman models

The description of the Koopman models given here is based on [128] and [129], and we refer to these papers for detailed proofs and results. Koopman models are a family of methods which approximate the Koopman operator

$$\mathcal{K}_\tau = e^{\tau\mathcal{L}},$$

defined by the following equation: for all bounded measurable observables f:

$$(\mathcal{K}_\tau f)(x) = \mathbb{E}\left(f(x_{t+\tau})|x_t = x\right) \ .$$

As the Koopman operator describes the average time evolution of the system observables, its eigenfunctions and eigenmodes are relevant representations of the dynamics of the system. It is thus reasonable to assume that the eigenfunctions corresponding to the largest eigenvalues of the Koopman model can constitute suitable collective variables. In fact, the eigenfunctions of the Koopman operator are also eigenfunctions of the generator $\mathcal{L}$, which are proven to be good collective variables by several requirements in Section 1.1.5. The eigenvalues λ_i^τ of $\mathcal{K}_\tau$ are also related to those of $\mathcal{L}$:

$$\lambda_i^\tau = e^{\lambda_i \tau}.$$

Of course, the Koopman operator is infinite dimensional and in most cases, its analytical expression is unknown. Koopman models use a finite dimensional, data-driven approximation of the operator and apply spectral decompositions to obtain its dominant eigenpairs. The data consists of one (or several) trajectory $X = (x^1, \dots, x^n)$ of n configurations, and the corresponding τ-shifted conformations, which we denote by $Y = (y^1, \dots, y^n)$. X and Y are projected on a dictionary of basis functions $\Psi = (\Psi_1, \dots, \Psi_M)$. The input data is thus M dimensional and is denoted by $\Psi_X = (\Psi(x^1), \dots, \Psi(x^n)) \in \mathbb{R}^{n\times M}$ and $\Psi_Y = (\Psi(y^1), \dots, \Psi(y^n)) \in \mathbb{R}^{n\times M}$.

1.2.3.B Variational approach to conformational dynamics

The variational approach to conformational dynamics [130–132] (VAC) is based on the variational Rayleigh principle:

Theorem 3. *For a given **self-adjoint** operator K on a Hilbert space with d largest eigenvalues $\lambda_1, \dots, \lambda_d$, it holds:*

$$\sum_{k=1}^{d} \lambda_k = \sup \sum_{k=1}^{m} \langle K v_k, v_k \rangle \,,$$

where the maximization is taken over all possible orthonormal families of m vectors $v_1, \dots, v_d$, i.e. $\langle v_i, v_j \rangle = \delta_{ij}$. The maximum value is reached for the first m eigenvectors of the operator K associated with $\lambda_1, \dots, \lambda_d$.

This theorem provides a way to approximate the eigenvectors of the Koopman operator on the Hilbert space $L^2(\nu)$: Given a dictionary of basis functions $\Psi_1, \ldots, \Psi_M$, VAC seeks orthonormal functions $f_1, \ldots, f_d$ in the space spanned by the basis functions:

$$f_k = \sum_{i=1}^{M} a_i^k \Psi_i = (a^k)^T \Psi,$$

such that

$$\sum_{k=1}^{d} \langle \mathcal{K}_\tau f_k, f_k \rangle_{L^2(\nu)}$$

is maximized. Here, we assume the basis functions $\Psi_1, \ldots, \Psi_M$ to be orthonormal, i.e., the covariance matrix $(C_0)_{ij} = \langle \Psi_i, \Psi_j \rangle_{L^2(\mu)}$ is the identity matrix. Note that $\langle \mathcal{K}_\tau f_k, f_l \rangle_{L^2(\mu)} = (a^k)^T C_\tau a^l$ where

$$(C_\tau)_{ij} = \langle \mathcal{K}_\tau \Psi_i, \Psi_j \rangle_{L^2(\mu)} = \mathbb{E}_\nu(\Psi_i(x_\tau)\Psi_j(x_0))$$

is the time-lagged correlation matrix. This yields:

$$\sup_{\substack{f_1,\ldots,f_d \\ \text{orthonormal}}} \sum_{k=1}^{d} \langle \mathcal{K}_\tau f_k, f_k \rangle_{L^2(\nu)} = \sup_{\substack{a^1,\ldots,a^d \\ \text{orthonormal}}} \sum_{k=1}^{d} (a^k)^T C_\tau a^k = \sup_{\substack{a^1,\ldots,a^d \\ \text{orthonormal}}} \sum_{k=1}^{d} \langle C_\tau a^k, a^k \rangle_{\mathbb{R}^M}.$$

Using the variational principle of Theorem 3 on the operator C_τ on the Hilbert space $\mathbb{R}^M$, this maximum is reached by choosing the a^k as the eigenvectors of C_τ corresponding to its d dominant eigenvalues.

In the general case, the basis functions $\Psi_1, \ldots, \Psi_M$ are not orthonormal but can be transformed into a set of orthonormal basis functions $\tilde{\Psi}_i = \sum_{j=1} M C_0^{-1/2}(j,i)\Psi_j$, which changes the eigenvalue problem for choosing the optimal orthonormal vectors a^k to:

$$C_\tau a^k = \lambda_k C_0 a^k.$$

The operator C_τ and C_0 are in practice approximated with the time-lagged covariance matrix and the covariance matrix respectively:

$$C_\tau = \frac{1}{n-1} \Psi_X \Psi_Y^T, \quad C_0 = \frac{1}{n-1} \Psi_X \Psi_X^T \ .$$

The matrix

$$M_{\text{VAC}} = C_0^+ C_\tau \in \mathbb{R}^{M \times M}$$

is thus an M-dimensional approximation of $\mathcal{K}_\tau$ (here, C_0^+ denotes the Moore-Penrose inverse of C_0). Eigenfunctions of $\mathcal{K}_\tau$ can thus be approximated by the eigenvectors $\phi_l \in \mathbb{R}^M$ of M_{VAC}, which solve the eigenvalue problem:

$$C_\tau \phi_l = \lambda_l C_0 \phi_l. \tag{1.38}$$

In practice, the definitions of C_0 and C_τ can also be made symmetrical

$$C_\tau^s = \frac{1}{2(n-1)} (\Psi_X \Psi_Y^T + \Psi_Y \Psi_X^T), \quad C_0^s = \frac{1}{2(n-1)} (\Psi_X \Psi_X^T + \Psi_Y \Psi_Y^T) \ ,$$

so that the eigenvalue problem (1.38) has real valued solutions. Note that, because the Rayleigh principle holds only for self-adjoint operators, and $\mathcal{K}_\tau$ is self-adjoint only when the dynamics are reversible, VAC can only be applied in the reversible case. An extension of the approach to the general case of non-reversible dynamics is the variational approach for Markov processes [129] (VAMP).

Remark 3. *The widely used time lagged independent component analysis (tICA) [133] is a special linear case of VAC where the basis functions are the identity functions, i.e., when no projection of the data is performed. In general, the choice of dictionary Ψ is the only part of VAC which introduces nonlinearity, and greatly impacts the computed approximation.*

The collective variable is then the projection over the d dominant eigenfunctions $a^1, \dots, a^d$:

$$\xi(x) = ((a^1)^T \Psi(x), \dots, (a^d)^T \Psi(x)).$$

1.2.3.C Extended dynamic mode decomposition

The extended dynamics mode decomposition (EDMD) [128] uses a norm minimization to approximate the Koopman operator. More precisely, EDMD computes the matrix

$$M_{\text{EDMD}} = \Psi_Y \Psi_X^+ = (\Psi_Y \Psi_X^T)(\Psi_X \Psi_X^T)^+ = C_\tau^T C_0^+ = M_{\text{VAC}}^T.$$

Its transpose M_{EDMD}^T is the solution to the minimization problem:

$$\min_{K \in \mathbb{R}^{M \times M}} \|\Psi_Y - K^T \Psi_X\|_F^2 = \min_{K \in \mathbb{R}^{M \times M}} \frac{1}{n} \sum_{j=1}^{n} \|\Psi(y^j) - K^T \Psi(x^j)\|_2^2 \ ,$$

where $\|.\|_F$ is the Frobenius norm. This problem is then proven to converge, when $n \to \infty$ to the minimization problem

$$\min \sum_{i=1}^{M} \|\mathcal{K}_\tau \Psi_i - \hat{K} \Psi_i\|_{L^2(\nu)}^2$$

where the minimization is over all linear mappings $\hat{K} : \mathbb{V} \to \mathbb{V}$, where $\mathbb{V} \subset L^2(\nu)$ is the space spanned by the basis functions Ψ_i. We refer to [128] for a proof of the equivalence of the two minimization problems. The transpose of the matrix M_{EDMD} is thus an approximation of the Koopman operator. Consequently, its left eigenvectors can be computed to approximate the eigenfunctions of the Koopman operator. The obtained CV is the same as the VAC CV, i.e. the eigenfunctions of the VAC matrix M_{VAC}.

1.2.3.D Markov state models

Markov state models (MSM) [129, 134, 135] are methods for modeling random processes using the Markovian assumption on states defined as subsets of the conformational space: the next state of the system is determined only by the current state. MSMs are extensively used in molecular dynamics [136, 137]. This section briefly summarizes the method. For a more extensive overview of the method, its variations and advances, we refer the reader to [134, 135] and references therein.

A general framework in MSMs starts with the decomposition of the configuration space into M states $s_1, \dots, s_M$. This decomposition is often done with the use of a classical

clustering method (e.g. K-means) on available simulated data, making it a purely geometric decomposition. Along with this decomposition, a lag time τ is defined. The lag time is chosen short enough to resolve the relevant slow timescales of the dynamics, and long enough to stay within the Markovian assumption: The probability of the new state of the system at time $t_{k+1} = t_k + \tau$ depends only on its state at time t_k. MSMs are thus a way of coarse graining the process under study: the smaller the number of states or the larger the lag time, the coarser the dynamics. Once the states and the lag time are defined, the MSM is constructed as a $M \times M$ transition matrix representing the transition probabilities during the time span τ between each pair of states (diagonal values represent the probabilities for each state to stay in the same state after τ). The transition matrix can be computed by, e.g., a Maximum Likelihood estimator or using Bayesian methods for example [138–140].

Importantly, when conditions of reversibility and ergodicity are fulfilled, the MSM transition matrix is proven to approximate the Koopman operator [141]. In fact, MSMs can be seen as a special case of VAC where the basis dictionary is defined as the M indicator functions corresponding to the M states $\Psi_i(x) = \mathbb{1}_{s_i}(x)$. The eigenvalue decomposition of the transition matrix is thus performed to gain more insight into the system dynamics and the obtained eigenvectors are approximations of the transfer operator eigenfunctions.

Often, the initial M state decomposition is done using a very high number of states M. Consequently, a possible additional step, called dynamical clustering, is to coarse grain the MSM into a smaller number of states using the computed transition matrix [142–144].

1.2.3.E Learning the basis dictionary of Koopman models

As mentioned in Remark 3, the choice of the basis dictionary Ψ is crucial to methods such as VAC, VAMP or EDMD. Indeed, the basis functions are the only step of these methods that introduces non-linearity to the learned CVs. Additionally, the basis dictionary determines the space of matrices over which the Koopman operator is approximated (as linear combinations of the basis functions). As a consequence the quality of the approximation is deeply impacted by this choice. To automatize the selection of a proper dictionary, new methods using neural networks coupled with a well defined optimization function were devised. VampNets [145] use the VAMP variational principle [129] to construct a score for any given basis dictionary. A neural network Ψ_{NN} which maps inputs to the basis functions is then trained to optimize this score. Similarly, state free reversible VampNets [146] also use neural networks to learn an optimal basis dictionary, but are restricted to reversible dynamics and use VAC principles to construct their learning score. The more recent Deep-tICA method [147] combines this neural network based approach for estimating eigenvectors of tICA/VAC with the iterative procedure presented in works such as RAVE and MESA to perform on the fly enhanced sampling and CV learning. Generally, the obtained CV is the VAC/VAMP projection over the neural network learned dictionary mapping:

$$\xi(x) = ((a^1)^T \Psi_{\text{NN}}(x), \dots, (a^d)^T \Psi_{\text{NN}}(x)).$$

1.2.3.F Time-lagged autoencoder, variational dynamics encoders

Time lagged autoencoders [122] have the same general structure as the original autoencoders described in Section 1.2.2.C, but learn to output a time lagged conformation instead of the input conformation. In the same notations as used in Section 1.2.3.A, the learning dataset is X and the desired output is Y (instead of X itself). The intuition is that the obtained bottleneck will be optimized to learn the slowly varying features of the input,

i.e. coordinates that do not change much in the time-lag τ, implying that the obtained low dimensional representation will capture the slow modes of the dynamics. In a similar fashion to the analogy between linear autoencoders and PCA transformation, it is shown in [122] that linear time-lagged autoencoders actually learn the same CV as tICA (in the reversible case). This implies that time-lagged autoencoders could be intuitively thought of as a deep learning based approximation of VAC where the dictionary Ψ is learned as part of the encoder.

Variational dynamics encoders [123] also introduce time-lag to learn slow CVs, but learn these CVs using variational rather than regular autoencoders. As for regular autoencoders, the collective variable is the encoder mapping and can be computed for any point x as

$$\xi(x) = f_{\text{enc}}(x).$$

1.2.3.G Past-Future information bottleneck

In what could be seen as a variation of RAVE, the authors of the original method [100] devised the past-future information bottleneck [119]. In this modification, the autoencoder is trained to learn a time lagged conformation from the input conformation, instead of the input itself. Additionally, the bottleneck CV is learned as a linear combination of the predefined RCs $\chi_1, \ldots, \chi_M$, and its learned distribution is directly used in the biasing procedure, instead of being matched to the distribution of a single putative RC χ. The CV can be computed for any point x as

$$\xi(x) = f_{\text{enc}}(x) = \sum_{i=1}^{M} w_i \chi_i(x).$$

1.2.4 Using input features and knowledge of conformational states

One of the least explored fields of ML for collective variable discovery is supervised learning. This is more than justified by the fact that supervised learning models rely on the knowledge of a label set y associated with the dataset X. In molecular dynamics, this label set assigns for example a metastable state to each sample. Yet the assumption that these states are known and distinguished makes supervised learning models non applicable in many cases. Indeed, the assumption that not only are all the metastable states of the system known, but that conformations corresponding to each of these states have been identified, is very limiting. Especially, this implies that the transition to any new previously unknown state can be difficult to achieve or determine. Nevertheless, in cases where all conformational states are known and the samples can indeed be assigned to states, it seems essential to include this information to the learning of a collective variable, as this will help recover a CV which distinguishes the various conformational states by construction.

1.2.4.A Supervised learning decision functions as collective variables

Using simple supervised learning algorithms, namely support vector machine (SVM) and logistic regression (LR), the decision functions learned to separate metastable states are used as collective variables [148]. For simplicity, we consider a system formed of only two metastable states $A \subset \mathcal{X}$ and $B \subset \mathcal{X}$, such that $A \cap B = \emptyset$.

Support vector machine. Support vector machine (SVM) is a linear classifier which separates the space of features using the hyperplane that maximizes the gap between the two classes, i.e. the minimal distance between two points from different classes. The method optimizes the projection coefficients $w \in \mathbb{R}^D$ and the bias $b \in \mathbb{R}$ such that for every configuration $x^j \in \mathbb{R}^D$ with label $y^j = -1$ (state A) or $y^j = 1$ (state B):

$$y^j(w^T x^j - b) \geq 1 \ .$$

By optimizing w and b to satisfy the equality above for all x^j and y^j, the hyperplane $w^T x - b = 0$ represents a separation between the two classes: State A (i.e. class $y = -1$) falls on one side of the hyperplane, and State B on the other. This version of SVM is called hard-margin SVM and works under the assumption that the data is linearly separable. This is often not the case, and instead the soft-margin version of the method is used, where the optimal coefficients and bias are obtained by solving:

$$\min_{w \in \mathbb{R}^D, b \in \mathbb{R}} \sum_{j=1}^{n} \max\left(0, 1 - y^j(w^T x^j - b)\right) + \lambda \|w\|_2^2 \ .$$

Here λ is a term used to determine a tradeoff between increasing the margin size (λ large) and ensuring the datapoints are all classified correctly (λ small). The distance of each point to the SVM hyperplane is then used as the collective variable, for any x:

$$\xi(x) = \frac{w^T x - b}{\|w\|_2}.$$

Logistic regression. Using the same notation as before, the logistic regression (LR) classifier models the conditional probability $P(y = 1|x)$ as a sigmoid function:

$$P(y = 1|x) = \sigma(w^T x - b) \ ,$$

where $\sigma(z) = \frac{1}{1+e^{-z}}$. The coefficients $w \in \mathbb{R}^D$ and bias b are then optimized to maximize the log-likelihood of the model.

$$\begin{aligned}
\sum_{j=1}^{n} \log(P(y = y^j|x^j) &= \sum_{j=1}^{n} \mathbb{1}_{y^j=1} \log(P(y = 1|x^j)) + \mathbb{1}_{y^j=-1} \log(1 - P(y = 1|x^j)) \\
&= \sum_{j=1}^{n} \mathbb{1}_{y^j=1} \log\left(\frac{1}{1 + e^{-w^T x+b}}\right) + \mathbb{1}_{y^j=-1} \log\left(\frac{e^{w^T x-b}}{1 + e^{-w^T x+b}}\right) \\
&= \sum_{j=1}^{n} \log\left(\frac{1}{1 + e^{-y^j(w^T x-b)}}\right).
\end{aligned}$$

The loss to minimize is thus:

$$\min_{w \in \mathbb{R}^D, b \in \mathbb{R}} \sum_{j=1}^{n} \log\left(1 + \exp\left(-y^j(w^T x^j - b)\right)\right) + \lambda \|w\|^2 \ ,$$

where λ is a regularization parameter. The probability $P(y = 1|x)$ represents the decision function of the logistic regression model, and is used as a collective variable which can be computed for any point x:

$$\xi(x) = P(y = 1|x) = 1 - P(y = -1|x) = \sigma(w^T x + b).$$

Alternatively, the odds ratio $\frac{P(y=1|x)}{1-P(y=1|x)}$ can be used.

Note that any linear or nonlinear supervised learning method used for classification can be used here instead of SVM or LR, as long as the corresponding decision function is a differentiable mapping. Additionally, both SVM and LR can be extended to handle situations with more than two classes, the corresponding decision function being d-dimensional when the number of classes (i.e. states) is $d+1$.

1.2.4.B Linear discriminant analysis

Linear discriminant analysis (LDA) is a linear supervised learning method for classification and dimensionality reduction which uses Fisher's discriminant. Given the dataset X of n samples, each belonging to one of C classes, the method separates the scatter (or the covariance) matrix of the dataset into two components: the within scatter matrix

$$S_w(X) = \sum_{c=1}^{C} \left[\sum_{i \in s_c} (x_i - \bar{x}_c)(x_i - \bar{x}_c)^T \right],$$

and the between scatter matrix

$$S_b(X) = \sum_{c=1}^{C} n_c (\bar{x}_c - \bar{x})(\bar{x}_c - \bar{x})^T.$$

Here, s_c, n_c and $\bar{x}_c$ are respectively the set, the number and the mean of the samples belonging to class c, and $\bar{x}$ is the mean of all samples. The method then seeks to maximize the Fisher discriminant:

$$\max_{W \in \mathbb{R}^D} J(W), \qquad J(W) = \frac{W S_b(X) W^T}{W S_w(X) W^T} \ . \tag{1.39}$$

This optimization problem seeks a projection of the data $\xi(X) = WX$ into the direction W which maximizes the scatter between classes $\tilde{S}_b(\xi(X)) = W S_b(X) W^T$, while minimizing the scatter within each class $\tilde{S}_w(\xi(X)) = W S_w(X) W^T$. The optimization problem (1.39) is then reformulated and solved as a generalized eigenvalue problem:

$$S_b(X) W = \lambda S_w(X) W.$$

Note that the above eigenvalue problem has at most $C-1$ non-zero eigenvalues as by construction, S_b has rank at most $C-1$. Indeed, S_b is a sum of C outer product matrices $(\bar{x}_c - \bar{x})(\bar{x}_c - \bar{x})^T$, thus each of rank at most 1, and $\sum_{c=1}^{C} n_c(\bar{x}_c - \bar{x}) = 0$.
The projection

$$\xi(x) = (W_1 x, \ldots, W_{C-1} x)$$

is the learned CV.

In harmonic LDA [149] the computation of the within scatter matrix is changed into a harmonic average instead of an arithmetic average of the class specific scatter matrices. The obtained CV is shown to be in practice more efficient for class separation. In deep LDA [150], the Fisher discriminant is used in the loss function of a nonlinear neural network, making the network efficiently learn nonlinear descriptors or features, and use LDA on these descriptors to obtain the CV.

1.2.4.C Multitask learning, extended autoencoders

The autoencoder structure introduced in Section 1.2.2.C can be used to include information on metastable states by adding a second output layer linked to the bottleneck CV layer and independent of the decoder. This second output optimized on a supervised learning task (e.g. state classification) simultaneously as the remainder of the autoencoder learns dimensionality reduction and input reconstruction, in a multi-task learning fashion [151]. This simple idea is the basis of the extended autoencoders introduced in [152], where the authors use an autoencoder which includes the usual decoder output, and a nonlinear regression output trained to learn the committor function. As mentioned above, both outputs are independently computed from the bottleneck CV layer. Consequently, the learned CV contains a mix of information necessary for both the decoder reconstruction and the accurate prediction of the committor.

1.3 Main Contributions

In this section, I briefly describe the main contributions of this work. The first one in Section 1.3.1 concerns our iterative algorithm for CV discovery, which uses sample reweighting to ensure convergence of the learned CV. The second contribution, summarized in Section 1.3.2 concerns the use of machine learning methods to learn collective variables, guide biased simulations and sample transitions between metastable states of HSP90. I also participated in the writing of an overview article [20] following a CECAM (Center Européen de Calcul Atomique et Moléculaire) discussion meeting [1]. The meeting was entitled "Coarse-graining with Machine Learning in Molecular Dynamics". It was co-organized by Sanofi, Ecole des Ponts ParisTech and Sorbonne University, and brought together a total of 29 participants.

1.3.1 Finding collective variables using autoencoders and biased trajectories

Machine learning and dimensionality reduction models require a certain amount of good quality data to learn relevant information. In the case of molecular dynamics, the availability of a "complete" trajectory, i.e. which visits the whole conformational states, is often the initial problem at hand due to the presence of metastability. A natural solution to overcome this difficulty is to use learning models in an iterative fashion. This can be done by training a model to learn a suboptimal CV from available data, which does not cover the whole space of conformations, then using this CV for enhanced sampling to generate additional learning data, and so on. Autoencoders have been used in iterative methods for CV learning in previous literature. For example, molecular enhanced sampling with autoencoders (MESA) [93, 94] is a framework that alternates between autoencoder learning of CVs and free-energy biasing along those CVs. This kind of iterative procedure can in theory be used for any of the dimensionality reduction models presented in Section 1.2 and any collective variable biasing method, i.e. the reaction coordinate based free energy methods presented in Section 1.1.3. However, it is important to recall that biased sampling makes the distribution of the sampled configurations drift from the Boltzmann–Gibbs density. Changing the distribution of the data results in changing the loss function optimized by the model. In

[1] website: `https://cermics-lab.enpc.fr/cecam_ml_md/`

the case of an iterative algorithm for CV learning and enhanced sampling, this implies that at each iteration, the training data coming from a biased simulation has a different distribution, and the model thus optimizes a different loss. This means that iterative models are not guaranteed to converge to a certain CV, regardless of whether they end up obtaining a good sampling of the conformational space.

Chapter 2 consists of a reprint of the work published in [19] where we present a new iterative algorithm for CV learning with autoencoders, named FEBILAE for "Free Energy Biasing and Iterative Learning with AutoEncoders". Our algorithm is inspired from methods such as MESA, but adds a simple yet crucial reweighting step of the data sampled from free energy biased simulations, so as to ensure the consistency of the optimized loss, and thus the convergence of the learned CVs. Figure 1.3 shows the general framework of FEBILAE.

As mentioned above, this iterative algorithm can be used with any collective variable based biasing technique and any dimensionality reduction method that provides a differentiable mapping from the configurational space to the CV space. In this study, we use autoencoders combined with eABF 1.1.4.B. This implementation is coined AE-ABF. The method is applied to three systems of increasing complexity:

- A 2D toy system of a three well potential. This system is first used fto demonstrate how the reweighting procedure can be essential for the convergence of the iterative method. The AE-ABF method is then applied to this system, and convergence of the CV, as well as full sampling of the conformational space are reached after 4 iterations.
- Alanine dipeptide in vacuum. This simple system is extensively used in ML applications for MD simulation analyses. Here, the AE-ABF methods learns a CV of alanine dipeptide after $4-5$ iterations. This CV is then proven upon extensive analysis to be highly correlated to the dihedral angles Φ and Psi, and almost match them in sampling efficiency.
- Chignolin mini-protein. AE-ABF is also applied to this small 10-residue protein, whose folded, misfolded and unfolded states are adequately described by the learned autoencoder CV. This variable's efficiency for biased sampling of the chignolin conformational space is also showcased in comparison to a physically relevant CV of the system.

1.3.2 Exploring conformations of HSP90 using machine learning guided biased simulations

The 90 kDa heat shock protein (HSP90) is a large protein of great interest in cancer research, as well as many other diseases. Understanding its mechanism of action is central to developing drugs which target this protein. Previous works on this system show that the activity of the protein can be regulated by targetting a specific domain of it, the N-terminal domain (NTD), which binds to the ATP.

In Chapter 3, we use machine learning and clustering methods to drive biased MD of the solvated HSP90 NTD. First, clustering of various crystal structure of the NTD is applied to determine the conformational states of this domain. This clustering model is based on the dihedral angle values in a specific region of interest situated near the active site (i.e. the ATP binding site) of the NTD. Six conformational clusters are identified (see Figure 1.4) and conformations representing each states are selected. Then, short unbiased MD simulations are run from each of the identified states, in order to compose a large and structurally diverse dataset of conformations. This dataset is then used as a training set

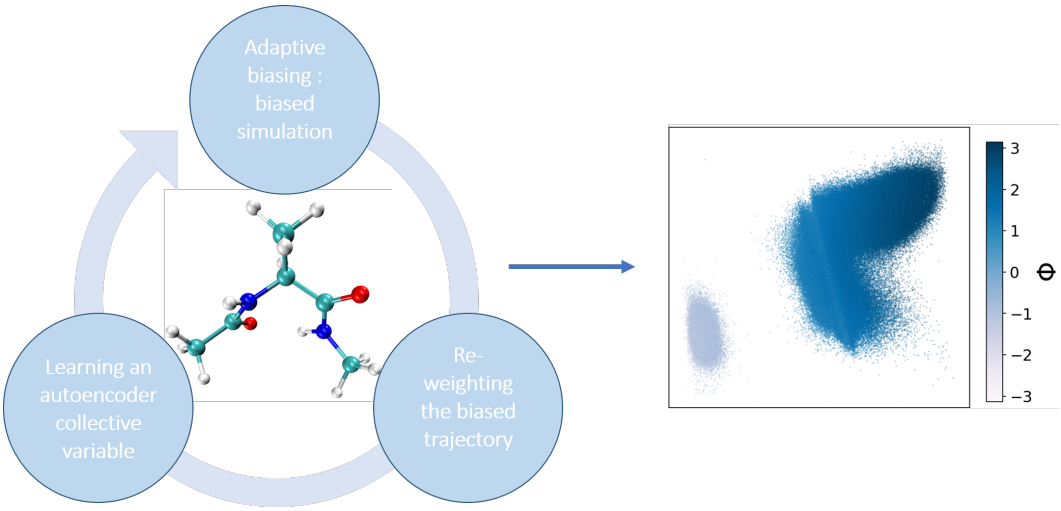


Figure 1.3: A general framework for FEBILAE

to learn collective variables of HSP90 using an autoencoder. The obtained autoencoder CV is analyzed to assess its efficiency for distinguishing between the NTD conformation states. Then, eABF simulations are performed using the autoencoder CV. To analyze the obtained trajectories, a protocol is devised for detecting transitions between the previously identified states. In this protocol, a pre-identification of possible transitions is first performed using the values of the autoencoder CV visited during the biased simulations. Then, further analysis is conducted, based on RMSD computations local to the region of interest near the active site. Additionally, dihedral clustering and dihedral PCA, are used to further investigate the structural changes in the active site region of interest throughout the biased simulation, and link these changes to the observed transitions. Finally, a hydrogen bonds based analysis is also conducted. While multiple biased simulations were performed withing this work, the transition identification protocol is mainly illustrated on one 100 ns simulation which is proven to achieve a transition between two of the identified states. Other biased simulations which achieved transitions are also illustrated in supplementary results. Generally, the obtained results show that the autoencoder CV provides a clear separation between the identified states of the HSP90 NTD. Additionally, using this CV for eABF simulations is shown to enable sampling of transitions between the conformational states.

1.4 Résumé de la thèse en langue francaise

Avec l'amélioration continue de la capacité de calcul des ordinateurs, les méthodes d'apprentissage automatique ont permi le développement de nouvelles solutions aux problèmes dans divers domaines. En particulier, l'apprentissage automatique a été largement utilisé au cours de la dernière décennie dans le domaine de la biochimie computationnelle et de la découverte et développement de nouveaux médicaments. Cela inclut l'application de méthodes d'apprentissage automatique pour la définition de nouvelles molécules, la détermination de sites importants dans les protéines ciblées, la conception de champs de force adéquats fondés sur des résultats expérimentaux ou encore l'amélioration de l'efficacité de l'échantillonnage

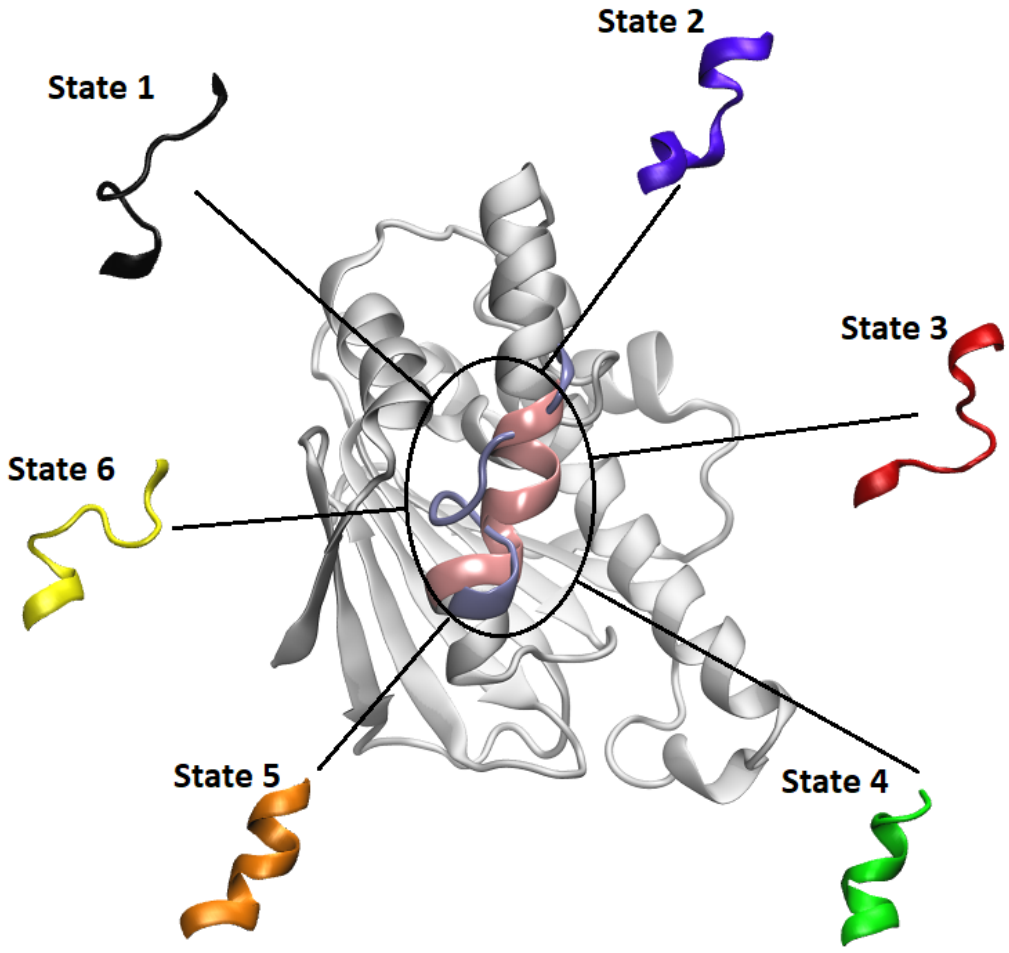


Figure 1.4: Six identified conformational states of HSP90

des conformations moléculaires d'un système donné. Cette thèse de doctorat se concentre sur la dernière tâche consistant à utiliser des méthodes d'apprentissage automatique pour améliorer l'échantillonnage en dynamique moléculaire. En effet, les simulations de dynamique moléculaire se sont avérées être un outil très utile en complément des expériences en laboratoire. Malgré leur large utilisation pour capturer les phénomènes rapides, il existe encore de nombreux cas où les échelles de temps accessibles aux simulations de dynamique moléculaire sont bien plus petites que les échelles de temps nécessaires pour l'observation des changements conformationnels importants du système, en raison de la présence de barrières hautes dans le profil énergétique.

Pour faire face à ce problème, plusieurs méthodes d'échantillonnage amélioré ont été conçues pour atténuer les difficultés d'échantillonnage associées à la métastabilité. La plupart de ces méthodes peuvent être globalement divisées en deux catégories selon qu'elles utilisent ou non des variables collectives (CV), également appelées coordonnées de réaction, qui sont des représentations de petite dimension du système:

- Les méthodes sans variables collectives modifient la distribution canonique en modifiant par exemple la température ou l'hamiltonien du système pour accélérer le franchissement de barrières énergétiques ou entropiques. Cette catégorie comprend, par exemple, le Simulated Tempering, le Parallel Tempering, le Replica Exchange MD, la simulation multicanonique, la dynamique accélérée par température citetemp4 ou l'algorithme de Wang-Landau.

- Les méthodes utilisant des variables collectives modifient la dynamique du système en ajoutant un biais afin d'accélérer la dynamique en aplatissant les barrières énergétiques le long d'une CV choisi. La plupart de ces méthodes calculent simultanément l'énergie libre associée à ces CV. Des exemples notables incluent la Metadynamics qui biaise l'évolution du système en utilisant un potentiel construit comme une somme de variables gaussiennes centrées le long de la trajectoire de la CV; de même, l'Umbrella Sampling ajoute un potentiel de biais le long des CV, souvent de forme harmonique, pour forcer le système à visiter des régions intermédiaires entre des états métastables. Les méthodes ABF pour adaptive biasing force ajoutent une force à la dynamique du système afin d'éliminer toute force moyenne agissant le long des CV, rendant la dynamique plus diffusive le long de ces directions. L'efficacité de ces méthodes repose de manière cruciale sur la connaissance préalable d'une ou plusieurs CV qui contiennent la plupart des fonctions métastables de la dynamique, et distinguent ainsi clairement les états métastables.

Dans des cas simples, la CV peut être choisie suivant l'intuition et la connaissance préalable du système en question. C'est par exemple le cas de la dialanine, molécule pour laquelle deux CV connues sont les angles dièdres ϕ et ψ. Pour les systèmes plus complexes, des CV adéquates sont souvent difficiles à deviner. Dans ce cas, les variables collectives peuvent être identifiées à l'aide d'algorithmes d'apprentissage automatique et de réduction de dimensionalité. En plus d'être utilisées pour accélérer l'échantillonnage, les variables collectives construites par apprentissage automatique aident également à acquérir une connaissance précieuse du système étudié, à savoir en facilitant la visualisation de ses différents états, ainsi que de son profil d'énergie libre.

Dans ce travail, d'importantes notions et définitions de la dynamique moléculaire sont d'abord présentées avant de passer en revue les algorithmes d'apprentissage automatique de pointe qui ont été conçus ou appliqués ces dernières années pour la construction automatique de variables collectives. Ensuite, la méthode développée au cours de cette thèse, baptisée "Free energy biasing and machine learning with autoencoders" (FEBILAE), est introduite. Cette méthode utilise un schéma itératif pour générer alternativement de nouvelles simulations et apprendre les variables collectives à partir de ces simulations en utilisant des autoencodeurs. Enfin, nous présentons l'application de méthodes d'apprentissage automatique à un véritable système d'intérêt. Ici, des autoencodeurs sont utilisés pour apprendre les variables collectives de la protéine chaperone HSP90, dans le but d'effectuer des simulations biaisées de ce système.

1.4.1 Recherche de variables collectives à l'aide d'auto-encodeurs et de trajectoires biaisées

Tout modèle d'apprentissage ou de réduction de dimensionnalité nécessite une certaine quantité de données de bonne qualité pour apprendre des informations pertinentes. Dans le cas de la dynamique moléculaire, le manque de données (échantillonnage incomplet) est le

problème initial qui se pose. Une solution naturelle pour surmonter cette difficulté consiste à utiliser des modèles d'apprentissage de manière itérative. Cela peut être fait en construisant des modèles optimisées via les données disponibles, puis en utilisant les CV apprises pour un échantillonnage amélioré afin de générer des données d'apprentissage supplémentaires, et ainsi de suite. Les auto-encodeurs ont été utilisés dans des méthodes itératives pour l'apprentissage de CV dans la littérature précédente. Par exemple, MESA (échantillonnage amélioré par autoencodeurs) est une méthode qui alterne entre l'apprentissage par autoencodeur des CV et le biaisage d'énergie libre (plus précisément par Umbrella Sampling) le long de ces CV. Notons tout d'abord que ce type de procédure itérative peut en théorie être utilisé pour tout modèle de réduction de dimensionnalité et toute méthode de biais CV. Cependant, il est important de noter qu'un échantillonnage biaisé fait dériver la distribution des configurations échantillonnées de la densité de Boltzmann–Gibbs. La modification de la distribution des données entraîne la modification de la fonction optimisée par le modèle d'apprentissage automatique.

Dans le cas d'algorithmes itératifs d'apprentissage de CV et d'échantillonnage amélioré, cela implique qu'à chaque itération, les données d'apprentissage provenant d'une simulation biaisée ont une distribution différente, et le modèle optimise une fonction différente. Cela signifie que les modèles itératifs tels que MESA ne sont pas garantis de converger vers une certaine CV, qu'ils finissent ou non par obtenir un bon échantillonnage de l'espace de configuration.

Nous présentons ici un nouvel algorithme itératif pour l'apprentissage de CV avec autoencodeurs, nommé FEBILAE pour "Free Energy Biasing and Iterative Learning with AutoEncoders". Notre algorithme s'inspire de méthodes telles que MESA et ajoute une étape simple mais cruciale de repondération des données obtenues de simulations biaisées, afin de garder inchangée la fonction optimisée par les modèles d'apprentissage, ce qui permet la convergence des CV apprises.

L'algorithme FEBILAE peut être utilisé avec toute méthode de biaisage basée sur des variables collectives et toute méthode de réduction de dimensionnalité qui fournit une fonction différentiable de l'espace des configurations moléculaire à l'espace de CV. Dans ce travail, nous utilisons des auto-encodeurs combinés avec ABF. Cette implémentation est dénommée AE-ABF. La méthode est appliquée à trois systèmes de complexité croissante:

- Potentiel à trois puits en 2 dimensions. Ce système est d'abord utilisé pour montrer comment la procédure de repondération peut être essentielle à la convergence de la méthode itérative. La méthode AE-ABF est ensuite appliquée à ce système, et la convergence de la CV, ainsi que l'échantillonnage complet de l'espace conformationnel sont atteints après 4 itérations.

- Dialanine sous vide. Ce système simple est largement utilisé dans les applications d'apprentissage automatique pour léanalyse de simulations en dynamique moléculaire. Ici, les méthodes AE-ABF apprennent une CV dde la dialanine après $4-5$ itérations. Il est ensuite prouvé, après une analyse approfondie, que cette CV est fortement corrélée aux angles dièdres Φ et Psi, et qu'elle les égale presque en termes d'efficacité d'échantillonnage.

- Chignoline. AE-ABF est également appliqué à cette petite protéine de 10 résidus, dont les états replié, mal replié et déplié sont décrits de manière adéquate par la CV apprise par auto-encodeur. L'efficacité de cette CV pour l'échantillonnage biaisé de l'espace conformationnel de la chignoline est également présentée et comparée à une CV physiquement pertinente du système.

1.4.2 Exploration des conformations de HSP90 à l'aide de simulations biaisées guidées par apprentissage automatique

HSP90 est une protéine d'un grand intérêt pour la recherche sur le cancer, ainsi que sur de nombreuses autres maladies. La compréhension de son mécanisme d'action est essentielle au développement de médicaments ciblant cette protéine. Des travaux antérieurs sur ce système montrent que l'activité de la protéine peut être régulée en ciblant un domaine spécifique de celle-ci, le domaine N-terminal (NTD), qui se lie à l'ATP.

Dans ce travail, nous utilisons des méthodes d'apprentissage automatique et de clustering pour guider les simulations biaisé du NTD de HSP90. Tout d'abord, diverses structures cristallines du NTD sont regroupées pour déterminer les états conformationnels de ce système. Ce modèle de regroupement est basé sur les valeurs d'angles dièdres dans une région d'intérêt spécifique située près du site actif (c'est-à-dire le site de liaison à l'ATP) du NTD. Six clusters conformationnels sont identifiés (voir Figure 1.4) et les conformations représentant chaque état sont sélectionnées. Ensuite, de courtes simulations moléculaires non biaisées sont effectuées à partir de chacun des états identifiés, afin de composer un ensemble de données de conformations vaste et structurellement diversifié. Cet ensemble de données est ensuite utilisé comme ensemble d'apprentissage pour construire les variables collectives de HSP90 à l'aide d'un auto-encodeur. La CV obtenue est analysée pour évaluer son efficacité à distinguer les états confomationnels du NTD. Ensuite, des simulations ABF sont effectuées à l'aide de la CV. Pour analyser les trajectoires obtenues, un protocole est mis au point pour détecter les transitions entre les états précédemment identifiés. Dans ce protocole, une pré-identification des transitions possibles est d'abord effectuée à partir des valeurs de la CV visitées lors des simulations biaisées. Ensuite, une analyse plus approfondie est effectuée, basée sur des calculs RMSD localisés à la région d'intérêt à proximité du site actif. De plus, le clustering des angles dièdres et une ACP basée sue ces angles sont utilisés pour étudier plus encore les changements structurels dans la région du site actif d'intérêt tout au long de la simulation biaisée et lier ainsi ces changements aux transitions observées. Enfin, une analyse basée sur les liaisons hydrogène est également effectuée. Bien que plusieurs simulations biaisées aient été effectuées dans le cadre de ce travail, le protocole d'identification de transitions est principalement illustré sur une simulation de 100 ns qui réalise une transition entre deux des états identifiés. D'autres simulations biaisées qui ont réalisé des transitions sont également illustrées dans des résultats supplémentaires. Généralement, les résultats obtenus montrent que la CV de l'auto-encodeur fournit une séparation claire entre les états identifiés du NTD. De plus, l'utilisation de cette CV pour les simulations ABF permet l'échantillonnage des transitions entre les états conformationnels.

Chapter 2

Chasing collective variables using autoencoders and biased trajectories

This chapter is a reprint of the article published in [19] to the Journal of Chemical Theory and Computation.

2.1 Introduction

In the last decades, molecular dynamics (MD) simulations have helped gain insight into the microscopic and macroscopic properties of biomolecular processes. However, the time scales accessible to MD simulations are often significantly smaller than the times needed for the observation of slow conformational changes of the systems under study [1,2]. This is due to the presence of energy or entropy traps in the energy landscape, which causes the system to be stuck within metastable states and thus hinders the full exploration of the configurational space. As a consequence, thermodynamic quantities (obtained from trajectorial averages) can not be accurately estimated.

To cope with this issue, several methods for enhanced sampling have been designed to mitigate the sampling difficulties associated with metastability [3,4]. Most of these methods can be broadly divided into two categories according to whether or not they use collective variables (CV), also known as reaction coordinates, which are low dimensional or coarse-grained representations of the system:

- Collective variable free methods alter the canonical distribution by e.g. modifying the system temperature or the system Hamiltonian in order to accelerate crossing energetic or entropic barriers. This category includes, for example, simulated tempering [5], parallel tempering [6], replica exchange MD [7], multicanonical simulation [8], temperature-accelerated dynamics [9] or the Wang-Landau algorithm [10].

- Collective variable based methods modify the system's dynamics by adding a bias in order to accelerate the dynamics by flattening the energy barriers along a chosen CV. Most of these methods simultaneously calculate the free energy associated to these

CVs. Notable examples include metadynamics [11, 12], which biases the system's evolution using a potential constructed as a sum of Gaussian variables centered along the trajectory of the CV; likewise, umbrella sampling [13, 14] adds a biasing potential along the CVs, often of harmonic form, to force the system to visit intermediate regions between metastable states. Adaptive biasing force (ABF) methods [15–17] add a biasing force to the system dynamics so as to eliminate any average force acting along the CVs, rendering the dynamics more diffusive along these directions. The efficiency of these methods crucially relies on the prior knowledge of a proper low dimensional CV which contains most of the metastable functions of the dynamics, and thus in particular clearly distinguishes between the metastable states.

In simple cases, the CV can be chosen based on intuition and prior knowledge of the system at hand. This is for example the case for alanine dipeptide [153], for which two known CVs are the ϕ and ψ dihedral angles. For more complex systems, adequate CVs are often difficult to guess. There have been attempts in the last years to extract CVs using dimensionality reduction and machine learning (ML) methods [79].

Collective variable discovery methods range from simple linear projections such as principle component analysis (PCA) or factor analysis, to more elaborate algorithms involving non linear and/or dynamically relevant projections of the configurational space. Below, we recall some notable works on automatic CV discovery and design, but we do not aim to be exhaustive nor do we give a detailed comparison of these algorithms. For a more complete overview on optimization and learning methods for CV discovery and in molecular dynamics in general, we refer the reader to Refs.20, 82, 83. We choose here to distinguish between three broad categories of methods: Operator-based methods, which aim at building a coarse grained description of the dynamics; metastable state separation methods, which use supervised machine learning; and unsupervised learning methods for dimensionality reduction.

Operator-based methods account for the dynamical properties of the system under study by using the approximation of a transfer operator or generator associated with the dynamics. Notable examples are the variational approach to conformational dynamics [128,130] (VAC) and its linear version, time lagged independent component analysis (tICA), the extended dynamic mode decomposition [128,154] (EDMD), Diffusion Maps [107,108] or Markov State Models [134, 135], which can also be considered a special case of VAC. Methods like VAC can often be optimized by using a well defined dictionary of functions to represent the system (instead of using the system coordinates in the case of tICA [133]). For instance, VAMPNets [145] and state-free reversible VAMPNets [146] learn this dictionary using neural networks.

Methods for metastable state separation use supervised learning for feature selection and feature engineering [78,148,155]. Supervised learning models are trained on a labeled dataset and learn a mapping from the datapoints to their labels. In CV discovery, the dataset is the set of sampled configurations and the labels are usually their corresponding states (which are thus assumed to be known). In particular, feature selection based CV learning uses a set of candidate CVs as input features. Each feature is then given an importance score according to its contribution to the learned model, and the most relevant features are selected as final CVs. Feature engineering methods on the other hand compute a new CV from the input features. Notable examples include Ref 148 where decision functions learned by support vector machines are used as collective variables; Ref 156 where a 1-dimensional path CV is constructed as a nonlinear combination of classifier CVs obtained from a supervised neural network model; or Ref 150 where a neural network is also used in combination with an LDA

objective function.

Unsupervised learning models for CV identification, on the other hand, are trained on unlabeled datapoints (namely the sampled configurations without any additional information). These learning models aim at recovering patterns and intrinsic properties of the visited configurational space. Currently, the most used non linear ML dimensionality reduction method for CV discovery is auto-associative neural networks, also known as autoencoders (AE) [115]. Autoencoders are a type of neural networks that aim at learning low dimensional representations of the data. The network is composed of two parts. The first part is the encoder that learns the representation or encoding of the input. The second part is the decoder, it simultaneously learns to reconstruct the input from the encoding. Then, the obtained encoding map can be used as a CV. Many state-of-the-art methods for CV unsupervised learning include autoencoders in their framework. For example, time lagged autoencoders [122] train the decoder to predict the time-lagged configuration instead of the input configuration, thus factoring time evolution into the learned CV. Some works [121] use variational autoencoders [118], a class of models which combine autoencoder structure with Bayesian inference to learn a latent variable distribution through a probabilistic (instead of deterministic) encoder and decoder, resulting in a generative decoder in addition to the dimensionality reducing encoder. Variational dynamics encoders [123] employ the time-lag refinement on variational autoencoders. To learn CVs which resolve the different metastable states, the authors in Ref 120 use Gaussian mixture variational AEs. Another example is Ref 152 where extended autencoders are used to predict the committor function. Of course, many methods for unsupervised CV learning do not include autoencoders, instead using, e.g. Bayesian models [157] or information theory based methods [158].

Naturally, any learning or dimensionality reduction model requires a certain amount of good quality data to learn relevant information. In the case of molecular dynamics, the lack of data (incomplete sampling) is the initial problem at hand. A natural solution to overcome this difficulty is to use learning models in some iterative fashion. This can be done by training models on available data and using the learned CVs for enhanced sampling to generate additional learning data, and so on. Autoencoders have been used in iterative methods for CV learning in previous literature. For example, molecular enhanced sampling with autoencoders (MESA) [93, 94] is a framework that alternates between autoencoder learning of CVs and free-energy biasing (more specifically umbrella sampling) along those CVs. Note first that this kind of iterative procedure can in theory be used for any dimensionality reduction model and any CV biasing method. However, it is important to note that biased sampling makes the distribution of the sampled configurations drift from the Boltzmann–Gibbs density. Changing the distribution of the data results in changing the loss function optimized by any model. In the case of iterative algorithms for CV learning and enhanced sampling, this implies that at each iteration, the training data coming from a biased simulation has a different distribution, and the model optimizes a different loss. This means that iterative models such as MESA are not guaranteed to converge to a certain CV, regardless of whether they end up obtaining a good sampling of the configurational space. Even in cases when the search for a collective variable of the system is not the goal, this still poses the issue of how to design a stopping rule for the iterative procedure.

Here, we present a new iterative algorithm for CV learning with autoencoders, named FEBILAE for "Free Energy Biasing and Iterative Learning with AutoEncoders". Our algorithm is inspired from methods such as MESA, but adds a simple yet crucial reweighting step of the data sampled from free energy biased simulations, so as to ensure the consistency of the optimized loss, and thus the convergence of the learned CVs. Note that a reweighting

protocol is also used in other iterative methods, notably the reweighted autoencoded variational Bayes for enhanced sampling (RAVE) [100,119]. RAVE uses a variational autoencoder that takes as input one or more pre-selected variables and iteratively learns a CV and its distribution. This CV is then used for biasing. In RAVE, the learned CV is always one dimensional in practice and is learned as a linear combination of the preselected variables (through a linear encoder). This is the key difference of the specific method compared to FEBILAE: our CV is learned as a nonlinear function of the input coordinates, rather than a linear combination of a preselected set of order parameters or candidate collective variables. Consequently, the FEBILAE CV is representative of the whole system, and does not require prior knowledge of the system to select the input, although it lacks interpretability.

The FEBILAE algorithm can be used with any collective variable based biasing technique and any dimensionality reduction method that provides a differentiable mapping from the configurational space to the CV space. Moreover, in order to accelerate convergence, we propose an iterative procedure where information from each step, namely the trajectory data, the estimated free energy, or the learned model, are judiciously used in the following steps of the iterative procedure. We finally present the results of our implementation of the algorithm, which we coin AE-ABF, as it uses ABF to perform enhanced sampling using autoencoder learned CVs.

This article is organised as follows. We first provide in Section 2.2 an introduction to autoencoders and their use for dimensionality reduction and then move on to a theoretical analysis highlighting that the learned model is always dependent on the distribution of the training data. We continue by demonstrating how a simple reweighting procedure of the loss function can be applied to target a specific density distribution. We then introduce in Section 2.3 our algorithm for iteratively learning a CV, and present some refinements that can be incorporated to the method for more efficiency, and possibly for a faster convergence. For an immediate illustration of the theoretical points made in Sections 2.2 and 2.3, we intertwine the presentation of these theoretical concepts with numerical results obtained on a 2-dimensional toy example. Section 2.4 then summarizes the practical details and parameters of the implementation of AE-ABF for molecular systems. Finally, Section 2.5 presents the results obtained when applying AE-ABF to two different systems: alanine dipeptide in vacuum, and solvated chignolin. Section 2.6 contains our conclusions and some variants of the method which could be useful for applications to other systems. Finally, additional results and details are given in the supplementary material.

2.2 Learning autoencoder collective variables from (un)biased samples

We briefly introduce in Section 2.2.1 autoencoders and their use for dimensionality reduction. Section 2.2.2 then recalls the usual processing steps required to convert trajectory data to autoencoder inputs. We then describe in Section 2.2.3 how autoencoders are trained. We can then make precise in Section 2.2.4 how the training is impacted by the distribution of the data, and present a reweighting procedure to correct a bias in the distribution of data points. Finally, Section 2.2.5 introduces a 2-dimensional toy example, which we use to illustrate the points made in Sections 2.2.3 and 2.2.4 in particular.

2.2.1 Autoencoders

Autoencoders (AE) [115] are a type of neural network designed for unsupervised learning tasks. The aim is usually to learn a new representation of the data, called an encoding. The AE is composed of two parts: the *encoder* learns the new representation and the *decoder* simultaneously learns to reconstruct the original data from this representation. The AE thus seeks to approximate the identity function. When the encoder is composed of one fully connected layer which reduces the dimension, together with a *linear* activation function, its learned representation is essentially the same as that of a PCA projection of the same dimensionality [116, 117]: more precisely, the two models project on the same bottleneck space, but not using the same vectors. In general, however, AEs are used with *nonlinear* activation functions. This allows for nonlinear encoding functions, and thus potentially better encoders than those restricted to stay within the smaller class of linear functions.

AEs can have different topologies depending on the learning task, the data size and dimensionality, etc. Here, we describe the general autoencoder topology used in this work. We denote by $\mathcal{X} \subseteq \mathbb{R}^D$ the data space, and by $\mathcal{A} \subseteq \mathbb{R}^d$ a lower dimensional space ($d < D$). Figure 2.1 presents the typical topology of the AEs used in our work. The autoencoder can be represented by a mapping $f = f_{\text{dec}} \circ f_{\text{enc}}$ where $f_{\text{enc}} : \mathcal{X} \to \mathcal{A}$, $f_{\text{dec}} : \mathcal{A} \to \mathcal{X}$ and $\circ$ is the function composition operator i.e. $f_{\text{dec}} \circ f_{\text{enc}}(x) = f_{\text{dec}}\left(f_{\text{enc}}(x)\right)$. It is symmetrical in structure, fully connected, and contains $2L$ layers. Each hidden layer is of dimension $d_\ell = d_{2L-\ell}$ for $\ell = 1 \dots L$, and the output layer is of dimension $d_{2L} = D$ (Note that, by convention, the input layer does not count as a layer of the network). Each layer $\ell \in 1, \dots, 2L$ has an activation function g_ℓ and is connected to the previous layer by a projection matrix $W_\ell \in \mathbb{R}^{d_\ell \times d_{\ell-1}}$, and a bias vector $b_\ell \in \mathbb{R}^{d_\ell}$. There are thus $K = \sum_{\ell=1}^{L} d_\ell d_{\ell-1} + d_\ell$ learnable real parameters denoted by $(p_1, \dots, p_K) \in \mathbb{R}^K$. As the activation functions are predefined and do not change during learning, the autoencoder function is fully described by its parameters $\mathbf{p} = (p_k)_{k=1,\dots,K}$. We indicate this dependence as $f_{\mathbf{p}}$. The general formula for $f_{\mathbf{p}}$ is:

$$\forall x \in \mathcal{X}, \qquad f_{\mathbf{p}}(x) = g_{2L}\left[b_{2L} + W_{2L}\, g_{2L-1}\left(b_{2L-1} + W_{2L-1} \dots g_1(b_1 + W_1 x)\right)\right],$$

where the activation functions are by convention applied element-wise: for $z = (z_1, \dots, z_{d_\ell})$, it holds that $g_\ell(z) = (g_\ell(z_1), \dots, g_\ell(z_{d_\ell}))$. Note that $f_{\mathbf{p}}$, as well as f_{enc}, are differentiable functions when all the activation functions $(g_\ell)_{1 \le \ell \le 2L}$ are.

2.2.2 From trajectory to training data

There is a distinction to be made between the configurational space of the simulation, and the data space over which the learned model is optimized. Indeed, a preprocessing step is usually necessary to obtain a usable dataset from the sampled molecular trajectory. A notable example is the elimination of rotational and translational invariances through centering and structural alignment of the configurations to a reference structure, or by using internal coordinates (distances, angles, etc). Another example is using only a subset of the coordinates to reduce the data dimensionality, for example by taking out solvent molecules and hydrogen atoms, whose motions are often considered irrelevant. For the remainder of the paper, we denote by q the configurations of the system, and by x the corresponding post-processed data points. We however allow for an abuse of notation by keeping the same notation for various objects whose argument is either q or x, depending on the context. For

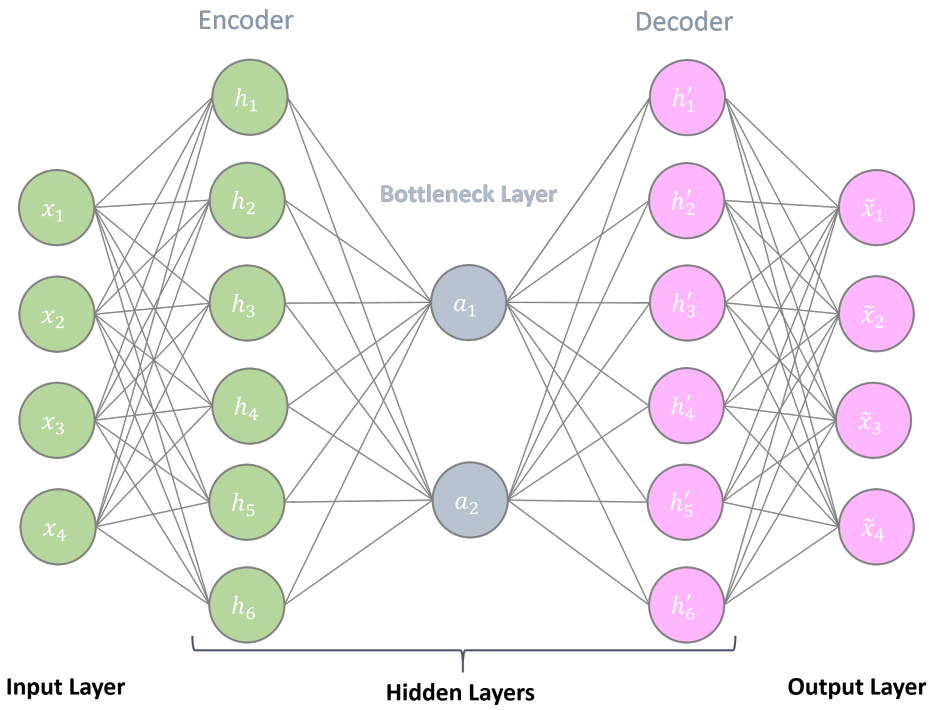


Figure 2.1: Example of an autoencoder topology. Here, the dimension of the data is $D = 4$ and there are $2L = 4$ layers: 3 hidden layers (including the bottleneck) and the output layer, of respective dimensions $d_1 = 6$, $d_2 = 2$, $d_3 = 6$ and $d_4 = 4$. The parameters can be represented by four matrices W_1, W_2 (for the encoder) and W_3, W_4 (for the decoder), such that: $\mathbf{a} = (a_1, a_2) = f_{\text{enc}}(\mathbf{x}) = g_2(b_2 + W_2 g_1(b_1 + W_1\mathbf{x}))$ and $\tilde{\mathbf{x}} = f_{\text{dec}}(\mathbf{a}) = g_4(b_4 + W_4 g_3(b_3 + W_3\mathbf{a}))$ where $g_1, \ldots, g_4$ are activation functions, and $b_1, \ldots, b_4$ are biases (not shown in the topology).

example, the probability measures $\mu(dq)$ and $\mu(dx)$ are denoted by the same symbol, even though $\mu(dx)$ is actually the image of $\mu(dq)$ by the application $q \mapsto x$.

2.2.3 Learning from unbiased samples

Consider a probability distribution μ on $\mathcal{X} \subset \mathbb{R}^D$. Typically in MD, μ is the Boltzmann-Gibbs distribution and $\mathcal{X}$ is the configurational space of the system under study. We seek to encode configurations of $\mathcal{X}$ sampled from μ in a smaller dimensional space $\mathcal{A} \subset \mathbb{R}^d$, where $d < D$, using an autoencoder (AE). Theoretically, the optimal parameters $\mathbf{p}_\mu$ minimize the expected loss

$$\mathcal{L}(\mu, \mathbf{p}) = \mathbb{E}_\mu(\|X - f_{\mathbf{p}}(X)\|^2) = \int_{\mathcal{X}} \|x - f_{\mathbf{p}}(x)\|^2 \, \mu(dx) \, , \tag{2.1}$$

where the subscript μ in $\mathbb{E}_\mu$ indicates that X is a random variable distributed according to μ. We denote by $\mathbf{p}_\mu$ a solution to the minimization of $\mathcal{L}(\mu, \mathbf{p})$, provided it exists (which

is always assumed here):

$$\mathbf{p}_\mu \in \operatorname*{argmin}_{\mathbf{p}\in\mathbb{R}^K} \mathcal{L}(\mu, \mathbf{p}). \tag{2.2}$$

Note that $\mathbf{p}_\mu$ is a priori not unique, and depends of course on the distribution μ.

Empirical distribution

In practice, the optimization problem (2.2) is not easily solved, in particular because $\mathcal{L}$ is unknown and must be approximated. For this, the distribution μ is replaced by an empirical distribution $\hat{\mu}$ corresponding to an available dataset of $N \in \mathbb{N}$ points $x^1, \ldots, x^N$ drawn from the distribution μ. In MD, these datapoints are typically the configurations sampled during a simulation of the system, represented e.g by atomic positions or internal coordinates. The autoencoder is thus optimized in practice using the empirical distribution:

$$\hat{\mu} = \frac{1}{N}\sum_{i=1}^{N} \delta_{x^i} \ .$$

The AE parameters are thus optimized in order to minimize the empirical loss:

$$\mathcal{L}(\hat{\mu}, \mathbf{p}) = \frac{1}{N}\sum_{i=1}^{N} \|x^i - f_{\mathbf{p}}(x^i)\|^2 \ . \tag{2.3}$$

Using the law of large numbers (ergodic theorem), it holds that:

$$\mathcal{L}(\hat{\mu}, \mathbf{p}) \xrightarrow[N\to\infty]{} \mathcal{L}(\mu, \mathbf{p}) \qquad \text{almost surely,}$$

which motivates minimizing with respect to $\mathbf{p}$ the empirical loss $\mathcal{L}(\hat{\mu}, \mathbf{p})$ instead of $\mathcal{L}(\mu, \mathbf{p})$. The optimization problem thus becomes:

$$\text{Find} \quad \mathbf{p}_{\hat{\mu}} \in \operatorname*{argmin}_{\mathbf{p}\in\mathbb{R}^K} \mathcal{L}(\hat{\mu}, \mathbf{p}). \tag{2.4}$$

2.2.4 Learning from biased samples

To accelerate the exploration of the phase space of the system under study, MD simulations can be biased to target another distribution than the Boltzmann-Gibbs reference measure. The resulting dataset is thus not drawn from the original distribution of interest μ, but from a biased distribution $\widetilde{\mu}$. Since we want to optimize the loss (2.1), we need to reweight the configurations x sampled from $\widetilde{\mu}$ by a factor

$$w(x) = \frac{\mu(x)}{\widetilde{\mu}(x)}.$$

Note that to ensure that $w(x)$ is finite, we always assume that μ is absolutely continuous with respect to $\widetilde{\mu}$ (i.e. $\mu(A) = 0$ for all measurable sets $A \subset \mathcal{X}$ such that $\widetilde{\mu}(A) = 0$).

The reweighting corresponds to considering the loss function

$$\mathcal{L}(\mu, \mathbf{p}) = \mathcal{L}\left(\frac{\mu}{\widetilde{\mu}}\widetilde{\mu}, \mathbf{p}\right) = \int_{\mathcal{X}} \|x - f_{\mathbf{p}}(x)\|^2 \, w(x)\widetilde{\mu}(dx). \tag{2.5}$$

Reweighting ensures that the same optimization problem as (2.2) is solved, even when the data points are not distributed according to μ.

To approximate the expectation (2.5) with respect to $\widetilde{\mu}$, we again rely on an empirical weighted distribution:

$$\widehat{\mu}_{\text{wght}} = \sum_{i=1}^{N} \widehat{w}_i \delta_{x^i} \qquad \widehat{w}_i = \frac{\mu(x^i)/\widetilde{\mu}(x^i)}{\sum_{j=1}^{N} \mu(x^j)/\widetilde{\mu}(x^j)}. \tag{2.6}$$

The new discrete loss is thus a weighted average loss over the $\widetilde{\mu}$ sampled data:

$$\mathcal{L}(\widehat{\mu}_{\text{wght}}, \mathbf{p}) = \sum_{i=1}^{N} \widehat{w}_i \|x^i - f_{\mathbf{p}}(x^i)\|^2 \ , \tag{2.7}$$

which converges by the ergodic theorem almost surely to $\mathcal{L}(\mu, \mathbf{p})$ when $N \to \infty$.

Note that computing $(\widehat{w}_i)_{1 \le i \le N}$ only requires the knowledge of μ and $\widetilde{\mu}$ up to a multiplicative constant.

2.2.5 Learning from free energy biased simulations: 2D toy example

This section provides a 2D example for free energy biasing. This system is then used to illustrate the necessity of reweighting when training models over biased simulations. We use throughout the section some common definitions of free energy and Boltzmann–Gibbs density measure,recalled in Supp. Mat. Section 2.7.1.A for completeness.

Let us introduce the following three well potential, previously considered in other works [159]:

$$\begin{aligned} V(x_1, x_2) = {} & 3\mathrm{e}^{-x_1^2}\left(\mathrm{e}^{-(x_2-1/3)^2} - \mathrm{e}^{-(x_2-5/3)^2}\right) - 5\mathrm{e}^{-x_2^2}\left(\mathrm{e}^{-(x_1-1)^2} + \mathrm{e}^{-(x_1+1)^2}\right) \\ & + 0.2x_1^4 + 0.2(x_2 - 1/3)^4 \ . \end{aligned} \tag{2.8}$$

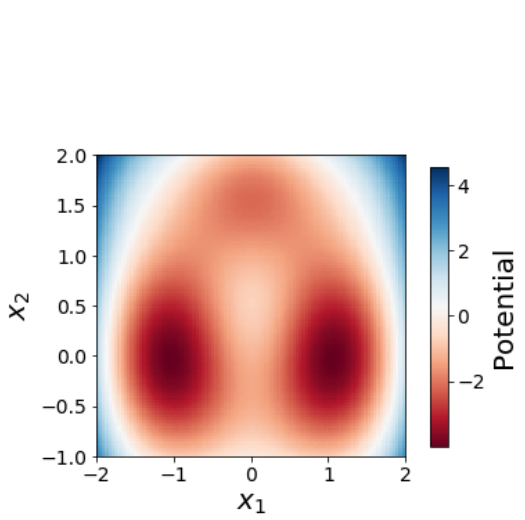
In this example, the configurations q are represented by the 2 dimensional coordinates of the particle: $q = (x_1, x_2)$. The potential V has two deep wells centered at $q_{\text{L}} = (-1.113, -0.03685)$, $q_{\text{R}} = (1.113, -0.03685)$, and a shallow well around $q_{\text{C}} = (0, 1.7566)$. We consider the overdamped Langevin dynamics:

$$dq_t = -\nabla V(q_t)\ dt + \sqrt{\frac{2}{\beta}}\ dB_t, \tag{2.9}$$

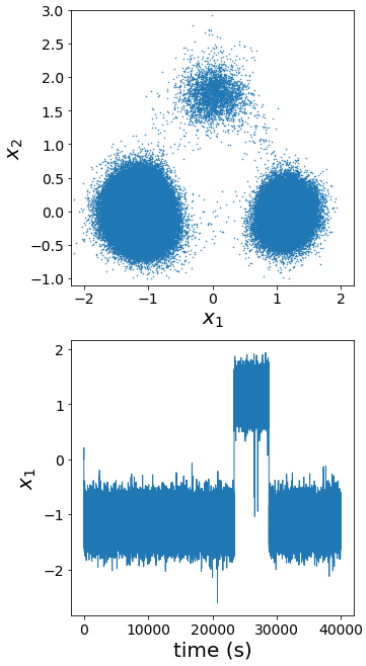
where B_t is a 2-dimensional Brownian motion. The dynamics are discretized using the Euler–Maruyama scheme:

$$q^{j+1} = q^j - \nabla V(q^j)\Delta t + \sqrt{\frac{2\Delta t}{\beta}} G^j \ , \tag{2.10}$$

where $\{G^j\}_{j \ge 0}$ is a sequence of independent standard normal random vectors and $q^j \approx q_{j\Delta t}$. We simulate a sample trajectory of this system, with inverse temperature $\beta = 4$ and time-step $\Delta t = 10^{-3}$, starting from initial condition $q^0 = (-1, 0)$. Figure 2.2 gives a scatter plot of this trajectory as well as the time evolution of the coordinate x_1 .



(a)



(b)

Figure 2.2: Simulations with the 2D three well potential (2.8) at $\beta = 4$. Left: Potential function, two deep wells and one shallow well can be seen. Right: Overdamped Langevin trajectory. Top. Scatter plot of the sampled points. Bottom: Time evolution of the coordinate x_1 through the example trajectory. Metastability is observed.

The metastability of the dynamics is quite well characterized by the direction x_1, whereas x_2 is not enough to distinguish between the two deep wells. In the sequel, we consider $\xi_1(q) = x_1$ and $\xi_2(q) = x_2$ as two different choices of CV.

The free energies corresponding to the CVs ξ_1 and ξ_2 are respectively:

$$F_1(x_1) = -\beta^{-1} \ln \left(\int_{\mathbb{R}} \mathrm{e}^{-\beta V(x_1,x_2)} dx_2 \right), \quad F_2(x_2) = -\beta^{-1} \ln \left(\int_{\mathbb{R}} \mathrm{e}^{-\beta V(x_1,x_2)} dx_1 \right). \tag{2.11}$$

The quantities $F_1(x_1)$ and $F_2(x_2)$ can be easily approximated in this low dimensional example by a numerical quadrature. We can therefore easily sample from the three following probability measures: the unbiased Boltzmann-Gibbs probability distribution ν associated with the potential V, and the biased Boltzmann-Gibbs probability distributions ν_{F_i} associated with the biased potentials $V - F_i \circ \xi_i$ for $i = 1, 2$.

We now perform experiments to compare the results obtained from biased trajectories using different collective variables, with those from an unbiased trajectory. For this, we first simulate three trajectories. All three are long enough to sample the three potential wells of our system. The first trajectory is unbiased, following (2.10) with a timestep $\Delta t = 10^{-3}$ for $N = 4 \times 10^7$ steps. Every 40th configuration is kept as a datapoint. The second trajectory is free energy biased using $\xi_1(x_1, x_2) = x_1$ as CV, i.e., it follows (2.10) with potential $V - F_1 \circ \xi_1$

instead of V, and a timestep $\Delta t = 10^{-3}$ for $T = 2\times10^6$ steps. Configurations are saved every 2 timesteps. The third trajectory is free energy biased using $\xi_2(x_1, x_2) = x_2$ as a collective variable, and using the same simulation settings as the second trajectory. Note that the second and third trajectories are shorter: the duration of these trajectories were chosen so that they visit the full configurational space. These three trajectories serve as our training datasets. The datapoints are thus sampled respectively from the unbiased Boltzmann–Gibbs measure $\mu = \nu$, the free energy biased Boltzmann–Gibbs measure $\widetilde{\mu}_1 = \nu_{F_1}$, and the free energy biased Boltzmann–Gibbs measure $\widetilde{\mu}_2 = \nu_{F_2}$.

We next build five small autoencoders with the following topology: $D = 2$ (input data dimensionality), $2L = 2$ layers containing respectively 1 and 2 nodes, and $g_1(z) = \tanh(z)$ and $g_2(z) = z$, the identity function. The number of parameters is thus $K = 7$ (including 3 bias parameters). We use tanh as the bottleneck activation function in order to obtain a bounded learned collective variable with known bounds $[-1, 1]$. All five autoencoders are initialized with the same random network parameters. We then separately train these autoencoders for a maximum of 200 epochs each as follows:

- The first AE is trained on the unbiased trajectory;
- The second AE is trained on the ξ_1-biased trajectory;
- The third AE is trained on the ξ_1-biased trajectory, but with reweighting, meaning that each data point $x^i = (x_1^i, x_2^i)$ contributes to the learning loss with

$$\widehat{w}_i = N\frac{\mu(x^i)/\widetilde{\mu}_1(x^i)}{\sum_{j=1}^{N}\mu(x^j)/\widetilde{\mu}_1(x^j)} = N\frac{\mathrm{e}^{-\beta F_1(x^i)}}{\sum_{j=1}^{N}\mathrm{e}^{-\beta F_1(x^j)}}\ .$$

- The fourth AE is trained on the ξ_2-biased trajectory;
- The fifth AE is trained on the ξ_2-biased trajectory, but reweighted using

$$\widehat{w}_i = N\frac{\mu(x^i)/\widetilde{\mu}_2(x^i)}{\sum_{j=1}^{N}\mu(x^j)/\widetilde{\mu}_2(x^j)} = N\frac{\mathrm{e}^{-\beta F_2(x^i)}}{\sum_{j=1}^{N}\mathrm{e}^{-\beta F_2(x^j)}}\ .$$

As discussed in Supp. Mat. Section 2.7.1.D, the multiplication by the number of samples N allows to comply with the default normalization of the weights of the ML package we use. A subset of the data is kept as validation set (here, 10%), and early stopping is applied when the validation loss no longer improves for 20 consecutive epochs.

We compare the obtained encoders using heat maps. The values of the encoded bottleneck functions are computed over the (discretized) space $(x_1, x_2) \in [-2, 2] \times [-1, 2.5]$. The results are given in Figure 2.3. Note that the aim here is to compare encodings obtained from the different models, and possibly to have insight on how these encodings depend on the coordinates x_1 and x_2. Let us emphasize in particular that it is for example possible to obtain a function of x_1 instead of x_1 itself as encoder variable. For this reason, and to simplify the comparison between the obtained CVs, they are renormalized to have a range that is exactly between 0 and 1:

$$\xi_{\mathrm{AE}}^{\mathrm{norm}}(x) = \frac{\xi_{\mathrm{AE}}(x) - \xi_{\mathrm{AE}}^{\mathrm{min}}}{\xi_{\mathrm{AE}}^{\mathrm{max}} - \xi_{\mathrm{AE}}^{\mathrm{min}}}, \tag{2.12}$$

where $\xi_{\mathrm{AE}}^{\min}$ and $\xi_{\mathrm{AE}}^{\max}$ are respectively the minimum and maximum values taken by the encoder function ξ_{AE} over the 2-dimensional space $[-2,2]\times[-1,2.5]$. We draw two important conclusions upon interpreting the obtained results:

- First, the encoding learned through unbiased training on the reference data obtained from a long unbiased trajectory (top panel) is a function of x_1, which we recall is considered a good CV as it distinguishes the three potential wells. The direction x_1 therefore corresponds to the target unbiased representation that we want to find using the remaining models (in particular with the reweighted learning). The encodings obtained from the reweighted training over the two biased trajectories (middle and bottom right panels) also represent bijective functions of x_1. The encodings obtained through biased unweighted learning, however, encode different variables (middle and bottom left panels). This is therefore an example where reweighting proves necessary for obtaining results that are consistent with unbiased learning.

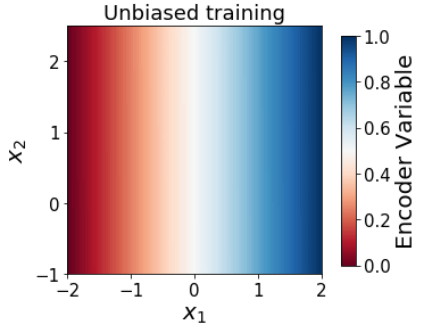
- It is interesting to note the difference between the CV obtained from biased learning over the ξ_1-trajectory (middle left panel) and the one obtained from the ξ_2-trajectory (bottom left panel). It is quite clear that the latter is closer to a monotonic function of the target learning result, x_1, than the former. These results can be intuitively explained as follows: Free energy biasing changes the distribution along the CV (here ξ_1 or ξ_2) to a uniform law, making the corresponding direction no longer relevant. This prompts the model to learn other directions that are still relevant in the biased simulation. In the case of the trajectory obtained by biasing with $F_1 \circ \xi_1$, the x_1 direction is no longer a relevant feature of the data space and the encoder does not recover it. On the other hand, biasing using $F_2 \circ \xi_2$ does not completely annihilate the relevance of the variable x_1 in the sampled data. This explains why the CV learned from biasing with ξ_2 is closer to x_1.

This last point is very important as it means that if an iterative learning run is performed without reweighting, then whenever a good CV is learned at iteration n, this CV may be cancelled out in the next iteration, making convergence difficult to achieve.

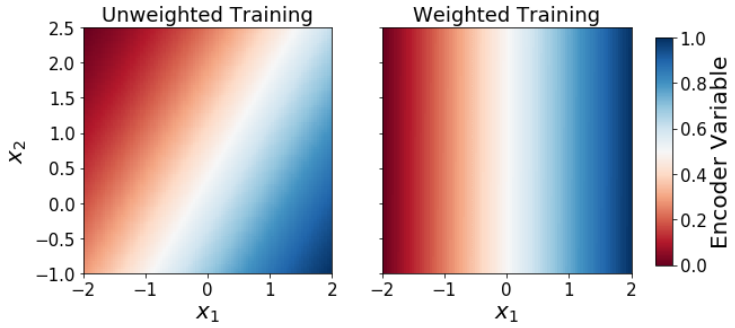
2.3 Iterative reweighted learning of CVs with autoencoders

In this section, we introduce our algorithm for the iterative learning of a CV from biased trajectory data. As mentioned in the introduction, our algorithm is in part inspired by the work done in Refs. 93 and 94, where MESA, an iterative method for learning CVs on the fly while performing enhanced sampling was devised. MESA alternates between learning CVs with autoencoders and using those CVs for extrapolation with Umbrella Sampling, until convergence.

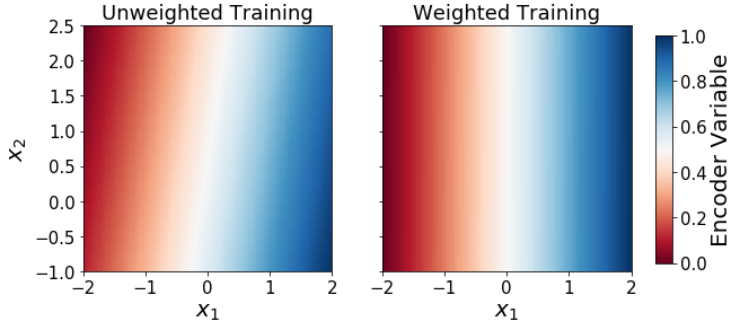
The major difference between our algorithm and MESA is that we use reweighting of the configurations sampled from free energy biasing to target the unbiased loss corresponding to the Boltzmann-Gibbs distribution. Indeed, as observed in the experiments of Section 2.2.5, training on biased data does not yield the same results (same encoded CVs) as with unbiased data. Additionally, we saw that performing free energy biasing using a certain choice of CV may make this direction irrelevant within the sampled data, which prevents the model from learning it. This means first that a naive iterative learning algorithm without reweighting



(a) No biasing



(b) Biasing with CV $\xi_1 = x_1$



(c) Biasing with CV $\xi_2 = x_2$

Figure 2.3: Heatmaps of the values of the encoder variables over the discretized 2D space $[-2, 2] \times [-1, 2.5]$: (a) Unbiased learning outputs a CV close to $\xi_1(x) = x_1$. (b) and (c) Left: Training on biased simulations leads to learned CVs that depend on the free energy biasing. Right: Reweighting enables learning a CV that is independent of the sampling procedure.

is not theoretically guaranteed to converge, as each new CV (learned with a new AE) is different from the previous one; and second, that this can result in iterations of the algorithm where the learned CV is not necessarily a relevant choice for the next free energy biasing. Reweighting the loss is expected to solve this issue in the iterative method the same way it did in the experiments shown in Section 2.2.5. At each step, we train the autoencoder on the

reweighted loss, using the reweighting introduced in (2.7), so as to remove bias from the data sampled in the free energy biasing simulation. Another major difference in our approach compared to MESA is that we rely on adaptive techniques to sample configurations and compute the free energy for reweighting them. Let us however emphasize that our approach could also be used with other methods to compute free energy profiles.

We first give in Section 2.3.1 a detailed description of our algorithm. We then introduce in Section 2.3.2 adaptive biasing methods in general and the adaptive biasing force method in particular, as it constitutes our choice of biasing procedure. Then, some additions and refinements of the algorithm are discussed in Section 2.3.3 to transfer information between consecutive iterations of the algorithm. Finally, Section 2.3.4 provides a first application of the algorithm to the 2-dimensional toy example introduced in Section 2.2.5.

2.3.1 Iterative algorithm for CV learning: General description

This section provides a description of the steps of our iterative algorithm, which we call FEBILAE, for "Free Energy Biasing and Iterative Learning with AutoEncoders". The algorithm is also summarized in the pseudo-code of Algorithm 1.

Initialization

The first step of the algorithm is to produce an initial unbiased trajectory to start from, traj_0. This requires providing the algorithm with a simulation setup S (dynamics, physical conditions, etc), and an initial configuration q^0. The trajectory traj_0 is preprocessed as necessary (see Section 2.2.2) to obtain an initial training dataset $(x^1, \dots, x^N)$ where N is the number of samples, provided as an input to the method. As traj_0 is unbiased, the sample weights associated to this first training are uniform: $\widehat{w}_j = 1$ for all $j \in \{1, \dots, N\}$. A first autoencoder, AE_0 is then initialized (with a given topology, hyper-parameters and random parameters A_{init}) and trained on this dataset. A first collective variable, ξ_0, is thus computed from AE_0.

Iterative procedure and stopping rule

At each iteration $i \geq 1$ of the algorithm, the following steps are performed:

- **New trajectory.** The previous CV ξ_{i-1} is used to perform adaptive free energy biasing under a given simulation method and setup S_{AB}, and starting from the same initial configuration q^0 as the one used in the initialization step, or from a new initial configuration (e.g. the last sampled configuration of the previous trajectory). The biased trajectory traj_i is saved, along with the estimation of the free energy F_i corresponding to the CV ξ_{i-1}.

- **Preprocessing and sample weight computation.** The trajectory is preprocessed to obtain the new training dataset $(x^1, \dots, x^N)$, and the weight of each sampled datapoint is computed as defined in (2.6) and (2.7) (with the weights multiplied by N in order to comply with the default setting of the ML package used for implementation , see Supp. Mat. Section 2.7.1.D).

- **Autoencoder training and new CV.** A new autoencoder, with the same topology and initial parameters as in the previous iterations (A_{init}), is trained on the new dataset traj_i to optimize the reweighted loss (2.7). As mentioned in Section 2.2.1, the

autoencoder has the advantage of outputting a mapping ξ_i from the input configuration to the collective variable, as well as its gradient, which is also required for most CV biasing methods. Additionally, the interval(s) of the CV, over which biasing is applied, must be determined. These intervals can be estimated using the extreme values which the CV takes over the training data traj_i.

- **Stopping rule.** The algorithm stops when one of the two following conditions is met: either the algorithm has reached a given maximum number of iterations $I_{\max}$, or the CV has converged. The last step of each iteration thus requires to assess CV convergence. In this paper, we consider that the learned CV has converged when the new CV ξ_i can be mapped (using e.g. a simple regression model) to the previous CV ξ_{i-1}, with a high enough precision score, higher than a given threshold $s_{\min}$. We use linear regression as a mapping model to check whether $\xi_i \approx \Phi(\xi_{i-1})$. We refer to Supp. Mat. Section 2.7.2 for a more detailed discussion on how to compute the regression score. The value of $s_{\min}$ depends on the system under study, the complexity and dimension of the computed CVs, and the regression model used to compute the regression score. It is determined separately for each of the systems studied in this paper, using a bootstrap procedure made precise in Supp. Mat. Section 2.7.2. When the algorithm stops, the last learned CV is kept as the final CV. Its free energy is estimated and used to fully sample the configurational space of the system.

Remark 4 (CV dimensionality)**.** *When the optimal dimensionality of the CVs is unknown, one can use the following method proposed in Ref.93 to determine it: several autoencoders $\Phi_1, \dots, \Phi_M$, with different values for the dimensionality d of the bottleneck layer ($d \in \{1, \dots, M\}$), are all trained at each iteration. The optimal value of d is then determined by plotting the FVE (fraction of variance explained) $1 - \frac{\sigma_{\text{res}}(d)}{\sigma_{\text{tot}}}$ as a function of d, where*

$$\sigma_{\text{res}}(d) = \sum_{i=1}^{N} \|x^i - \Phi_d(x^i)\|^2,$$

is the residual sum of squares and

$$\sigma_{\text{tot}} = \sum_{i=1}^{N} \|x^i - \bar{x}\|^2, \quad \bar{x} = \frac{1}{N} \sum_{i=1}^{N} x^i,$$

is the total sum of squares. The optimal value of d corresponds to a plateau or "knee" in the FVE curve, meaning that no considerable improvement in the optimized loss is obtained by adding another dimension to the bottleneck space. This is similar to what is done in PCA, where spectral gaps in the distribution of eigenvalues are used to determine the optimal number of principal components to keep.

Algorithm 1: FEBILAE

Input: Initial condition q^0, autoencoder topology and initialization parameters A_{init}, number of samples N, simulation procedure S and adaptive biasing procedure S_{AB}, maximum number of iterations $I_{\max}$, minimum convergence score $s_{\min}$.

Output: CV ξ_{final} and corresponding PMF F_{final}.

Sample $\text{traj}_0 \leftarrow S(q^0, N)$. Initialize autoencoder $\text{AE}_0 \leftarrow A_{\text{init}}$.
Train AE_0 on traj_0 with weights $(\widehat{w}_0, \dots, \widehat{w}_N) = (1, \dots 1)$.
Extract the encoder function $\xi_0 : x \mapsto \xi_0(x)$.
Set $i \leftarrow 0$, $s \leftarrow 0$.
while $i < I_{\max}$ & $s < s_{\min}$ **do**
  Set $i \leftarrow i + 1$.
  Sample $(\text{traj}_i,\, F_i) \leftarrow S_{\text{AB}}(q^0, N, \xi_{i-1})$.
  Compute weights $(\widehat{w}_j)_{1 \le j \le N}$ as $\widehat{w}_j = N \dfrac{\mathrm{e}^{-\beta F_i(\xi_{i-1}(x^j))}}{\sum_{n=1}^{N} \mathrm{e}^{-\beta F_i(\xi_{i-1}(x^n))}}$.
  Initialize autoencoder $\text{AE}_i \leftarrow A_{\text{init}}$.
  Train AE_i on traj_i with sample weights $(\widehat{w}_j)_{1 \le j \le N}$.
  Extract the encoder function $\xi_i : x \mapsto \xi_i(x)$.
  Set $s \leftarrow \text{regscore}(\xi_{i-1}, \xi_i)$.
Set $\xi_{\text{final}} \leftarrow \xi_i$.
Sample $\text{traj}_{\text{final}}$, $F_{\text{final}} \leftarrow S_{\text{AB}}(q^0, N_{\text{final}}, \xi_{\text{final}})$ with N_{final} large enough to ensure PMF convergence.

2.3.2 Sampling with the (extended) Adaptive Biasing Force method

With the exception of low dimensional systems ($D \le 3$) such as the example used in Section 2.2.5, the free energy of a system associated with a given CV cannot be easily estimated. Adaptive sampling algorithms replace the actual free energy F by an estimated function F_t in the biased dynamics at time t, meaning that the potential becomes $V - F_t \circ \xi$. The estimate F_t is updated on-the-fly so that it converges to F as the molecular dynamics simulation proceeds. The two categories of adaptive biasing techniques are [160] Adaptive Biasing Potential (ABP) methods, where the free energy F_t is estimated and its gradient, the so-called mean force, is then used in the dynamics; and Adaptive Biasing Force (ABF) methods where the mean force is estimated directly, and the free energy is subsequently obtained by numerical integration through a Helmholtz projection [66].

In this work, we choose to work with ABF, and more precisely extended system ABF (eABF) [67] for free energy biasing. We call this version of the algorithm AE-ABF. Section 2.3.2.A gives a brief description of ABF. We then recall its main limitation, namely that it requires the computation of the second order derivatives of the used CVs. This motivates using eABF, which is described in Section 2.3.2.B.

2.3.2.A Adaptive biasing force

The adaptive biasing force method estimates the mean force associated with the collective variable ξ. For ease of notation, we assume in this section and the following one that ξ is one dimensional, with values in $\mathcal{A} \subseteq \mathbb{R}$, but the generalization to a higher dimensional CV

is straightforward. By differentiating both sides of Equation (2.23), one obtains:

$$F'(z) = -\frac{1}{\beta}\int_{\Sigma(z)} f(q)\frac{\mathrm{e}^{-\beta V(q)}\delta_{\xi(q)-z}(dq)}{\mathrm{e}^{-\beta F(z)}} \ , \tag{2.13}$$

where f is called the local mean force:

$$f = \nabla V \cdot \frac{\nabla\xi}{|\nabla\xi|^2} - \frac{1}{\beta}\,\mathrm{div}\left(\frac{\nabla\xi}{|\nabla\xi|^2}\right) \ . \tag{2.14}$$

Note that the equation above contains second order derivatives of ξ. Equation (2.13) shows that the derivative of the free energy $F'(z)$ is related to the conditional average of the local mean force as

$$F'(z) = \mathbb{E}_\nu\left(f(q)|\xi(q) = z\right) \ , \tag{2.15}$$

with ν defined in (2.22). This suggests, for instance, that the biasing force at time t can be estimated as follows when considering a single long trajectory:

$$\Gamma_t(z) = \frac{\int_0^t f(q_s)\ \delta^\varepsilon\left(\xi(q_s) - z\right) ds}{\int_0^t \delta^\varepsilon\left(\xi(q_s) - z\right) ds} \ ,$$

where $\varepsilon > 0$ and δ^ε is an approximation of the Dirac mass.

The estimated mean force is then used to bias the dynamics with the force $\Gamma_t(\xi(q_t))$. For example, the dynamics of ABF used in conjunction with Langevin dynamics is:

$$\begin{cases} dq_t = M^{-1}p_t\ dt, \\ dp_t = -\nabla V(q_t)dt + \Gamma_t(\xi(q_t))\nabla\xi(q_t)dt - \gamma p_t dt + \sqrt{2\gamma M\beta^{-1}}dB_t, \end{cases}$$

where M is the mass matrix and $\gamma > 0$ is the friction coefficient.

2.3.2.B Extended system Adaptive biasing force

Equation (2.14) shows that regular ABF requires the knowledge of second order derivatives of the CV ξ to compute the local mean force f. The analytical expression of this quantity is quite cumbersome for most choices of CVs, especially when ξ is vector valued. In particular, as our CVs are encoder based mappings and can involve complex non-linear activation functions, extracting the second order derivatives is not a viable option.

To overcome the limitations in computing the second term of (2.14), a method coined extended system ABF (eABF) [67] was devised. A fictitious degree of freedom λ is added to the configurational space and the new potential is:

$$V^{\mathrm{ext}}(q,\lambda) = V(q) + \frac{\kappa}{2}|\xi(q) - \lambda|^2 \ , \tag{2.16}$$

where κ is a force constant. The collective variable $\xi^{\mathrm{ext}}(x,\lambda) = \lambda$ is used instead of the original CV ξ. Using Equation (2.23), the resulting free energy F^{ext} is a convolution of a Gaussian kernel and the free energy associated with ξ:

$$\mathrm{e}^{-\beta F^{\mathrm{ext}}(\lambda)} = \int_{\mathcal{A}} \chi_\kappa(\lambda - z)\mathrm{e}^{-\beta F(z)}dz \ ,$$

where χ_κ is a Gaussian kernel with variance $1/(\kappa\beta)$. When $\kappa \to \infty$, $F^{\text{ext}}(\lambda)$ converges to $F(\lambda)$. Thus, a simple estimator of the real free energy F is to use $F = F^{\text{ext}}$ directly. This naive estimator is biased of course, given that in practice $\kappa < \infty$. The new extended mean force does not depend on second order derivatives of the CV ξ: Only the gradient of ξ is needed for computing the gradient of V^{ext}. ABF can therefore easily be applied to the new extended system.

Denoting by ρ the momentum of λ, and by M^{ext} the extended mass matrix (which includes m_λ, the fictitious mass of λ), the Langevin dynamics of the eABF trajectory are:

$$\begin{cases} dq_t^{\text{ext}} = (M^{\text{ext}})^{-1} p_t^{\text{ext}}\ dt, \\ dp_t^{\text{ext}} = \left(-\nabla V^{\text{ext}}(q_t, \lambda_t) + \Gamma_t^{\text{ext}}(\lambda_t)\mathbf{u}\right) dt - \gamma p_t^{\text{ext}} dt + \sqrt{2\gamma\beta^{-1}M^{\text{ext}}} dB_t, \end{cases} \tag{2.17}$$

where $q_t^{\text{ext}} = (q_t, \lambda_t)$, $p_t^{\text{ext}} = (p_t, \rho_t)$, $\mathbf{u}^T = (0, \dots, 0, 1)$ and Γ_t^{ext} is the estimate at time t of the mean force associated with λ in the extended system:

$$\Gamma_t^{\text{ext}}(\lambda) = \frac{\int_0^t f^{\text{ext}}(q_s)\ \delta^\varepsilon\left(\lambda_s - \lambda\right) ds}{\int_0^t \delta^\varepsilon\left(\lambda_s - \lambda\right) ds}\ , \qquad f^{\text{ext}}(q) = \frac{\partial V}{\partial \lambda}(q) = \kappa(\lambda - \xi(q)).$$

In practice, the above equation is discretized in time using a timestep Δt. Denoting by $q^{\text{ext},j}, p^{\text{ext},j}$ the approximations of $q_{j\Delta t}^{\text{ext}}, p_{j\Delta t}^{\text{ext}}$ at iteration j, the estimated mean force Γ_t^{ext} is considered to be constant in discrete bins of $\xi^{\text{ext}} = \lambda$ centered around points $z_1 \dots, z_k$ (uniformly spaced, for simplicity of presentation): for all $\ell \in \{1 \dots, k\}$, at time $j\Delta t$,

$$\forall z \in [z_\ell - \varepsilon, z_\ell + \varepsilon[, \qquad \Gamma_j^{\text{ext}}(z) = \frac{\sum_{i=0}^j f^{\text{ext}}(q^{\text{ext},i}) \mathbb{1}_{\{z_\ell - \varepsilon \le \lambda^i < z_\ell + \varepsilon\}}}{\sum_{i=0}^j \mathbb{1}_{\{z_\ell - \varepsilon \le \lambda^i < z_\ell + \varepsilon\}}}, \tag{2.18}$$

where $2\varepsilon = \frac{z_k - z_1}{k-1}$.

2.3.3 Transferring information between algorithmic iterations

This section discusses two ways of using part of the information learned in previous iterations to possibly improve or accelerate the learning or sampling in the next iteration. Following the notation in Algorithm 1, the iteration index is denoted by i.

2.3.3.A Using previous trajectories

At each iteration, the training dataset can be a combination of a fixed number $n_T \ge 1$ of previously sampled trajectories. Standard FEBILAE corresponds to $n_T = 1$. Using more than one trajectories provides the autoencoder with a larger, and therefore possibly more complete dataset. This refinement of the algorithm is very straightforward, and is thus directly included in the implementation used to obtain the results presented in Section 2.5.1 with $n_T = 2$. The results of this simple addition on the 2-dimensional example as compared to the basic algorithm are presented in Supp. Mat. Section 2.7.3.A.

2.3.3.B Free energy initialization

The algorithm AE-ABF as presented above starts each new eABF with a free energy profile initialized to 0. However, as the CV progressively converges to an optimal value throughout

the algorithm, the successive CVs from one iteration to the other are somewhat similar. Therefore, the free energy profile at the end of the previous iteration of eABF could be used to suggest a better initialization of the free energy or mean force for the new eABF run.

In order to make this discussion more precise, let us assume here again that the CV is one dimensional for the simplicity of exposition, although our approach can be extended to CVs of dimension larger or equal to 2. At iteration i, we call $\xi_i : \mathcal{X} \to [a_i, b_i]$ the CV used in (2.16) to perform the eABF simulation for this round, which we call eABF_i. The corresponding estimated free energy and mean force are respectively denoted by F_i and F_i'. Before starting eABF_{i+1}, we want to determine a function $\widetilde{F}_{i+1}$ (respectively $\widetilde{F}'_{i+1}$) as a good initialization of the free energy (respectively the mean force) using the previous free energy F_i and the CVs ξ_i and ξ_{i+1}. We first assume that the CVs have converged, namely that:

$$\xi_{i+1} = \Phi \circ \xi_i,$$

for some strictly monotonic function $\Phi \in C^1([a_i, b_i])$. We can then easily calculate F_{i+1} from F_i under this assumption. Indeed, for $Z = \Phi(z)$,

$$\mathrm{e}^{-\beta F_{i+1}(Z)} = \int_{\Sigma_{i+1}(\Phi(z))} \mathrm{e}^{-\beta V(q)} \delta_{\xi_{i+1}(q)-\Phi(z)}(dq),$$

where

$$\Sigma_{i+1}(\Phi(z)) = \{q \in \mathbb{R}^D | \xi_{i+1}(q) = \Phi(z)\} = \{q \in \mathbb{R}^D | \Phi(\xi_i(q)) = \Phi(z)\} = \Sigma_i(z).$$

By the co-area formula (Equation (3.14) in Ref. 22)

$$\delta_{\xi_{i+1}(q)-\Phi(z)}(dq) = \frac{|\nabla \xi_i(q)|}{|\nabla \xi_{i+1}(q)|} \delta_{\xi_i(q)-z}(dq) = \frac{1}{|\Phi'(z)|} \delta_{\xi_i(q)-z}(dq).$$

Therefore,

$$\mathrm{e}^{-\beta F_{i+1}(\Phi(z))} = \int_{\Sigma_i(z)} \mathrm{e}^{-\beta V(q)} \frac{1}{|\Phi'(z)|} \delta_{\xi_i(q)-z}(dq) = \frac{1}{|\Phi'(z)|} \mathrm{e}^{-\beta F_i(z)}. \tag{2.19}$$

The above equation provides an expression of F_{i+1} from F_i when Φ is known. In practice, we approximate Φ as an affine function using a linear regression model optimized using the datapoints sampled from the trajectory of ABF_i (and possibly also from previous trajectories as suggested in Section 2.3.3.A). This provides an approximate mapping:

$$\xi_{i+1}(x) = \omega_1 \xi_i(x) + \omega_2,$$

for which the following equality holds:

$$\forall Z \in [a_{i+1}, b_{i+1}], \qquad \mathrm{e}^{-\beta F_{i+1}(Z)} = \frac{1}{|\omega_1|} \mathrm{e}^{-\beta F_i\left(\frac{Z-\omega_2}{\omega_1}\right)},$$

and thus

$$\forall Z \in [a_{i+1}, b_{i+1}], \qquad F_{i+1}(Z) = F_i\left(\frac{Z-\omega_2}{\omega_1}\right) + \frac{\ln(\omega_1)}{\beta}. \tag{2.20}$$

Note that the additive constant $\ln(\omega_1)/\beta$ on the right hand side of the previous equality is irrelevant, since free energies are defined up to an additive constant.

Discretization. In practice, the ranges of ξ_i and ξ_{i+1} are discretized into k bins centered at $\{z_1, \cdots, z_k\}$ and $\{Z_1, \cdots, Z_k\}$. The centers of the bins are matched as follows: if $\omega_1 > 0$, then z_ℓ is associated with Z_ℓ; while if $\omega_1 < 0$, then z_ℓ is associated with $Z_{k-\ell+1}$. The initialization of the free energy F_{i+1} is thus computed for each bin ℓ' of ξ_{i+1} using its final value in the corresponding bin ℓ of ξ_i, and Equation (2.20). In practice, because ABF uses the mean force rather than the free energy, it is the mean force at each bin, $Z_{\ell'}$ that is initiated using the final estimate of the mean force in bin z_ℓ, i.e the value of the numerator of Equation (2.18) for time $N\Delta t$. Additionally, the number of samples for each bin of ξ_{i+1} is initialized with the values of the previous ABF run in the same manner (i.e. according to the sign of ω_1, using the final values (at time $N\Delta t$) of the denominator in Equation (2.18)). The new ABF simulation, ABF_{i+1} is then actually equivalent to continuing an ABF run with the CV ξ_{i+1}, instead of starting a new simulation.

Some results obtained with the free energy initialization scheme described in this section are given in Supp. Mat. Section 2.7.3.B.

2.3.4 Two-dimensional toy example

In order to illustrate the methodology developed in this section, we come back to the 2 dimensional example introduced in Section 2.2.5, with the same parameters as in this section unless otherwise specified. For this toy example, the routines for unbiased simulations as well as eABF simulations were implemented in python using the `numpy` module.

We demonstrate on this simple 2D example that reweighting the sampled configurations leads to a fast convergence to the CV $\xi_1(x) = x_1$ (which is the CV obtained by training on a long unbiased trajectory), whereas not reweighting can lead the algorithm to be trapped, continually learning different CVs between consecutive iterations. We start with the initial unbiased simulation, which is purposedly stopped before it crosses the first energy barrier: The trajectory is simulated starting from the initial condition $q^0 = (-1, 0)$, for 2×10^7 timesteps, with $\Delta t = 10^{-3}$, keeping 1 out of 50 datapoints. The time horizon of the simulation is thus $T = 2 \times 10^4$, and the dataset contains $N = 4 \times 10^5$ samples belonging to only one metastable state.

We then use this dataset to start two separate learning frameworks. The first one uses reweighting at each iteration as discussed in Section 2.3.1. The second one does not use reweighting before achieving a new round of training (i.e. the weights $\widehat{w}_j$ in (2.7) are all set to 1). All autoencoders are initialized with the same parameters at each iteration, and are trained using 80% of the data for training (and 20% for validation), with batch size $b = 400$, and $N/b = 8 \times 10^2$ steps per epoch for a maximum of 100 epochs with early stopping when the validation loss does not improve for 20 consecutive epochs. At each new iteration, for both schemes, eABF is performed for 1.2×10^6 timesteps with $\Delta t = 10^{-3}$, and 1 in every 3 configurations are kept. The time horizon of the simulation is thus $T = 1.2 \times 10^3$ and the datasets all contain $N = 4 \times 10^5$ samples. eABF is run with a force constant $\kappa = 50$. Note again that the biased simulations have a smaller time horizon than the unbiased simulation, since biasing accelerates the exploration of all states of the configurational space. The CV intervals are determined using the range $[z_{\min}, z_{\max}]$ of the values the CV takes over the training data. Regardless of the chosen CV interval, we use a grid of 200 bins to discretize the CV. Each simulation is started at the same point $(x_1, x_2) = (-1, 0)$. The estimated mean force over a certain bin is only applied after 100 samples are collected inside this bin. Finally, the CV gradients needed for ABF are extracted from the autoencoders using `keras` functions.

Figure 2.4 illustrates heat maps of the encodings obtained at each iteration for both schemes. All CVs were renormalized using Equation (2.12) to have a range within $[0, 1]$. As shown in Figure 2.4 left, after only two iterations, the reweighted model finds the CV x_1 and stabilizes. The biased model, on the other hand, learns a different direction at each new iteration, so that the learned CV does not converge, but rather seems to oscillate between two different functions.

2.4 Computational details

2.4.1 Software packages and libraries

We use the `keras` [161] library to wrap the `tensorflow` [162] neural network module in `python`.

In order to construct an entirely automated code, all molecular simulations were performed with the `openmm` software [163] within its `python` API. The adaptive biasing and collective variable analysis were mainly performed using `plumed` [164–166]. To link these two modules, we used the `openmmplumed` plugin module [167]. Additionally, the `colvar` [168] `abf integrate` utility was used to recover the potential of mean force from the gradients computed by eABF.

Our goal is to have an implementation that is automated and entirely runnable from `python`. Consequently, our implementation is practical and easy to use, but not necessarily computationally optimized.

2.4.2 Alanine dipeptide in vacuum

2.4.2.A Parameters

The algorithm was implemented with the following ML and MD parameters.

- **Machine Learning.** The input features are chosen as the Cartesian coordinates of only the backbone atoms of alanine dipeptide, instead of the complete peptide, making the input data x of dimension $D = 3 \times 8 = 24$. Structural rotational alignment (using the Kabsch algorithm [84]) to a reference configuration and re-centering are included in preprocessing to respectively eliminate rotational and translational invariances. Note that the same reference structure, namely a configuration which falls within the C5 state, is used for all our experiments.

  All autoencoders used for AE-ABF runs have the same symmetrical topology: In addition to the input layer of dimension $D = 24$, the encoder contains two fully connected layers, of respective dimensions $G = 40$ and $d = 2$ (computed using the FVE curve as explained in Remark 4), and all activation functions are tanh. This topology is the same as the one used by the authors of MESA [93, 94] on the same system. Note however that additional experiments were done with $G = 8$ instead. The obtained results (namely the learned CV) were the same.

  We use the `adam` [169] optimization algorithm. The model is trained for a maximum of 2000 epochs, and early stopping [170] is used to stop training when the validation loss stops decreasing for 50 consecutive epochs, making the actual number of epochs in practice approximately 300. The learning rate used is always $\eta = 10^{-3}$.

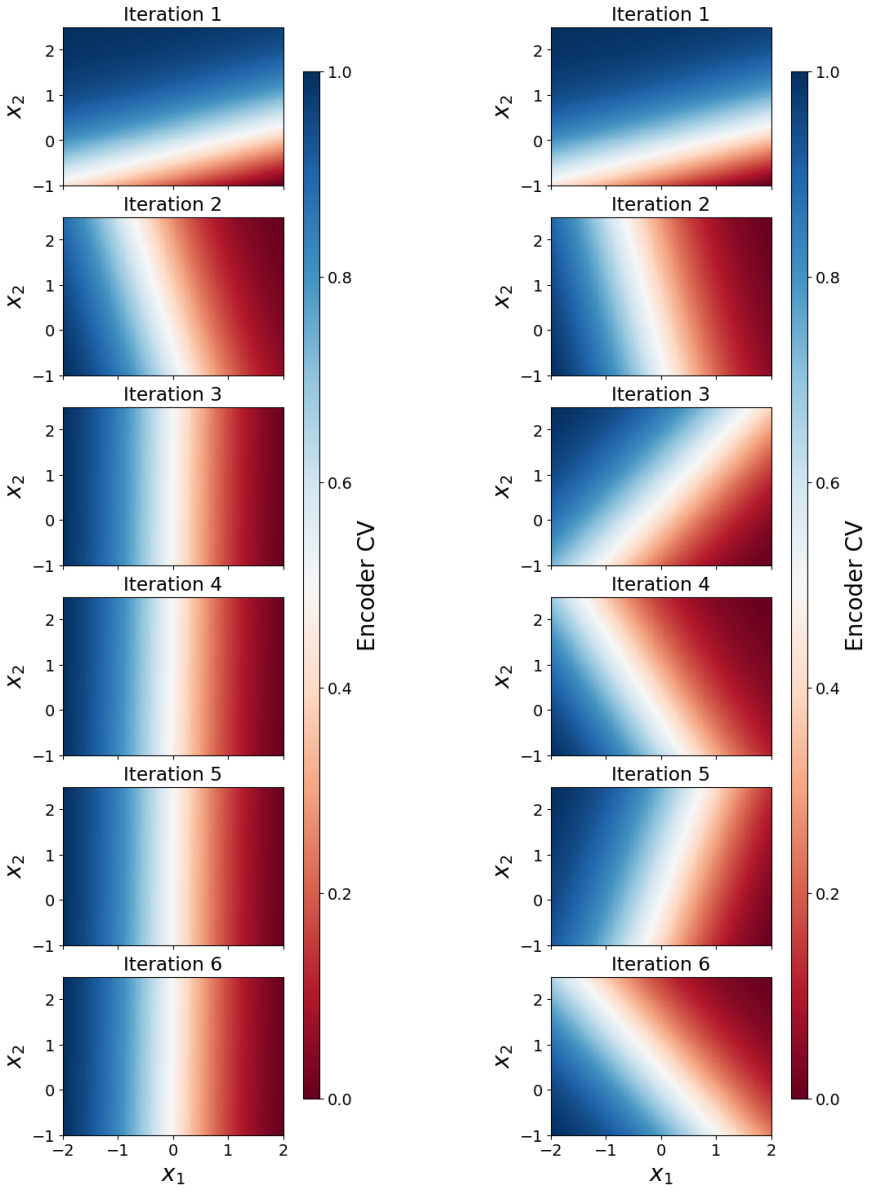


Figure 2.4: Encodings obtained at each iteration of AE-ABF with (left) and without (right) reweighting. Left: Reweighting guides learning to the same CV at each iteration, and the algorithm quickly converges to the CV $\xi_1(x) = x_1$. Right: The simulations are biased using a different free energy at each iteration. The CV does not stabilize.

At each new iteration of the algorithm, both the new trajectory and the previous one are combined to construct the training set. As shown in Supp. Mat. Section 2.7.3.A (for the 2D example), this can help the algorithm converge with smaller simulation

times, as well as possibly speed up the convergence.

- **Molecular Dynamics.** All simulations (biased and unbiased) are performed using the Amber ff99SB forcefield [171], under Langevin dynamics at a temperature $T = 300$ K, non periodic conditions with cutoff $r_c = 1$nm, a friction coefficient $\gamma = 1$ ps^{-1}, no bond constraints and a timestep $\delta t = 1$ fs. For consistency, the same peptide configuration is used as input in all runs, this configuration is obtained after 500 steps of energy minimization.

  For eABF simulations, collective variable biasing intervals are selected at each iteration of the AE-ABF algorithm as the ranges (along the two CV directions) of the CV values obtained from the encoders on the training data. A grid of a fixed value of 50 bins for each of the two dimensions of the CV is used. The force constant κ is kept to its default `plumed` value: $\kappa\beta = \frac{1}{(\delta z)^2}$, where δz is the grid spacing. The fictitious mass m_λ is set to $m_\lambda = \kappa \left(\frac{\tau}{2\pi}\right)^2$, where $\tau = 0.5$ is the default value for the relaxation time in `plumed` (also called extended time constant in `colvars`). The estimated biasing force is applied to a sampled point after a minimum of 500 samples are collected in the corresponding bin. The gradients of the CVs are numerically estimated in `plumed` instead of being extracted from the autoencoders. Additionally, `plumed` ensures a correct propagation of all calculated forces with respect to any translational and rotational alignments applied to the system.

2.4.2.B Simulation speed

Under the setup described in Sections 2.4.1 and 2.4.2.A, unbiased simulations ran with a speed of $\sim$ 1500ns/day, while biased eABF simulations using encoder CVs had a speed of $\sim$ 200 ns/day. As a comparison, biased simulations using regular CVs of the system (namely the dihedral angles Φ and Ψ) ran at a speed of $\sim$ 1000 ns/day. The computation of the encoder CVs and their approximate derivatives in `plumed` is thus computationally quite expensive compared to the other steps of the algorithm for this low dimensional system in vacuum. Note that for larger systems, in particular solvated systems, the overhead will be much smaller in proportion. Let us also emphasize again that no specific effort was made to obtain a more efficient computational chain, as the focus of this work is primarily methodological.

2.4.3 Chignolin in explicit solvent

Chignolin is a 10-residue miniprotein with a beta-hairpin structure. It is another well-studied system, particularly for its distinct folding states captured using MD at accessible timescales [29, 172]. Lacking an X-ray for the wild-type form, herein, we use the CLN025 mutant (PDB ID: 5AWL) in explicit solvent as a more realistic and challenging system of our approach. After the addition of hydrogen atoms, the system contains 166 atoms.

2.4.3.A Parameters

For chignolin, the AE-ABF algorithm was implemented using the following parameters:

- **Machine Learning.** The selected input features are the Cartesian coordinates of the $C\alpha$ atoms of the chignolin miniprotein. The input data is thus of dimension

$D = 3 \times 10 = 30$. Similarly to alanine dipeptide, structural rotational alignment to a reference structure, and centering are used to eliminate rotational and translational invariances. The reference structure used is a folded state conformation.

The autoencoders used for AE-ABF runs have an encoder made of two fully connected layers, of respective dimensions $G = 10$ and $d = 2$. All activation functions are tanh. The `adam` optimization scheme is used for a maximum of 2000 epochs with early stopping after no improvement over 50 consecutive epochs. The learning rate used is $\eta = 5 \times 10^{-4}$.

As for alanine dipeptide, each new autoencoder training uses both the new and the previous trajectory as a combined dataset.

- **Molecular Dynamics.** Chignolin is solvated using a water box with a padding of at least 1.2 nm. Na^+ and Cl^- ions are added to render the system neutral. All simulations are performed using the Amber ff99SB-ILDN forcefield [173] and the TIP3P water model [174], under Langevin dynamics at temperature $T = 340$K and friction coeficient $\gamma = 1$ ps^{-1}, with a timestep $\delta t = 2$ fs. Nonbonded interactions are computed with the particle mesh Ewald with cutoff $r_c = 1$ nm, and constrained hydrogen bonds (using the LINCS algorithm [175]). All simulations are performed in the NVT ensemble. The initial configuration for the simulations is obtained after a performing 1000 steps of energy minimization. For eABF simulations, collective variable biasing intervals are selected at each iteration of AE-ABF as the ranges (along the two CV directions) of the CV values obtained from the encoders on the training data. The variables are then rescaled to have a range of $[-1, 1]$. A grid of 50 bins with equal sizes is used for each of the two dimensions of the CV. All other parameters are set to the same values as for alanine dipeptide, with the exception of the force constant which was manually set to $\kappa\beta = 353.74$. This change to a lower value of κ was made to ensure the stability of the biased simulations. the other `plumed` parameters are the same as the ones used on alanine dipeptide.

2.4.3.B Simulation speed

In the case of solvated chignolin, and under the computational framework described in Sections 2.4.1 and 2.4.3.A, unbiased simulations ran at a speed of $500 - 600$ns/day, while biased simulations (using autoencoder CVs) ran at $50 - 70$ns/day.

2.5 Results

This section shows results of the AE-ABF algorithm applied to the systems of alanine dipeptide in vacuum and solvated chignolin.

2.5.1 Alanine dipeptide in vacuum

Conventionally the 2D intrinsic manifold of alanine dipeptide in vacuum is described by the Φ and Ψ backbone dihedrals [176, 177]. As mentioned in Remark 4, we have used the plateau method to check that the bottleneck optimal dimensionality is indeed $d = 2$. See Supp. Mat. Section 2.7.4 where we provide the FVE curve computed using a long unbiased trajectory of alanine dipeptide.

We first present in Section 2.5.1.A the ground truth CV, which is the CV obtained from training an autoencoder over a long unbiased simulation. The quality of this CV is assessed by measuring its ability to recover the free energy landscape of the dihedral angles Φ, Ψ. Section 2.5.1.B contains the results of applying AE-ABF to alanine dipeptide. For a concise presentation of this toy system, see Supp. Mat. Section 2.7.4.

2.5.1.A Autoencoder ground truth collective variable

The goal of AE-ABF is to obtain the same CV as the one we would obtain from training an autoencoder on a long unbiased simulation, and to use this CV to bias the dynamics for a better sampling. In order to check that the first goal is attained, we construct a reference CV which will serve as a ground truth to compare our results to. Additionally, this ground truth CV's ability to efficiently bias the dynamics is assessed.

Constructing the ground truth CV. We first sample a 1.5 μs trajectory, saving frames every 1.5 ps. The Ramachandran scatter plot of this trajectory is given in Figure 2.5a. We then train an autoencoder of the same topology and the same initial parameters and hyper-parameters (learning rate, activation functions, etc) as for the model described in Section 2.4.2.A. The resulting CV's projection on the 1.5 μs trajectory is plotted in Figures 2.5b and 2.5c. It can be observed that this ground truth CV first clearly separates the C7ax state from the others, and also distinguishes between the C5 and C7eq states as well as the dihedral angles (Φ, Ψ) do. Moreover, a strong correlation between the CVs and the dihedral angles can be observed (visually illustrated by color plots of the Φ and Ψ with respect to the CV space in Figures 2.5b and 2.5c). This correlation is further confirmed with a high value of the regression score between the ground truth CV and (Φ, Ψ): $R^2 = 0.967$.

Ground truth CV biasing efficiency. In order to assess the ground CV's efficiency for biasing the dynamics, we measure its ability to estimate the system's free energy, i.e the (Φ, Ψ) free energy surface. For this, we first run a long 500 ns eABF simulation using (Φ, Ψ) as collective variables in order to obtain a reference free energy landscape F. We then run a 500 ns eABF simulation using the ground truth CV. At any time t, the estimate G_t of the CV's free energy is then used to compute an approximation $\tilde{F}_t$ of F by reweighting histograms: using k bins to discretize (Φ, Ψ) in each direction, the estimate $\tilde{F}_t^{\,j,l}$ in the bin $[z_{1,j} - \varepsilon, z_{1,j} + \varepsilon) \times [z_{2,l} - \varepsilon, z_{2,l} + \varepsilon)$ for $1 \leq j, l \leq k$ is given by:

$$\exp(-\beta \tilde{F}_t^{\,j,l}) = \sum_{i=1}^{N_t} \exp(-\beta G_t \circ \xi(x^i)) \mathbb{1}_{\{z_{1,j} - \varepsilon \leq \Phi(x^i) < z_{1,j} + \varepsilon\}} \mathbb{1}_{\{z_{2,l} - \varepsilon \leq \Psi(x^i) < z_{2,l} + \varepsilon\}} \ , \quad (2.21)$$

where ξ is the ground truth CV, N_t is the number of samples collected at time t and $2\varepsilon = \frac{2\pi}{k}$. When a certain bin (j, l) of the (Φ, Ψ) space is not visited, the corresponding value of $\exp(-\beta \tilde{F}_t^{\,j,l})$ is of course 0, making the free energy in these bins infinite. Instead, the free energy $\tilde{F}_t^{\,j,l}$ in these unvisited bins is made equal to the maximum of the free energy values encountered in the visited bins.

The free energies F and $\tilde{F}_t$ are of course defined up to additive constants, which are chosen in order minimize the error between these two quantities (see Remark 5 in Supp. Mat. Section 2.7.5.A). Figure 2.6 shows the reference free energy, the free energy estimate at the final time $\tilde{F}_{500 \text{ ns}}$ and the error per bin between these two values. The C7ax state is less precisely

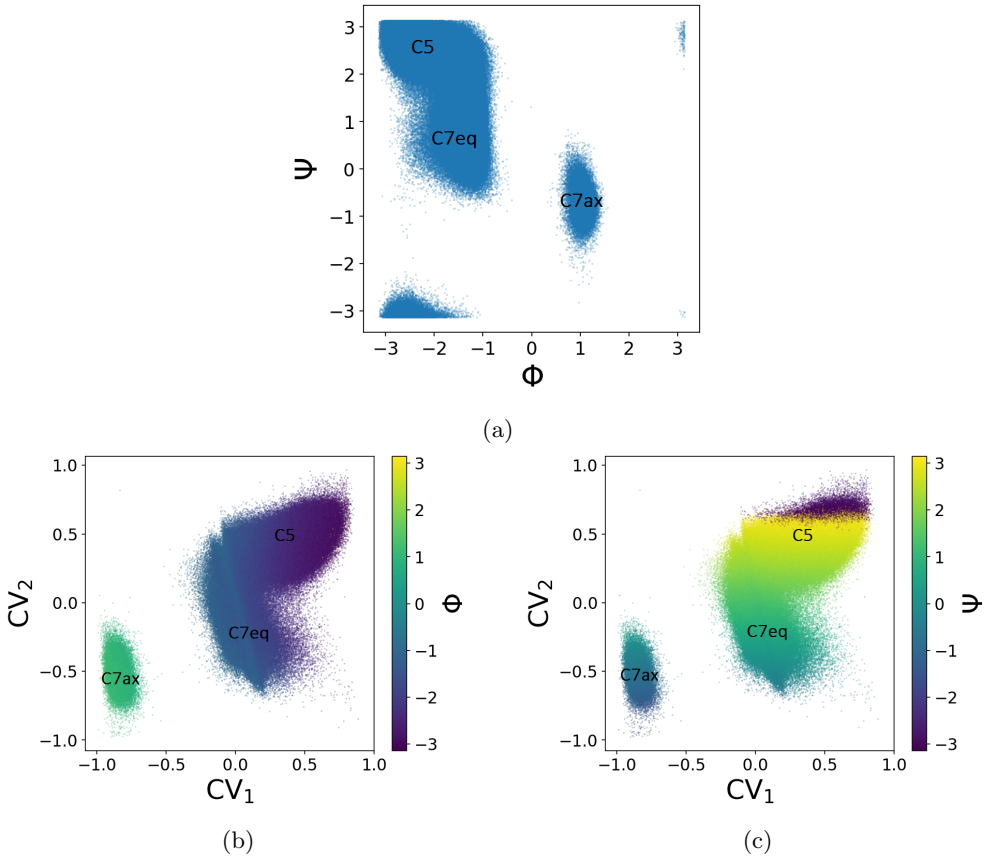


Figure 2.5: (a) Ramachandran scatter plot of a 1.5 μs unbiased trajectory of alanine dipeptide in vacuum which visits the three metastable states of the molecule. (b) Scatter plot of the autoencoder CVs learned from the unbiased trajectory, using Φ-based coloring. (c) Scatter plot of the autoencoder CVs learned from the unbiased trajectory, using Ψ-based coloring. The CVs are projected on the same 1.5 μs unbiased trajectory. The regions corresponding to C5, C7eq and C7ax are approximately defined using Φ and Ψ values.

estimated in shape than the C5 and C7eq states, particularly at extreme values of Ψ, where an incorrect local minimum is identified ($\Phi \approx 1$ and $\Psi \approx -2.5$). In addition, some transition regions are not visited during the simulation (dark purple regions in the middle plot of Figure 2.6). These regions include transition states located on the right and on the bottom left of the C7ax basin, indicating that transitions between C7ax and C5/C7eq through those paths are very rare. Similar errors in the free energy landscape, both in magnitude and localization, were observed in similar works [93].

In order to further assess the sampling efficiency of dynamics biased by the free energy associated with the ground truth CV, we also compute the number of transitions between metastable states per nanosecond encountered in a typical eABF simulation. For this, we run three eABF simulations of 50 ns each, using the ground truth CV as reaction coordinate, and obtain an average number of transitions of about ~ 63 per ns. This number is quite close

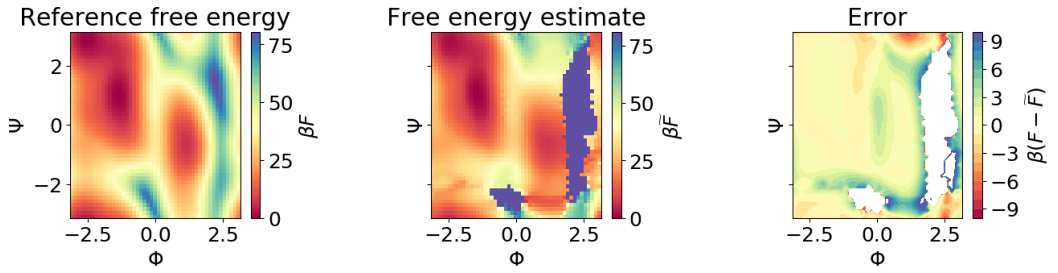


Figure 2.6: Left. Reference free energy computed by eABF over Φ, Ψ. Middle. Estimate of the free energy computed by reweighting histograms. Transition states located on the right of the C7ax basin are not visited. Right. Filled contour plot of the difference $\beta(F - \tilde{F})$ in each bin. The free energies are optimally aligned.

to the value of about ~ 80 transitions per ns obtained for eABF simulations using (Φ, Ψ) as collective variables. However, biasing with the ground truth CV does not enable the sampling of the transition states on the right of C7ax in the Ramachandran plot, contrarily to biasing with (Φ, Ψ). For more analysis on the sampling efficiency of the ground truth CV, see Supp. Mat. 2.7.5, where this CV is compared against (Φ, Ψ), as well as a CV obtained by training an autoencoder on configurations produced by a biased simulation without reweighting.

2.5.1.B Results of AE-ABF

We now apply AE-ABF to this system, and compare our learned CVs to the ground truth shown in Figure 2.5. Instead of stopping the algorithm at CV convergence, we run AE-ABF for a fixed number of iterations $I_{\max} = 9$ and later check at which iteration convergence has occured. This is to check whether the algorithm really stabilizes at the converged CV. We consider that convergence is reached at iteration i when the linear regression score $s(\xi_i, \xi_{i-1}) \geq s_{\min} = 0.996$.

To determine the simulation time per iteration of AE-ABF, a compromise should be made between the duration of the simulation (which we want to be as small as possible) and the time needed for the free energy estimate to be stable (indicating it has converged to the free energy) so that it can be used for the reweighting of sampled data. The results given below correspond to a simulation time of 10 ns per iteration. Other runs of AE-ABF using larger values of simulation times were also performed, and gave similar results.

For each simulation, atomic positions are recorded every 100 timesteps. This corresponds to $N = 10^5$ datapoints at each iteration. Figure 2.7 illustrates the sampled trajectories (Ramachandran plots) through 7 iterations of AE-ABF on alanine dipeptide in vacuum.

To compare the consecutive learned CVs, we project the encoders obtained from each iteration on the 1.5 μs unbiased trajectory. Figure 2.8 shows scatter plots of the autoencoder CVs in Φ and Ψ-based coloring. Regression scores between consecutive CVs, and between the CVs from each iteration and the ground truth CV (illustrated in Figure 2.5) are also given in Table 2.1. These regression scores are computed using data from the 1.5 μs unbiased trajectory, and serve only for the analysis of our results. Note that during the AE-ABF algorithm itself, the regression scores between consecutive CVs are computed using the AE-ABF trajectories (precisely the two last sampled trajectories at each iteration). This

Iteration	previous CV	Ground Truth CV	(Φ, Ψ)
0	–	0.9418	0.922
1	0.872	0.849	0.892
2	0.868	0.897	0.853
3	0.922	0.996	0.973
4	0.999	0.997	0.972
5	0.999	0.996	0.970
6	0.999	0.996	0.971
7	0.999	0.995	0.967
8	0.998	0.993	0.966
9	0.999	0.994	0.968

Table 2.1: Linear regression scores with AE-ABF for 9 iterations. Each line corresponds to an iteration, where the regression score is computed between the learned CV and: the CV from the previous iteration (2nd column); the ground truth CV of Figure 2.5 (third column); and the 2D vector (Φ, Ψ) (fourth column). CV convergence occurs at iteration 4, where the regression score is above 0.996. The regression score values show that the converged CV is almost perfectly similar to the ground truth CV, and also explains very well Φ and Ψ.

is the score mentioned in Algorithm 1, which is in practice used to monitor convergence during the algorithm and determine a stopping rule. These scores, which have values similar to the ones obtained by comparing with the $1.5\,\mu$s unbiased trajectory, are reported in Supp. Mat. Section 2.7.4.

As illustrated in Figure 2.7, the consecutive eABF simulations all explore the three main metastable states of alanine dipeptide in vacuum. Notably, the transition between the C7ax and C7eq states is more and more sampled through the iterations. In particular, iteration 4 shows an increase of the number of sampled transition state configurations. As shown in Figure 2.8 and Table 2.1, this coincides with the convergence of the learned CVs to a good approximation of the ground truth CV obtained in Figure 2.5. In conclusion, we have obtained virtually the same CV learned from a $1.5\,\mu$s trajectory, after only 4 iterations of AE-ABF, equivalent to a total of 40 ns of biased simulation time, in addition to the first 10 ns unbiased iteration.

2.5.2 Chignolin

This section provides results of AE-ABF applied to the solvated chignolin mini-protein [178]. We first present in Section 2.5.2.A the different conformational states of chignolin. Section 2.5.2.B then illustrates the 5 separate long unbiased trajectories of chignolin that were sampled in order to provide training data for learning a ground truth CV, which is presented in Section 2.5.2.C. The results of AE-ABF applied to solvated chignolin are presented in Section 2.5.2.D. Finally, the sampling efficiency of the autoencoder CV is compared to that of more traditional choices of CV (namely well defined interatomic distances).

2.5.2.A Metastable states of chignolin

The chignolin miniprotein is a small 10-residue protein which folds into a β hairpin in water. Three main states of the protein can be distinguished: the folded state, the misfolded state

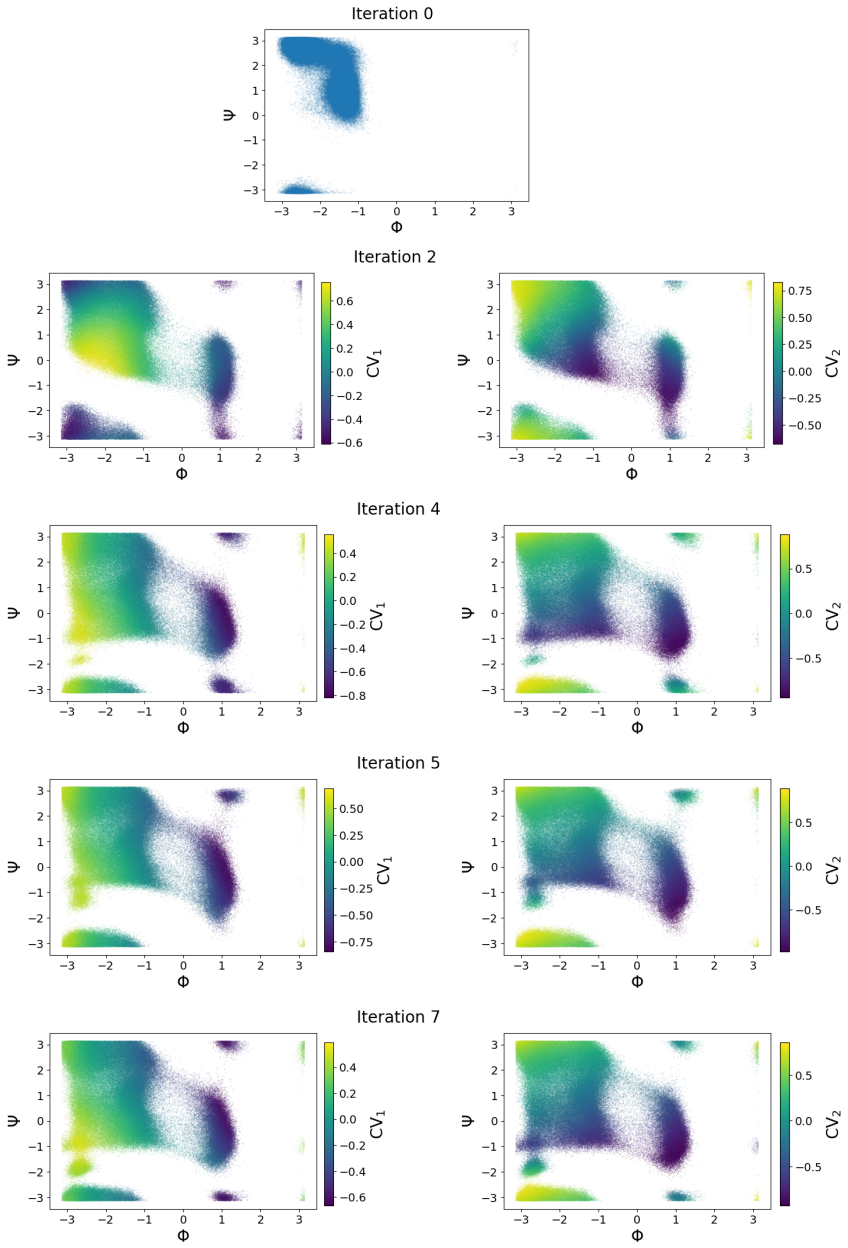


Figure 2.7: AE-ABF for 7 iterations (not all are shown here). Ramachandran scatter plots of each trajectory. The first trajectory is unbiased. The coloring corresponds to values of the first component of the CVs (Left) and the values of the second component of the CVs (Right). Visually, the first component of the CV converges approximately to a function of Φ, and the second component to a function of Ψ.

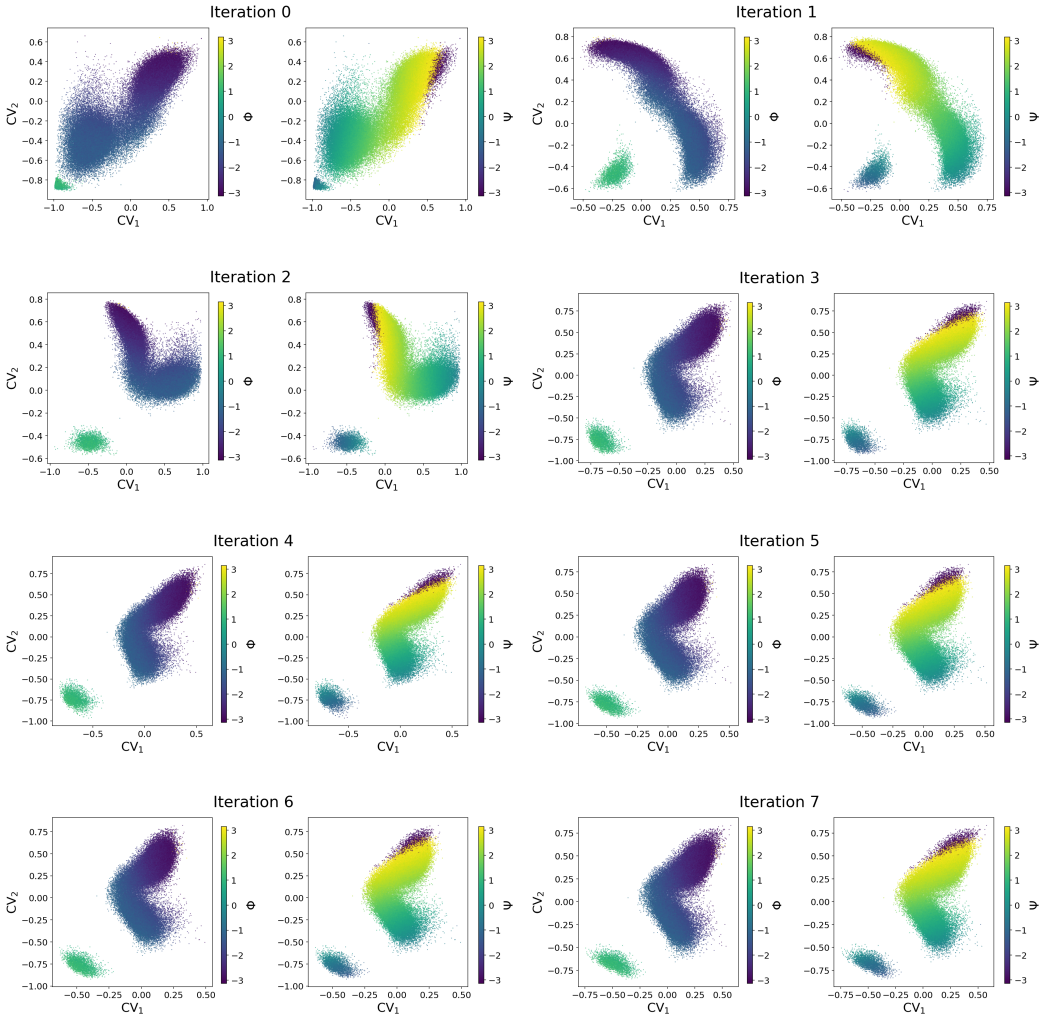


Figure 2.8: Autoencoder CVs through 7 iterations of AE-ABF (in addition to the initial unbiased iteration). The CVs are scatter plotted with Φ and Ψ colorings. The plots show that the CV obtained with AE-ABF is very similar to the ground truth CV shown in Figure 2.5. Plots were made using encoder projections over the same unbiased 1.5 μs simulation.

and the unfolded state [172]. These states are illustrated in Figure 2.9. The misfolded state is characterized by the formation of an incorrect hydrogen bond between ASP3N (nitrogen atom of residue 3: ASP) and GLY7O (oxygen atom of residue 7: GLY) instead of the correct hydrogen bond between ASP3N and THR8O (oxygen atom of residue 8: THR) formed in the folded state. In other words, the misfolded state is represented by small values of the distance D(ASP3N-GLY7O) while the folded state corresponds to small values of D(ASP3N-THR8O). The unfolded state shows a more open form of the hairpin and thus represents higher values of both distances.

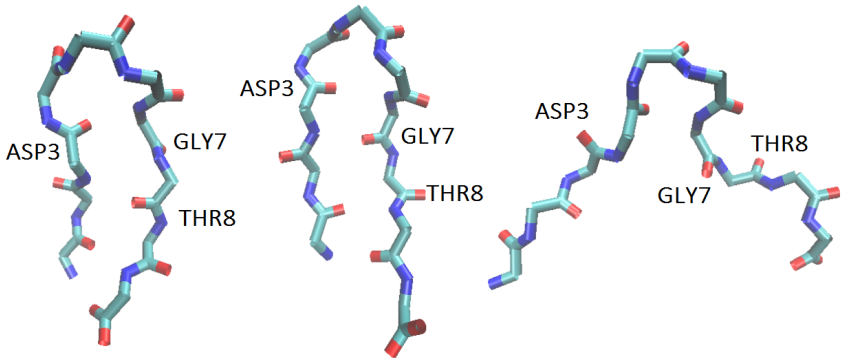


Figure 2.9: Three main states of chignolin in water. Left to right: Folded, misfolded and unfolded. The difference between the folded and misfolded states can be observed here in the orientation of the oxygen (red) atoms in GLY7 and THR8.

2.5.2.B Unbiased simulations

Following the protocol described in Section 2.4.3, we run 5 unbiased simulations of 2 μs each. For each simulation, configurations are saved every 5000 timesteps, i.e 10 ps. The resulting trajectories thus contain 2×10^5 configurations each. All simulations are started from the same initial configuration in the folded state. Figure 2.10 shows the alpha carbon root mean square deviation plotted for each trajectory. These RMSD values are computed with respect to the initial folded conformation. The folded, misfolded and unfolded states correspond approximately to RMSD values in ranges $[0, 0.2]$, $[0.2, 0.4]$ are $[0.4, 0.8]$ respectively (these values are expressed in nm). Each simulation accomplishes at least one transition among these three states. Additionally, we include in Figure 2.10 the values of the two distances of interest, D(ASP3N-GLY7O) and D(ASP3N-THR8O), observed during the simulations.

We also show in Figure 2.11 the free energy landscape for the reaction coordinate (D(ASP3N-GLY7O),D(ASP3N-THR8O)) computed using these five simulations using a histogram of 50 bins in each direction. The folded and misfolded states correspond to local minima, while the unfolded state is shallower and wider.

2.5.2.C The ground truth collective variable

The 5 simulations illustrated in the previous section are concatenated to form a dataset of 10^6 points, which we use to learn a ground truth CV. We train an autoencoder of the structure and parameters described in Section 2.4.3. The obtained CV is projected over

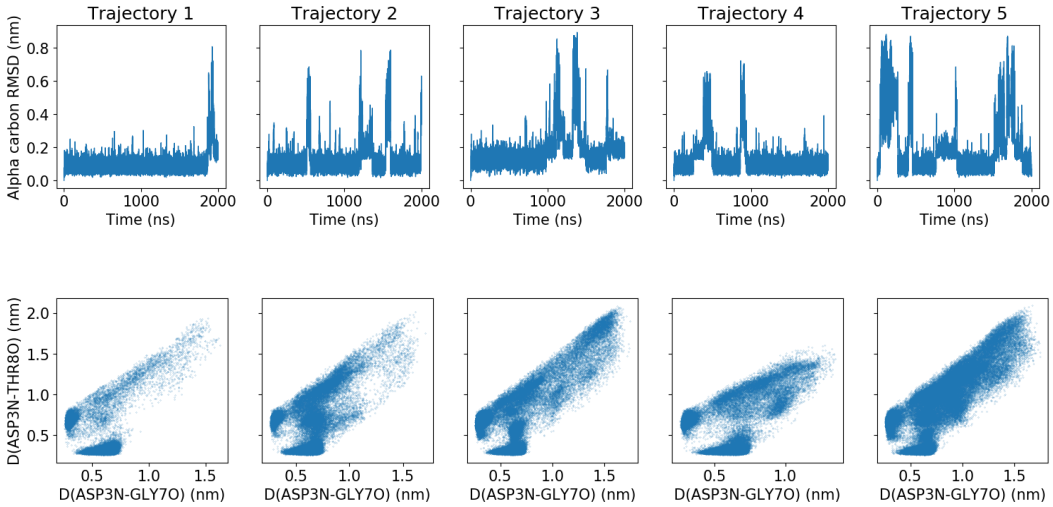


Figure 2.10: Top. Alpha carbon RMSD and radius of gyration values over each trajectory. Bottom. Scatter plots of values of the distances D(ASP3N-GLY7O) and D(ASP3N-THR8O) sampled for each trajectory.

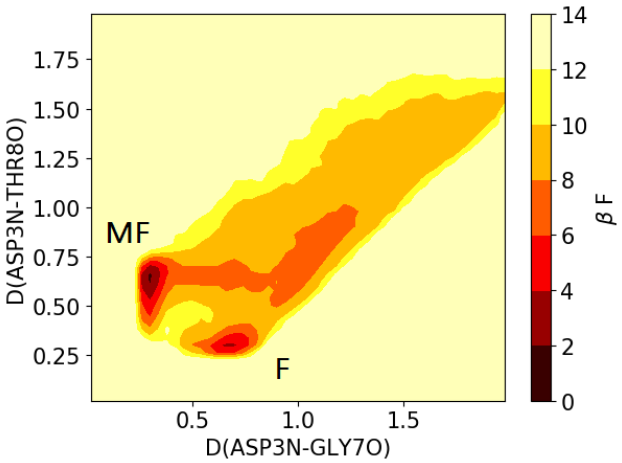


Figure 2.11: Free energy landscape of the 2D space formed by distances D(ASP3N-GLY7O) and D(ASP3N-THR8O). The folded (F) and misfolded (MF) states correspond to two separate basins. The unfolded state covers a more scattered area.

the dataset in Figure 2.12 and colored according to the values of the distances D(ASP3N-GLY7O) and D(ASP3N-THR8O). Figure 2.12 shows that the obtained CV separates the misfolded and folded states into two clusters, while the unfolded state covers a large part of the remaining space.

Similarly to alanine dipeptide, we seek to compare the CVs obtained by AE-ABF on chignolin against this ground truth CV.

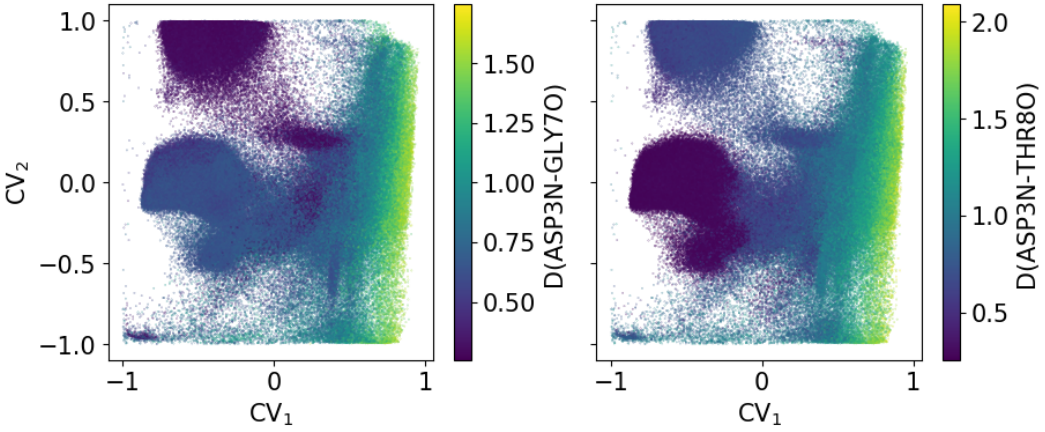


Figure 2.12: Autoencoder CV projected over the unbiased trajectories and colored according to values of D(ASP3N-GLY7O) (left) and D(ASP3N-THR8O) (right). Two clusters appear: The top cluster (located at $CV_1 < 0$ and $CV_2 > 0.5$) corresponds to the misfolded state, while the lower cluster (located at $CV_1 < 0$ and $-0.5 < CV_2 < 0.5$) represents the folded state. The rest of the 2D space corresponds to the unfolded state.

2.5.2.D Results of AE-ABF

We apply AE-ABF to chignolin using a fixed number of 7 iterations. We consider that CV convergence is reached when the regression score is higher than $s_{\min} = 0.78$, where $s_{\min}$ is computed using the bootstrapping procedure described in Supp. Mat. Section 2.7.2. The first unbiased simulation had a time horizon of 100 ns and the following biased simulation a time horizon of 50 ns. For each biased simulation, atomic positions are saved every 500 timesteps.

We compute the values of the carbon alpha RMSD and the two distances at each iteration. These values are plotted in Figure 2.13 to illustrate the gradual exploration of the conformational space of chignolin throughout the iterations of AE-ABF. The obtained plots clearly show that the learned CVs are able to accelerate crossing between the folded, misfolded and unfolded states.

Additionally, we again compute the regression scores between consecutive CVs, as well as between each CV and the ground truth CV. Results are presented in Table 2.2. The obtained regression scores are generally much lower than those obtained for alanine dipeptide, but we still achieve convergence of the CV as defined by the threshold determined in Supp. Mat. Section 2.7.2.

2.5.2.E Sampling with the autoencoder CV

The previous section shows that AE-ABF is able to sample the three states of Chignolin in the course of one iteration, i.e. a 50 ns simulation. Here, we compare sampling efficiency between the autoencoder CV (i.e the ground truth CV introduced in Section 2.5.2.C) and the 2D CV composed of the distances (D(ASP3N,GLY7O), D(ASP3N,THR8O)). For each CV, we sample 2 eABF simulations of 60ns each. The biasing domain for the autoencoder CV (resp. the distances) is $[-1, 1]^2$ (resp. $[0.2, 1.75]^2$). We compare the regions of the space

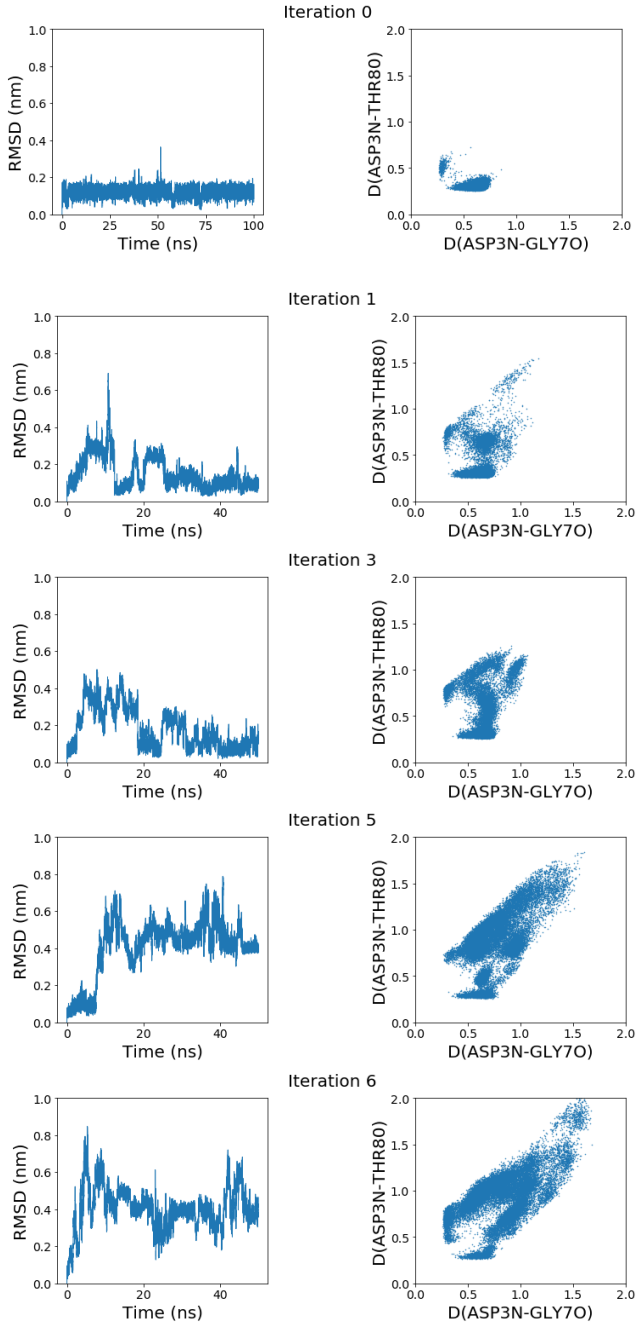


Figure 2.13: Six iterations of AE-ABF (in addition to the initial unbiased iteration). Left: Evolution of carbon alpha RMSD for each iteration. Right: Sampled values of the two distances. The final rounds of AE-ABF achieve a considerably better exploration of the conformational states. (Not all iterations are shown).

Iteration	previous CV	Ground Truth CV	vector of distances
0	–	0.119	0.099
1	0.344	0.495	0.363
2	0.349	0.413	0.594
3	0.517	0.772	0.705
4	0.684	0.765	0.585
5	0.947	0.790	0.600
6	0.855	0.801	0.674

Table 2.2: Linear regression scores with AE-ABF for 6 iterations. Each line corresponds to an iteration, where the regression score is computed between the learned CV and: the CV from the previous iteration (2nd column); the ground truth CV (third column); and the 2D vector (D(ASP3N,GLY7O), D(ASP3N,THR8O)) (fourth column). CV convergence occurs at iteration 5 (regression score above 0.78). The regression score values show that the converged CV is also well correlated with the ground truth CV. The CVs however do not achieve high regression scores with the distances (D(ASP3N,GLY7O), D(ASP3N,THR8O)).

visited by the eABF runs. The comparison is done over the distance space, where three regions are distinguished: $[0.4, 0.8] \times [0, 0.4]$ defines the folded state; $[0, 0.4] \times [0.4, 0.8]$ defines the misfolded state, while the rest of the space represents the unfolded state. Figure 2.14 shows the sampled values of (D(ASP3N,GLY7O), D(ASP3N,THR8O)), while Figure 2.15 shows the state assignments for each sample to track transitions. It can first be observed that the trajectories sampled with eABF using (D(ASP3N,GLY7O), D(ASP3N,THR8O)) cover a wider region in the unfolded state. However, these trajectories achieve fewer transitions to the misfolded and folded states, which are thus poorly sampled. This is possibly caused by the small proportion of the space taken up by these states in the distance space (see e.g. Figures 2.10 and 2.11) as opposed to the coordinate space (see Figure 2.12). Equivalently, this can be viewed as an issue concerning the choice of the biased sampling interval for (D(ASP3N,GLY7O), D(ASP3N,THR8O)). This issue does not arise for the autoencoder CV as it takes values in a bounded domain, making the sampling interval straightforward to determine.

2.6 Conclusion and Future Work

In this paper, we have given a new version of an iterative algorithm for collective variable learning with autoencoders and enhanced sampling. Our method relies on a very important reweighting procedure to ensure the convergence of the CV throughout the iterations. This reweighting procedure was both validated theoretically and in practice on simple toy examples. We have then fully described our method and have included some suggestions on several improvements of the basic algorithm based on the transfer of information between consecutive iterations. Finally, we have demonstrated the interest of our method on the alanine dipeptide system in vacuum as well as on the solvated chignolin system.

Future work involves a more in depth analysis of the information transfer refinements proposed in Section 2.3.3, specifically the free energy initialization procedure. Additionally, a third form of information transfer, where the previously learned model could be used to initialize the new learning model in a transfer learning fashion would also be interesting to investigate.

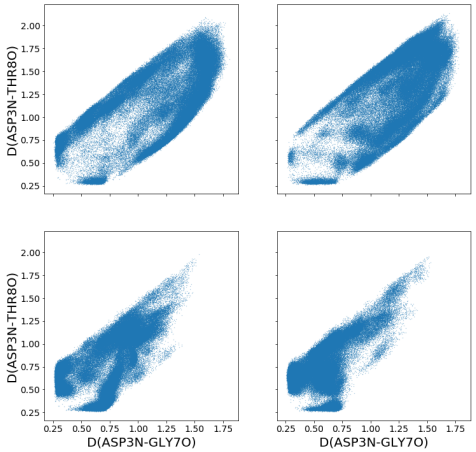


Figure 2.14: Sampled values of the distance space. Top: eABFs using (D(ASP3N,GLY7O), D(ASP3N,THR8O)). Bottom: eABFs using the ground truth CV.

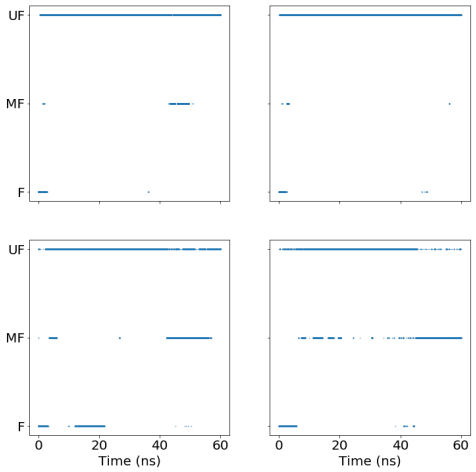


Figure 2.15: State assigned to each sample over 60ns eABF trajectories. Top: eABFs using (D(ASP3N,GLY7O), D(ASP3N,THR8O)). Bottom: eABFs using the ground truth CV.

The application of our algorithm using other learning models than autoencoders could also be explored. In particular, introducing an iterative algorithm with a transfer operator based method such as VAC is a very attractive possibility. It would also be interesting to explore whether using more sophisticated forms of neural networks e.g. convolutional topologies, could add something to the learned representation. Moreover, the choice of the data input representation is another part of the algorithm that could be optimized. So far, we have only worked with the aligned Cartesian coordinates, but directly using internal coordinates can prove efficient for more complex systems. In addition, more preprocessing steps could be considered to normalize the data, to obtain a set of already independent input variables, etc. For example, whitening transformations could be used to remove underlying

correlations.

Finally, the application of AE-ABF to more complex systems is crucial to test what the method can truly accomplish, and explore any possible limitations.

2.7 Additional definitions and results

2.7.1 Theoretical definitions and practical details

This section highlights some important theoretical and practical points on the implementation of FEBILAE.

2.7.1.A Collective variables and free energy biasing

In MD, a common choice for the biased distribution $\widetilde{\mu}$ is to consider a free energy biased distribution. In order to make this concept precise, we recall in this section some definitions on collective variables, free energy and free energy based biasing, including general formulas for the Boltzmann–Gibbs measure and the free energy biased measure. Here, we consider the positions of a given system $q = (q_1, \dots, q_m)$, where $q_i \in \mathbb{R}^3$ for systems in vacuum or $(L\mathbb{T})^3$ in the case of periodic boundary conditions in a cubic box. We denote by $D = 3m$ the dimensionality of the system. For this system, the marginal of the canonical measure ν with respect to the positions q is defined as:

$$\nu(dq) = Z_\nu^{-1} \mathrm{e}^{-\beta V(q)} dq \ , \tag{2.22}$$

where $V : \mathbb{R}^D \to \mathbb{R}$ is the potential energy function, $Z_\nu = \int_{\mathbb{R}^D} \mathrm{e}^{-\beta V(q)} dq$ is a normalization constant and $\beta = (k_B T)^{-1}$ is proportional to the inverse temperature. Here, ν corresponds to the original distribution μ considered in (2.1).

The full sampling of the canonical measure by molecular dynamics simulations is typically infeasible because of metastability. Many methods aim at overcoming this issue. In particular, free energy biasing methods change the original potential V to $V - F \circ \xi$, where F is the free energy associated with the collective variable $\xi : \mathbb{R}^D \to \mathbb{R}^d$. The CV ξ is often chosen such that $\xi(q_t) = z_t$ is a slow process given the dynamical evolution equation of q_t. The function ξ typically characterizes the conformational changes between metastable states. Recall that the free energy associated with a CV ξ and the canonical measure ν is (up to an irrelevant additive constant):

$$F(z) = -\frac{1}{\beta} \ln \left(\int_{\Sigma(z)} \mathrm{e}^{-\beta V(q)} \ \delta_{\xi(q)-z}(dq) \right) , \tag{2.23}$$

where $\Sigma(z) = \{q \in \mathbb{R}^D, \xi(q) = z\}$ and the delta measure $\delta_{\xi(q)-z}(dq)$ is supported on $\Sigma(z)$. The density of the image of ν by ξ is thus proportional to $\mathrm{e}^{-\beta F}$.

Free energy biasing corresponds to sampling the canonical measure associated with the bias potential $V - F \circ \xi$, that is:

$$\nu_F(dq) \propto \mathrm{e}^{-\beta(V - F\circ\xi)(q)} dq \ . \tag{2.24}$$

A simple computation shows that the marginal distribution in the ξ variable of this biased measure is uniform. Indeed, the density of the image of ν_F by ξ is:

$$\int_{\Sigma(z)} \mathrm{e}^{-\beta(V(q) - F\circ\xi(q))} \ \delta_{\xi(q)-z}(dq) = \mathrm{e}^{\beta F(z)} \mathrm{e}^{-\beta F(z)} = 1 \ . \tag{2.25}$$

The biased measure $\mathrm{e}^{-\beta(V-F\circ\xi)(q)}dq$ thus has a uniform marginal law along ξ. This implies that $\xi(q_t)$ is less metastable under the free energy biased dynamics.

The biasing procedure is however only as good as the CV used to perform it: if most of the metastability of the system is captured by the CV ξ (i.e. when the CV effectively resolves the metastable states), the method effectively renders the dynamics diffusive and annihilates metastability; conversely, if the CV ξ fails to describe the most important metastable directions, biasing along it will likely be ineffective. This last point motivates the need to find a good choice of collective variable.

2.7.1.B Autoencoder optimization step

The autoencoder is typically trained using a gradient descent optimization algorithm: the parameters $\mathbf{p}$ are sequentially modified at each optimization step using a steepest descent algorithm with a stepsize $\eta > 0$ called the learning rate. In this paper, we use mini-batching to approximate the gradient of the loss function, meaning that each optimization step only uses a subset (a mini-batch) of the data to modify the parameters $\mathbf{p}$. More precisely, we denote by b the number of points in a mini-batch. The data is first randomly reshuffled, then at each learning step r, the loss is computed as the mean loss over datapoints $x^{rb+1}, \dots, x^{(r+1)b}$:

$$\mathcal{L}_r(\mathbf{p}) = \frac{1}{b}\sum_{i=rb+1}^{(r+1)b} \|x^i - f_{\mathbf{p}}(x^i)\|^2.$$

The parameters $\mathbf{p}$ are thus updated as follows:

$$\mathbf{p}_r = \mathbf{p}_{r-1} - \eta\nabla_{\mathbf{p}}\mathcal{L}_r(\mathbf{p}_{r-1}).$$

To compute the gradient $\nabla_{\mathbf{p}}\mathcal{L}_r(\mathbf{p}_{r-1})$, back-propagation is used. The learning proceeds in epochs, where one epoch corresponds to the number of steps $\left\lfloor \frac{N}{b} \right\rfloor$ required to visit the entire dataset (with $\lfloor . \rfloor$ the integer part of a number). A maximum number of epochs is specified, after which training is stopped. The learning is usually stopped earlier, when the loss no longer decreases. To assess this, part of the data is used as a validation set: This subset is not used in training, and its mean loss is tracked to stop the gradient descent algorithm when it no longer decreases. This procedure is called early stopping. The validation loss is used instead of the total loss to stop the algorithm to avoid overfitting the dataset.

It is very important to note here that during learning and after, the model has access to the gradient of the value of any neuron with respect to the value of any other neuron from a previous layer (including the input). This is useful later, in the context of enhanced sampling, one needs to extract the mapping of the encoder and its gradient.

2.7.1.C Multiple solutions

As mentioned in Section 2.2.3, the optimization problem does not necessarily have a unique solution. In addition, the learning loss can have multiple local minima. More precisely, the value $\mathbf{p}_\mu$ obtained at the end of learning, which corresponds to one local minimum, may depend on the initial value given to $\mathbf{p}$ at the beginning of the learning, and on the value of the learning rate η used in the gradient descent. As a consequence, when we want to assess the impact of the distribution of the input data on the learned parameters, we make sure that, for a given test case, the numerical setting (initial parameters and topology of the autoencoder, optimization parameters, etc) are kept constant from one experiment to another.

2.7.1.D Sample weights normalization

For all the experiments performed in this paper, the weights used in Equation (2.7) are the ones defined in Equation (2.6) multiplied by N. This is to follow the default weight normalization used by the `keras` module in `python`, which we use for all our experiments, and where the weights are normalized to sum to the number of samples: $\sum_{i=1}^{N} w_i = N$, instead of 1. Of course, this only adds a multiplicative factor to the loss, and thus does not change anything to the learning from a theoretical viewpoint. This is simply a practical choice that allows us to use default values for most of the parameters of the optimization procedure.

2.7.2 Regression score to assess CV convergence

The AE-ABF algorithm stops when the CVs learned from the new iteration are sufficiently similar to the ones obtained from the previous iteration. We make precise in this section how this similarity is quantified.

The R^2 score in regression models

We consider a dataset $\mathcal{Z} = (z^1, \dots, z^N)$, with $z^i \in \mathbb{R}^p$, and corresponding values (called labels) $\mathcal{Y} = (y^1, \dots, y^N)$, with $y^i \in \mathbb{R}^{p'}$. We call an optimized regression model between $\mathcal{Z}$ and $\mathcal{Y}$ a mapping M from the inputs $z^1, \dots, z^N$ to the outputs $y^1, \dots, y^N$ trained so as to minimize the error

$$\sum_{i=1}^{N} \|M(z^i) - y^i\|^2,$$

where $\|\cdot\|$ is the Euclidean norm. Here we consider a linear model, which corresponds to $M(z) = Wz + b$, for a given matrix $W \in \mathbb{R}^{p \times p'}$, and bias vector $b \in \mathbb{R}^{p'}$.

The precision of this regression model, for the data $\mathcal{Z}$ and $\mathcal{Y}$, can be assessed using the R^2 score, also called coefficient of determination:

$$R^2(M, \mathcal{Z}, \mathcal{Y}) = 1 - \frac{\sum_{i=1}^{N} \|y^i - M(z^i)\|^2}{\sum_{i=1}^{N} \|y^i - \bar{y}\|^2},$$

where $\bar{y} = \frac{1}{N} \sum_{i=1}^{N} y^i$. The R^2 score is thus simply the fraction of variance explained by the regression model M over the dataset $(\mathcal{Z}, \mathcal{Y})$. Note that this score is equal to 1 when the model M is able to output exactly y^i for each z^i. On the other hand, a baseline model which outputs $M(z^i) = \bar{y}$ for all i will have a R^2 score equal to 0.

Using the R^2 score for CV comparison

To compare two CVs ξ and ξ' of dimensions p and p' respectively, we use a test trajectory $(q^1, \dots, q^N)$. We then optimize a linear regression model M between inputs $\mathcal{Z} = (z^1, \dots z^N) = (\xi(q^1), \dots, \xi(q^N))$ and labels $\mathcal{Y} = (y^1, \dots, y^N) = (\xi'(q^1), \dots, \xi'(q^N))$. The

CV score between ξ and ξ' is then defined as the coefficient of determination $R^2(M, \mathcal{Z}, \mathcal{Y})$. Typically, $p' = p$ and $\xi = \xi_{i-1}$, the CV proposed at the i-th iteration of Algorithm 1, while $\xi' = \xi_i$.

Determining the threshold value $s_{\min}$

The stopping rule of Algorithm 1 requires determining a value $s_{\min}$ of the R^2 score above which one can consider that the CVs are sufficiently similar in order to stop the loop. Note that this value depends on various parameters, one of which is the size N of the datasets that are used to learn the CVs. Let us illustrate one approach to determining $s_{\min}$. In the setting considered in Section 2.5.1, the number of datapoints used for learning a new model at each iteration of AE-ABF is $N = 10^5$. To determine a reasonable value for $s_{\min}$ with $N = 10^5$, we use the unbiased trajectories obtained as described in Sections 2.5.1.A and 2.5.2.B. The unbiased dataset of $N_{\text{ref}} = 10^6$ points is randomly split into 10 subsets $S_1, \dots, S_{10}$. An autoencoder is trained on each subset S_k and a CV, ξ_k is learned. Then the R^2 score between pairs (ξ_k, ξ_ℓ) is computed. This provides $\frac{10\times 9}{2} = 45$ different values of the R^2 score when $N_{\text{ref}}/N = 10$. This operation can be repeated r times to obtain $45r$ realisations of the R^2 score. This procedure allows to approximate the distribution of this score for $N = 10^5$. We choose to define $s_{\min}$ as the 5% percentile of this distribution, that is to say, $s_{\min}$ is such that 95% of the $45r$ values are larger than $s_{\min}$. Figure 2.16 shows the regression scores computed for alanine dipeptide and chignolin. For alanine dipeptide, using $r = 30$, and thus 1350 realisations of R^2, we find $s_{\min} \approx 0.996$. This motivates the choice of the value $s_{\min} = 0.996$ considered in Section 2.5.1.A. For chignolin, using $r = 15$, and thus 675 realisations of R^2, we find a much lower value: $s_{\min} \approx 0.78$. The difference between the two values obtained for two different systems shows the necessity of determining the threshold $s_{\min}$ for each new system.

Note that this method relies on having a large dataset (here $N_{\text{ref}} = 10N = 10^6$ samples) to sample N-sized subsets. In practice, this dataset may not be available at the beginning of the simulation. However, a dataset of size N is sampled at each iteration of AE-ABF. After a number N_{it} of iterations of the algorithm is achieved, the N_{it} datasets which have been obtained can be compiled and used as a larger dataset of size $N_{\text{ref}} = N_{\text{it}}N$ to determine $s_{\min}$. This means that N_{it} iterations of Algorithm 1 are performed without checking the convergence of the CV, after which a value of $s_{\min}$ is determined and Algorithm 1 is continued as usual.

2.7.3 Transferring information between iterations

2.7.3.A Using trajectories from previous iterations

This section illustrates results of AE-ABF applied with a sliding window of $n_T = 1$, 2 or 3 training trajectories: At iteration $i \geq 1$, the autoencoder is trained on the last $\min(n_T, i)$ sampled trajectories.

Recall that the results for alanine dipeptide given in Section 2.5 correspond to an implementation of the algorithm which includes a sliding window of $n_T = 2$ trajectories. Here, we use the 2D system of Section 2.2.5 as an illustrative example. We use a reduced trajectory horizon of only 2×10^5 timesteps at each iteration with $\Delta t = 10^{-3}$ and save samples every 5 timesteps. We therefore sample relatively small trajectories that do not always visit the three states of the configurational space, and simulations that are not sufficiently long the free energy profile to converge. We demonstrate here that for this case, using the two or

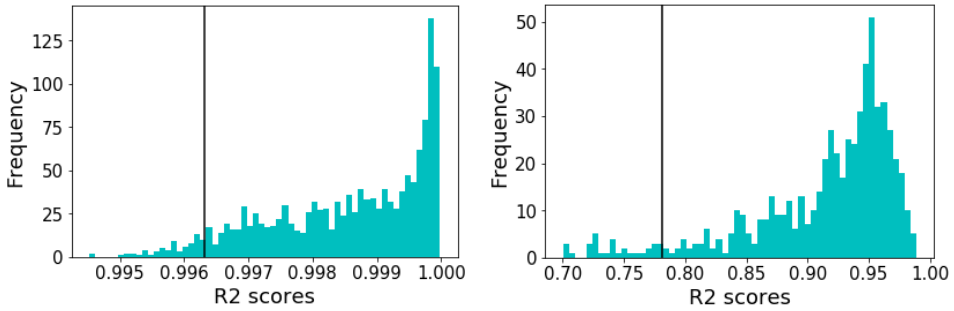
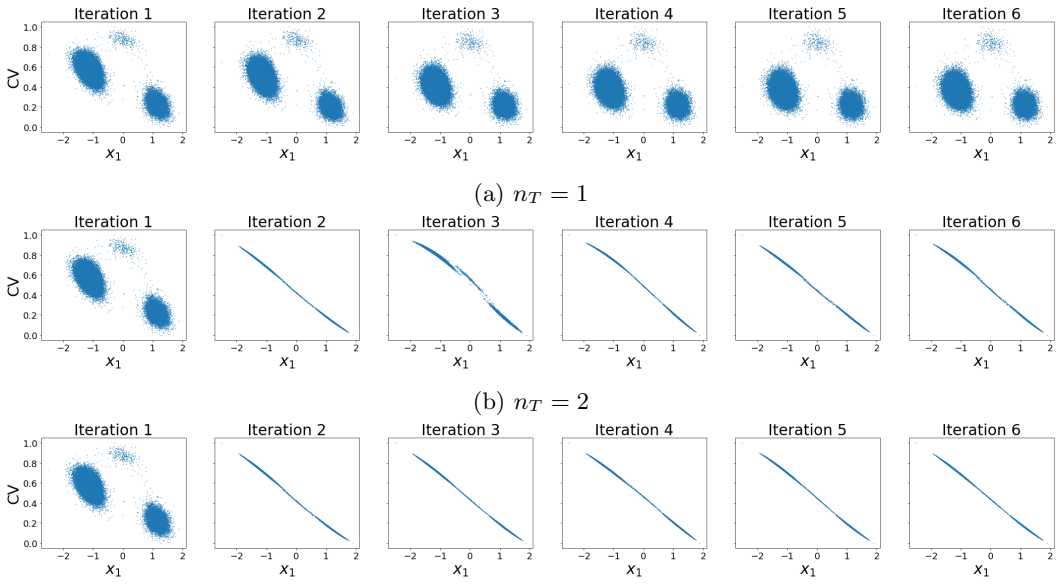


Figure 2.16: Histogram of the R^2 scores obtained using subsets of $N = 10^5$ points out of 10^6 points. The vertical black line indicates the 5th percentile. Right. Alanine dipeptide. Left. Chignolin.

three last trajectories for training ($n_T = 2$ or 3 respectively) enables convergence, contrarily to using only the new trajectory ($n_T = 1$).

We run three instances of AE-ABF for 6 iterations each, using respectively a sliding window of $n_T = 1$, 2 and 3 trajectories. The corresponding CVs are plotted using an unbiased test trajectory and shown in Figure 2.17. These plots show that learning on only one trajectory is not enough to achieve convergence to the CV $\xi(x, y) = x$, because the free energy does not converge, and the trajectories do not always explore the three metastable states. However, combining the trajectories enables convergence, as it allows for the resulting dataset to be complete (even if each trajectory only visits one of the two deep wells).



(a) $n_T = 1$

(b) $n_T = 2$

(c) $n_T = 3$

Figure 2.17: CVs obtained at consecutive iterations of AE-ABF using different numbers n_T of previous trajectories as training data.

2.7.3.B Results with free energy initialization on the 2D toy example

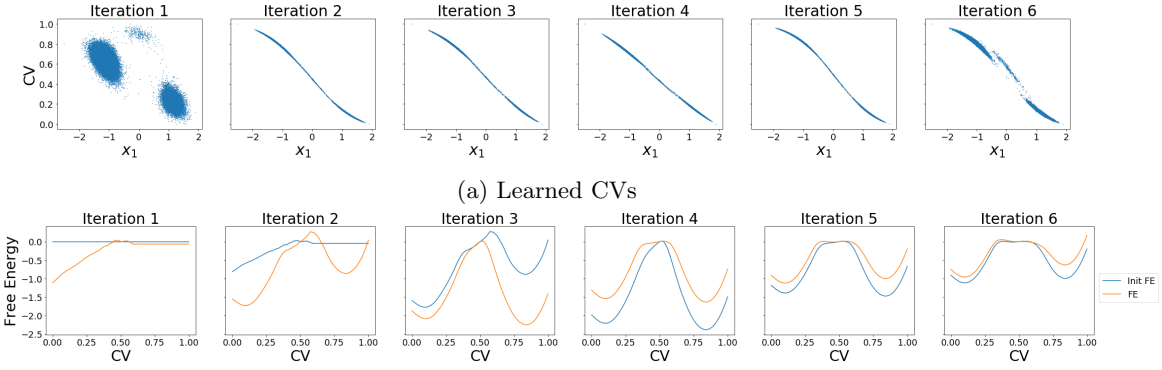
At each iteration of AE-ABF, an eABF run can be divided into two periods over each region of the configurational space. The first period only ends when a good estimate of the free energy has been computed. During this part of the simulation, the estimated bias is either not applied (when not enough samples have been gathered in the corresponding bin), or is not an accurate enough estimate to allow for a full exploration of the configuration space and thus to obtain a correct free energy. The second period represents the actual biasing of the system using the converged estimate. The free energy initialization scheme introduced as a refinement of our algorithm in Section 2.3 allows to reduce the time required for the first period, leaving more simulation time for the more important second period.

In this section, the free energy initialization procedure is implemented for the 2D three well potential system, to observe in practice the accuracy of this initialization and its impact on the convergence of the algorithm. This procedure is equivalent to running the same eABF while changing the biasing CV at each iteration of AE-ABF. To keep this ongoing eABF run consistent, at each new iteration, the simulation is started from the last sampled point of the previous iteration.

Note that the mean force initialization established in Equation (2.20) assumes a strong correlation between the consecutive CVs. Consequently, the initialization scheme is only applied when the regression score between the consecutive CVs is high enough. We chose to apply it when it is above $R^2_{\text{min}} = 0.9$.

Here, we reduce the trajectory time horizon to only 5×10^4 timesteps at each iteration with $\Delta t = 10^{-3}$, saving samples every 5 timesteps. Training is done over a sliding window of $n_T = 4$ trajectories. Without using free energy initialization, this time horizon is not enough to obtain CV convergence (even using $n_T = 4$ trajectories for learning), because the estimated free energy is far from stabilized.

Figure 2.18 shows the progressive convergence of the CV learned at each iteration to a monotonic function of the coordinate x_1, and the matching values of the free energy initialization. It is directly observed that, as the learned CV stabilizes, the free energy initialization is more and more accurate.



(a) Learned CVs

(b) Free Energies: initial guesses and final estimates

Figure 2.18: The initial guess of the free energy at iteration i (blue lines) is increasingly more similar to the free energy estimate at the end of iteration i (orange lines), as the CV converges (top scatter plots).

2.7.4 Additional details and results on alanine dipeptide

2.7.4.A States and collective variables

Alanine dipeptide is a small molecule of 22 atoms, including 8 backbone atoms. Alanine dipeptide in vacuum has been extensively studied and used in previous works, which makes it a good choice for bench marking. Figure 2.19 shows the structure of the molecule and highlights the three metastable states of alanine dipeptide in vacuum.

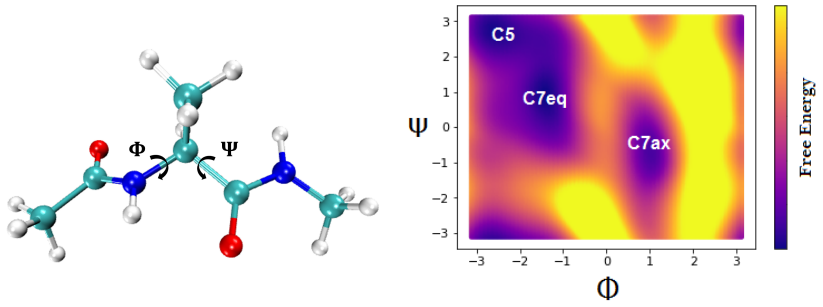


Figure 2.19: Alanine dipeptide. Left. Molecular structure. Right. Metastable states of the molecule in vacuum represented on the Ramachandran space using the free energy profile under a temperature of 300K (using 180bins in each direction). Transition times between C5 and C7eq states are fairly short, unlike the transition times between these states and the C7ax state.

2.7.4.B The FVE plateau method

Using the 1.5 μs unbiased trajectory obtained as described in Section 2.5.1.A, we train 10 different autoencoders with respective bottleneck dimensions $1, \ldots, 10$. We then compute as explained in Remark 4 the FVE value for each model, and plot these values to find a visual plateau in the FVE curve which allows to determine the optimal bottleneck dimensionality. The obtained FVE curve is shown in Figure 2.20. A knee is observed at bottleneck dimension $d = 2$.

2.7.4.C Real time regression scores of AE-ABF using the sampled trajectories

Table 2.1 above shows the R^2 scores between consecutive CVs of AE-ABF, using the 1.5 μs unbiased trajectory. As mentioned in the latter section, Table 2.1 only serves to analyze our results post run. In order to use the regression scores to establish a stopping rule, the long unbiased trajectory can of course not be used. Instead, the last two sampled *biased* trajectories are used at each iteration. Note that these trajectories are again reweighted to obtain the unbiased regression score. Table 2.3 shows the obtained scores, compared to the ones obtained by comparing to the reference CV constructed from the unbiased trajectory. The scores are quite similar. In particular, the convergence of the CV happens at the same iteration of AE-ABF for the score obtained from the long unbiased trajectory, and the ones obtained from the AE-ABF trajectories.

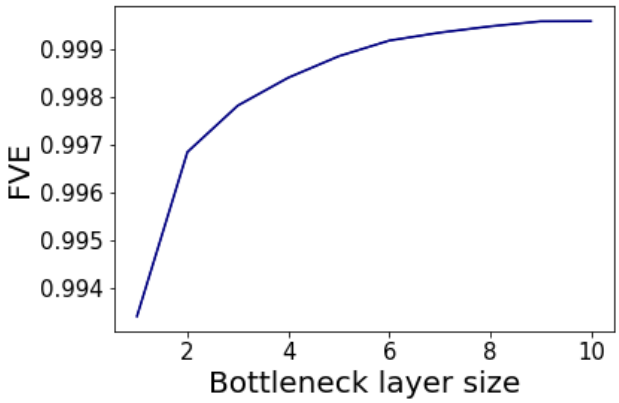


Figure 2.20: FVE curve obtained using the 1.5 μs unbiased trajectory as training data. The plateau occurs at dimension $d = 2$.

Iteration	over last 2 sampled trajectories	over reference trajectory
0	–	–
1	0.399	0.872
2	0.927	0.868
3	0.904	0.922
4	0.998	0.999
5	0.999	0.999
6	0.999	0.999
7	0.998	0.999
8	0.996	0.998
9	0.998	0.999

Table 2.3: Linear regression scores, AE-ABF for 9 iterations. Regression scores between consecutive CVs computed during the AE-ABF run using the sampled biased trajectories (first column), and using the long unbiased trajectory (second column).

2.7.5 Additional results: reweighted versus unweighted CVs

One of the goals of our algorithm is to find the same autoencoder CV that would have been obtained using long unbiased simulations. The results in Section 2.5.1 demonstrate that this goal is attained: The converged AE-ABF CV achieves a high regression score with respect to the ground truth CV. We note however that the ground truth CV by construction contains little information on transition states compared to metastable states, as it is trained on unbiased simulations. Yet incorporating some of this information could potentially improve the CV's ability to sample these transition states. Consequently, a CV that is trained on biased data may be more efficient in biasing the dynamics to enhance sampling.

In this section, we use the alanine dipeptide system to discuss this point. First, we compare in Section 2.7.5.A the sampling properties of the ground truth CV of Figure 2.5 and a CV obtained from unweighted biased data. Second, we perform in Section 2.7.5.B an AE-ABF run where the reweighting step is omitted. The sampled trajectories on this

unweighted AE-ABF are visually compared to those of a regular AE-ABF.

2.7.5.A Comparison between the ground truth CV and a CV obtained from unweighted biased trajectories

We compare here the ground truth CV shown in Figure 2.5 with a CV obtained from biased data. The latter CV is constructed by training an autoencoder on a simulation of 300 ns biased using the free energy associated with (Φ, Ψ), and applying no reweighting to the data points. The biased trajectory visits the quasi-totality of the (Φ, Ψ) space and sample multiple transitions. In this section, we denote by $\mathrm{CV_g}$ and $\mathrm{CV_u}$ the ground truth CV and the CV obtained from biased data, respectively (the subscript 'u' standing for 'unweighted'). The projections of $\mathrm{CV_u}$ and $\mathrm{CV_g}$ on the unbiased 1.5 μs dataset are shown in Figure 2.21.

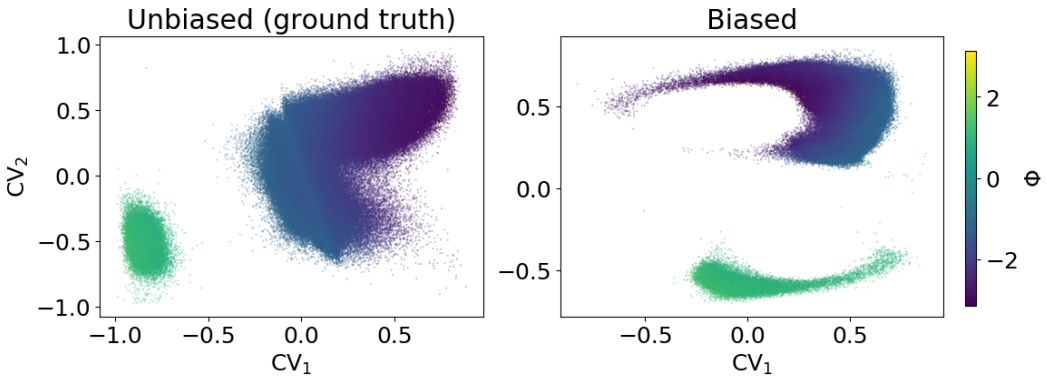


Figure 2.21: Projections of $\mathrm{CV_g}$ (left) and $\mathrm{CV_u}$ (right) over the 1.5 μs unbiased data, colored according to Φ values for clarity.

To compare the two CVs, we use the following methods:

- **Regression scores:** we measure the CVs' correlation to the dihedrals Φ and Ψ using the value of the regression score between each CV and the (Φ, Ψ) vector. $\mathrm{CV_g}$ achieves a regression score of 0.967, whereas the regression score of $\mathrm{CV_u}$ is 0.880. However, these scores are computed over configuration from the 1.5 μs unbiased trajectory, which contains only few transition states. We therefore also compute the regression scores using this time the 300 ns (Φ, Ψ)-biased trajectory which served as training data for learning $\mathrm{CV_u}$. This biased trajectory achieves a large number of transitions (over 5000 transitions between the C5-C7eq and the C7ax states). Over this trajectory, the computed regression scores between the CVs and (Φ, Ψ) are in general significantly lower. However, $\mathrm{CV_g}$ still achieves a higher regression score (0.781) than $\mathrm{CV_u}$ (0.410). When measuring these regression scores over the transition states alone, both scores are lower than 0.5, but this time $\mathrm{CV_u}$ outperforms $\mathrm{CV_g}$ (0.45 to 0.39). Here, transition state regions were determined approximately using values of Φ, with transition states corresponding to $\Phi \in [-0.5, 0.5] \cup [1.8, 2.6]$.

  These results indicate that in the case of alanine dipeptide, the unweighted CV does not seem to gain valuable information over the ground truth CV from the transition states in the dataset, and even loses (Φ, Ψ)-related information overall. One possible

explanation can be the presence of "unrealistic" conformations in the biased data that are not canceled out by reweighting.

- **Convergence of the free energy.** We run two eABF simulations, using $\mathrm{CV_g}$ and $\mathrm{CV_u}$. As done in Section 2.5.1 for the final free (see in particular (2.21)), at any time t of the obtained trajectory, the intermediate estimates G_t of the CV's free energy can be used to compute an intermediate approximation $\tilde{F}_t$ of the (Φ, Ψ) free energy F by reweighting histograms.

  The error between the current estimate of the mean force $\tilde{F}_t$ and the final estimate $\tilde{F}_{\mathrm{final}}$ obtained at the end of the simulation is computed at each timestep t as a weighted ℓ_2-distance between these two quantities:

$$\Delta F_t = \sqrt{\frac{\sum_{j,l=1}^{k} \exp(-\beta F^{j,l}) \; (\tilde{F}_t^{\,j,l} - \tilde{F}_{\mathrm{final}}^{j,l})^2}{\sum_{j,l=1}^{k} \exp(-\beta F^{j,l})}}, \tag{2.26}$$

  where the sum is over all bins (j, l) of (Φ, Ψ) and $F^{j,l}$ is the value of the reference free energy in the bin j, l.

  Remark 5. *The free energies are defined up to additive constants. In this paper, two values (e.g. estimates) F_1 and F_2 of the free energy are always optimally aligned by shifting them so as to minimize the ℓ_2-distance defined in* (2.26)*, where $(\tilde{F}_t^{\,j,l} - \tilde{F}_{final}^{j,l})$ is replaced by $(F_1^{j,l} - F_2^{j,l})$.*

  In order to assess the convergence speed of the free energy for each eABF run, we plot the time dependent value ΔF_t. As a point of comparison, the same protocol is used to compute approximations of F and their convergence speed using eABFs with the dihedrals (Φ, Ψ). In all three cases, the biasing CV (i.e. the ground truth CV, the (Φ, Ψ) dihedrals, or the CV obtained from unweighted biased data) is rescaled to have a range in $[-1, 1]^2$, and the number of bins in each direction is 50. Sampled points are saved every 10 timesteps. The simulations are 250ns long, which allows for free energy convergence. Figure 2.22 shows the value of $\beta \Delta F_t$ as a function of the eABF simulation time for the three CVs.

  It can be observed that the ground truth CV free energy profile stabilizes within a small simulation time comparable to that of the (Φ, Ψ) CV. This indicates that the conformational space is relatively well explored in a small amount of time when biasing with the ground truth CV. The convergence of the free energy profile for the CV obtained from unweighted biased data is slower, and even seems to not completely be reached at 250 ns.

- **Sampling transition states:** Finally, we compare $\mathrm{CV_g}$ and $\mathrm{CV_u}$ by estimating their ability to sample transitions between the C5/C7eq and the C7ax states. For this, we sample 2 eABF trajectories for each of $\mathrm{CV_g}$, $\mathrm{CV_u}$, and of (Φ, Ψ) for comparison. We divide the Φ space into 4 regions: C5/C7eq, TS1, C7ax, and TS2, corresponding respectively to: $[-\pi, -0.5] \cup [2.6, \pi]$, $[-0.5, 0.5]$, $[0.5, 1.7]$, and $[1.8, 2.6]$. We then plot

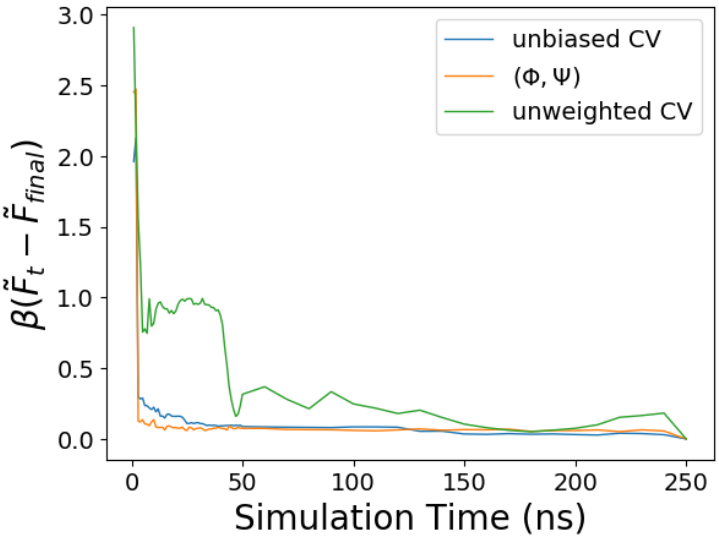


Figure 2.22: Evolution of the distance $\beta\Delta F_t$ between the final and current estimates of the free energy profiles, as a function of time.

in Figure 2.23 the regions sampled along the 6 eABF simulations using the sampled values of Φ. Figure 2.23 shows that CV_g enables a more efficient sampling of regions C5/C7eq, TS1 and C7ax than CV_u. The CV_u eABF trajectories seem to occasionally stay trapped in the C5/C7eq basin. Neither CV enables very high sampling of TS2 (compared to the other three regions, and especially compared to eABFs with (Φ, Ψ)). However CV_u eABF trajectories visit this region more often than CV_g eABF trajectories do.

In general, the obtained results suggest that CV_g is more efficient at sampling than CV_u. The latter even seems to be a rather poor choice of CV. However, it is important to note that our choice of a unweighted biased data does not constitute the only way of learning a CV from biased data, nor of incorporating transition state information into the CV. One could for instance only apply some partial reweighting to the data points, in order to remove the most unlikely configurations under the Boltzmann–Gibbs measure, while still keeping transition states associated with sufficiently low free energy barriers.

2.7.5.B AE-ABF without reweighting

This section presents the results of performing AE-ABF without reweighting. The unweighted AE-ABF is run for 9 iterations, using the same parameters as in Section 2.4.2.A, but of course without the use of reweighting to unbias the training. Just as was done for the reweighted results, we compute, for each iteration, the regression score of the learned CV with respect to the previous CV and with respect to (Φ, Ψ). Table 2.4 shows the obtained values. The regression scores between consecutive CVs are consistently lower than those obtained with the reweighted AE-ABF. More importantly, these regression scores are lower than the threshold $s_{\min} = 0.996$ determined in Supp. Mat. Section 2.7.2, indication that we cannot consider the CV to be converged. It is also important to note that the regression

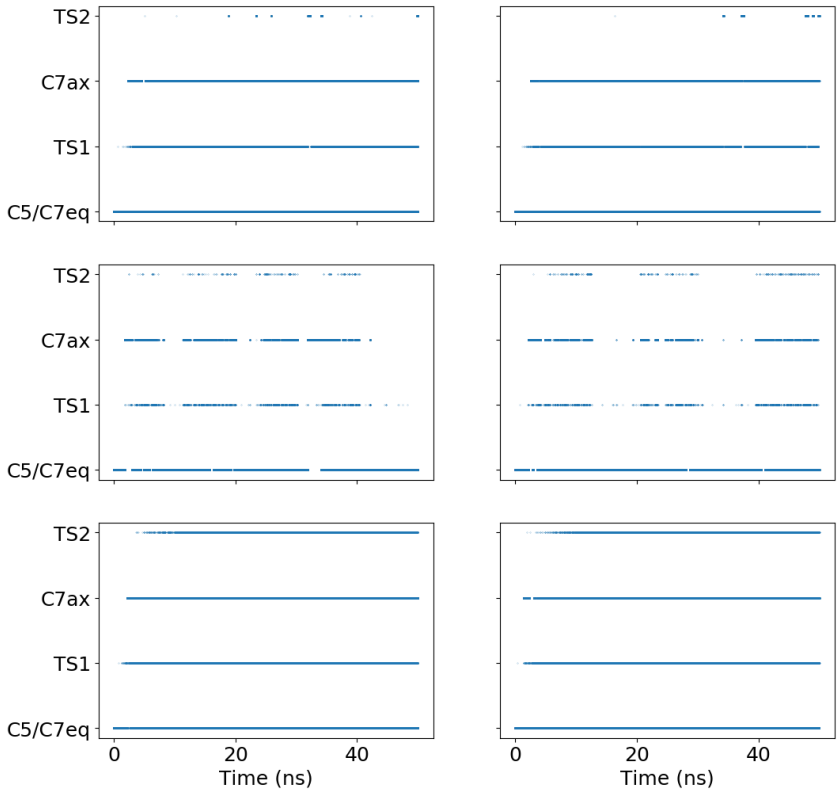


Figure 2.23: State assigned to each sample over 50ns eABF trajectories. Top: eABFs using CV_g. Middle: eABFs using CV_u. Bottom: eABFs using (Φ, Ψ).

scores between the CVs and (Φ, Ψ) is lower than with reweighted AE-ABF. These results are in accordance with the results obtained in the comparison between the unbiased ground truth and the biased CV in the previous section.

In addition, we plot in Figure 2.24 the sampled regions of the Ramachandran space at each iteration. It can be observed that the sampled trajectories are similar to those obtained with the regular weighted AE-ABF. Figure 2.25 shows the learned CV at each iteration projected on the unbiased 1.5 μs trajectory. It can be seen that, as implied by the values of the regression scores reported in Table 2.4, the learned CV does not converge.

While CVs trained on biased data do not seem to outperform CVs trained on unbiased (or reweighted) data in this particular test case, we cannot generalize this statement to every system. In cases where biased CVs would provide a sampling advantage, it would be interesting to test a hybrid version of our iterative algorithm where two CVs are computed at each iteration: one with reweighting to check the convergence of the algorithm, and one without reweighting to perform the next round of enhanced sampling.

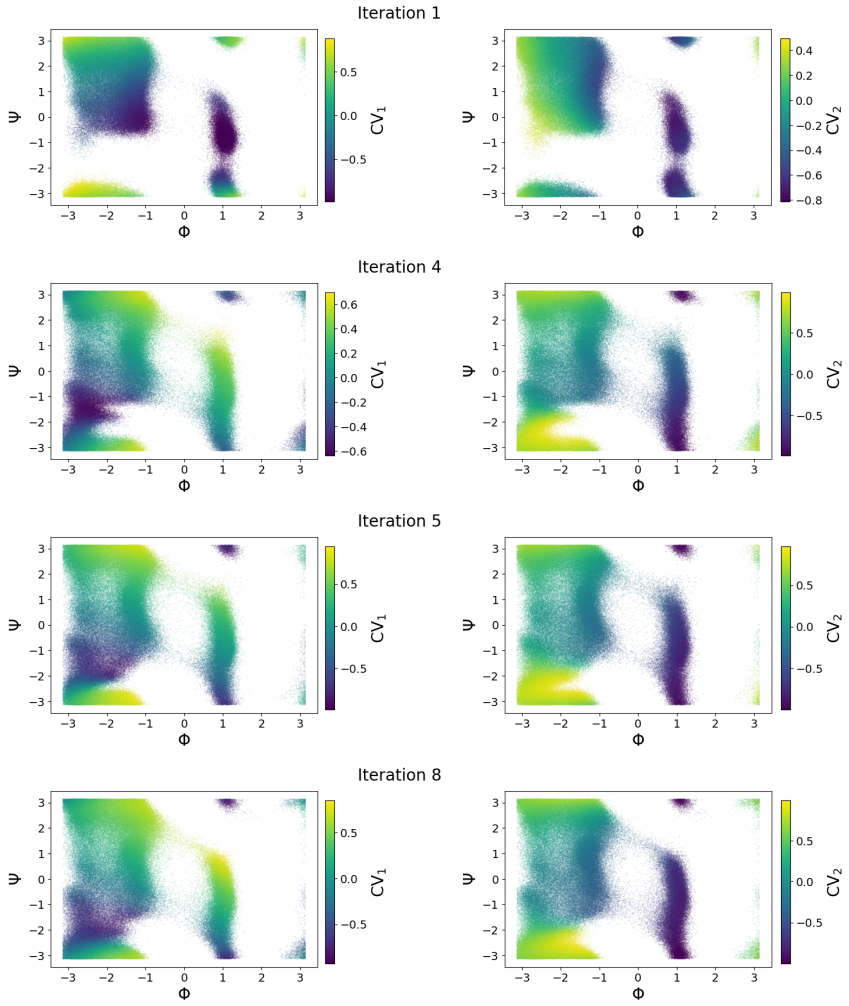


Figure 2.24: AE-ABF for 8 iterations without reweighting (not all are shown here). Ramachandran scatter plots of each trajectory. The coloring corresponds to values of the first component of the CVs (Left) and the values of the second component of the CVs (Right). The sampled (Φ, Ψ) regions are generally the same as those sampled during reweighted AE-ABF.

Iteration	previous CV	(Φ, Ψ)
0	–	0.935
1	0.850	0.896
2	0.867	0.963
3	0.915	0.930
4	0.987	0.924
5	0.911	0.910
6	0.935	0.902
7	0.882	0.935
8	0.850	0.879
9	0.901	0.890

Table 2.4: Linear regression scores with unweighted AE-ABF for 9 iterations. Each line corresponds to an iteration. The regression score is computed between the learned CV and: the CV from the previous iteration (second column); the 2D vector (Φ, Ψ) (third column). CV convergence does not occur. The regression score values with respect to (Φ, Ψ) are generally lower than those obtained with reweighted AE-ABF.

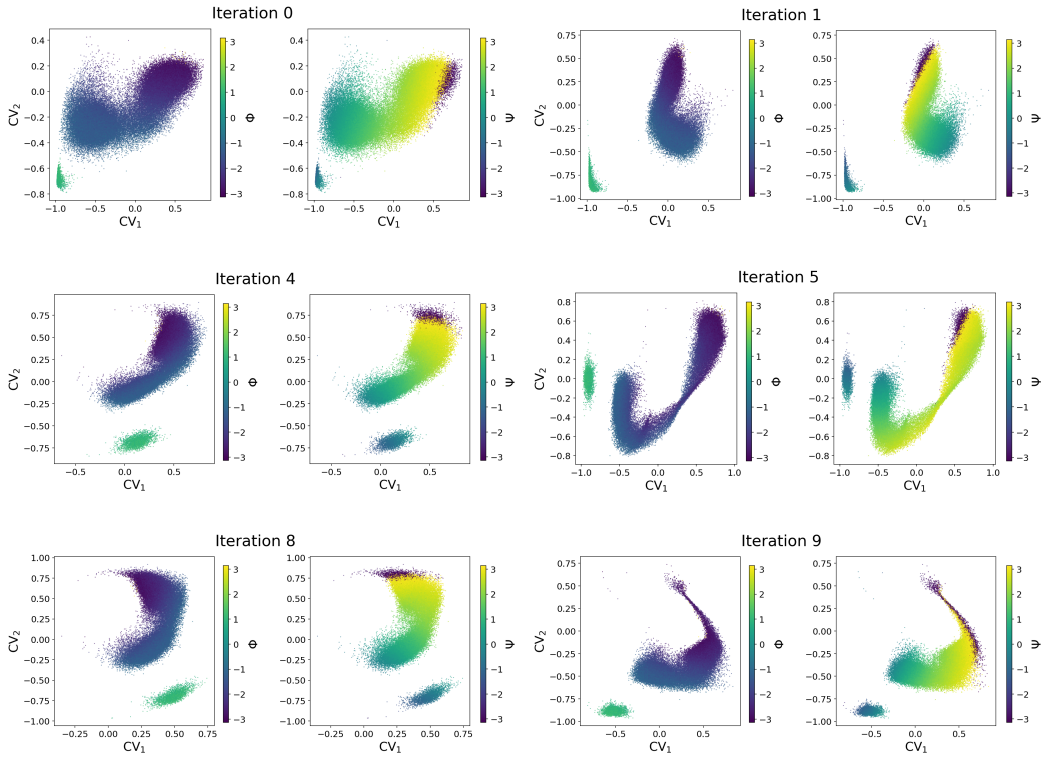


Figure 2.25: Autoencoder CVs through 9 iterations of AE-ABF without reweighting. The CVs are scatter plotted with Φ and Ψ colorings. The plots show that the CVs obtained at each iteration are not visually similar to the ground truth CV shown in Figure 2.5, or to each other.

Chapter 3

Exploring conformations of HSP90 using machine learning guided biased simulations

Chapter 2 focused on the development of a machine learning algorithm for the automatic discovery of collective variables. The algorithm we designed, FEBILAE, was implemented and applied to relatively small systems, where the molecule of interest comprises no more than 10 residues. In this chapter, we focus on the use of machine learning methods on larger molecules, such as proteins, which are usual targets in drug discovery. Because of the sizes of these systems and their often complex mechanism of action, a direct application of elaborate algorithms like FEBILAE can prove tricky, in part because biased simulations of these systems, even using a good CV, can take very long to achieve any desired transitions. As FEBILAE starts with a sub-optimal CV, learned from a small unbiased simulation, it is expected that applying it directly to large systems may be ineffective. Instead, in this chapter, a preliminary study of the protein is performed to identify and gain important knowledge on the states of the system before constructing a good collective variable using an autoencoder. The CV is subsequently used in biased MD simulations to drive transitions between the previously identified states. The autoencoder is thus used as an interpolator which helps uncover possible paths connecting the states. The system under study in this chapter is the N-terminal domain of the 90 kDa heat shock protein. In Section 3.1, we introduce this system and state the goals of this study. Section 3.2 then describes the numerical methods used in the preliminary study of the system in order to identify its conformational states, the learning of collective variables based on these states, and the simulations conducted throughout this work using the learned CVs. The results are then discussed in Section 3.3, where the learned collective variable is used to drive biased MD, and possible transitions are identified and confirmed based on structural definitions, including RMSD and dihedral angle analysis. Finally, Section 3.4 presents our conclusions and possible future directions.

3.1 Introduction

In this section, we give a brief presentation of the HSP90 protein, and more specifically the N-terminal domain, which is the site of the protein that will be studied in this chapter. We then state the goals of this study.

3.1.1 HSP90: structural information and mechanism of action

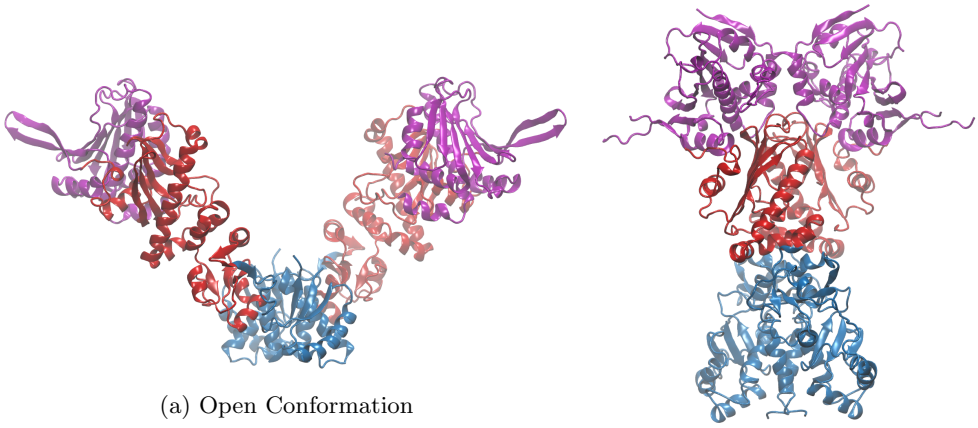
The 90 kDa heat shock protein (HSP90) is a cluster of chaperone proteins involved in various cellular processes [179–182]. It assists in the proper folding, stability and regulation of over 100 key cellular proteins, including proteins involved or required for tumor growth. Due to these functions, HSP90 is considered a promising drug target to fight diseases such as cancer, as well as Alzheimer's disease and diabetes [183–185].

HSP90 is a homodimer, i.e. a protein composed of two identical polypeptide chains called monomers (see Figure 3.1). Each monomer consists of three main structural domains, the N-terminal domain (NTD), the middle domain and the C-terminal domain (CTD) [186,187]. The N-terminal and middle domains are linked by a charged linker domain, essential for the chaperone function, helping with co-chaperone recognition [188].

HSP90's mechanism of action is an exchange between open and closed conformations on time scales ranging at least from milliseconds to several minutes [189]. It involves ATP binding, co-chaperone proteins, and a client protein [190]. When HSP90 is in apo-form (unbound), the two monomers are dimerized at their C-terminal domains. HSP90 is in this case in an open conformation, where the dimer is L shaped (Figure 3.1a). Once the ATP binds to the N-terminal domains and the inactive client protein binds to the middle domains, HSP90 is in a partially closed state, but the N-terminal domains are not bound yet. Co-chaperones are then recruited to help with the dimerization of the NTDs, and HSP90 shifts to its closed conformation (the two proteins of the dimer seem to twist around each other, forming a molecular clamp). This dimerization leads to the hydrolysis of the ATP molecules and activates the client protein. The activated client protein consequently unbinds, the NTDs are separated again, and the ADP and phosphate (which resulted from the ATP hydrolysis) are released. The dimer thus returns to its apo conformation again. Figure 3.1 shows the secondary structures of the open and closed conformations of HSP90. Both conformations were extracted from the Research Collaboratory for Structural Bioinformatic Protein Data Bank (RCSB PDB), a freely accessible database for the 3D structures of molecules.

The ATP binding occurs at the NTD of the protein [191]. The binding pocket is situated within the α-helices formed by residues $28-51$, $85-97$ and $123-130$ [192,193]. Because the ATP binding is a central part of the mechanism of action of HSP90, the binding pocket has been targeted in many drug-discovery efforts for ligand-based modulation of the protein's activity. In particular, common inhibitors of HSP90 activity such as geldanamycin, radicicol, and their derivatives, act by binding specifically to the NTD, at or near the ATP pocket, thus preventing the ATPase activity of HSP90 [193–195].

The NTD also contains a so called "lid" formed by amino acids 95 to 125, and composed of two helical segments and a highly flexible loop adjacent to the ATP binding site [196]. This loop is formed by amino acids 105 to 114. It has been shown in previous studies that this loop and the ATP lid itself present different conformations depending on the state (unbound, ATP-bound, ADP-bound, etc) of the N terminal domain [195, 197–200]. In this study, we base our analysis of HSP90 conformational states on this flexible loop. Figure 3.2



(a) Open Conformation

(b) Closed Conformation

Figure 3.1: The two main conformations of HSP90. The C-terminal domain is represented in blue, the middle domain in red and the N-terminal domain in purple. This figure was prepared using PDB entries 2IOQ (open) and 2CG9 (closed) of the RCSB databank.

shows the structure of the NTD and the structure of the loop of interest for two different conformations of the protein.

Despite the importance the ATP binding lid, and by extension the flexible loop of interest, its mechanism is not yet entirely understood due to the large time and space scales involved. Analysis of the protein dynamics within and between states is key to better understanding the HSP90 conformational cycle and the eventual targeting of this protein for drug design purposes.

3.1.2 Molecular dynamics of HSP90

Molecular dynamics of the HSP90 system have been conducted in previous studies with the aim of better understanding the mechanism of action of this protein and/or designing specific inhibitors [201]. These studies include simulations of the entire dimerized system, or of the NTD alone. In [202], the dissociation free energy of the NTD of HSP90 with ADP is computed using thermodynamic integration (see Section 1.1.3.B) with the NTD-ADP distance as the reaction coordinate. In [203], the authors compare the visited conformations of apo NTD to those of the NTD bound to 5 different inhibitors, and find significant changes in the structure of the ATP binding pocket. In [204], the authors analyze MD simulations of the NTD in its apo, ATP-bound, ADP-bound, and inhibitor bound conformations, and observe that the ATP-bound state of HSP90 presents a closed lid conformation and a somewhat constrained structure in comparison to the more relaxed and flexible conformations populated by the apo or ADP-bound and inhibitor bound states. In particular, the apo HSP90 is shown to visit different conformations of the lid, and that ligand binding stabilizes one conformation over the others. In another study [205], the same authors analyze the effect of the ATP binding on the other domains of HSP90, and locate possible allosteric sites [1] in the CTD and MD. In the same vein, the relationship between the NTD active site and

[1] Allosteric sites of a protein are regions distinct from the active site, but whose binding can still modulate the protein activity, i.e. enhance or decrease it.

allosteric sites of the CTD is examined in [206], where the full HSP90 and ATP complex is simulated with and without a number of allosteric ligands which bind to the CTD. In [207], the HSP90 system is also simulated in full, in apo and with ATP/ADP. The authors show that the apo conformation is not restricted to the open conformation shown in Figure 3.1, but can visit both stretched (i.e. the dimer angle reaching near 180°) and compact states (i.e. the dimer angle reaching near 0°). Umbrella sampling is then used to compute the free energy landscape over the dimer angle for the apo state, where the stretched and compact states are shown to correspond to low-energy configurations of apo-HSP90. Despite this conformational diversity of the apo state of HSP90, the NTD itself (as well as the other domains) is not shown to undergo any important conformational changes. In [208], simulations of the HSP90 dimer display motions spanning the whole molecule in relatively short (100 ns) timescales. These motions may indicate the start of a large conformational transition between two states of the protein.

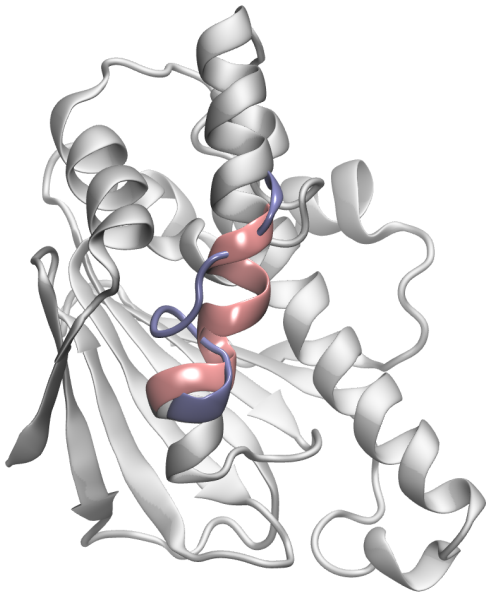


Figure 3.2: Structure of the HSP90 N-terminal domain. Two different conformations of the flexible loop: helix conformation in pink and loop conformation in blue. The two conformations are superimposed.

3.1.3 Machine learning for protein dynamics

In many of the works and methods presented in Section 1.2, and many other works in the field of machine learning based analysis of protein dynamics in general, the applications mostly concern often small toy systems such as alanine dipeptide, or mini-proteins such as chignolin, or Trp-Cage [19, 93], and are sometimes extended to relatively larger fast-folding proteins of 30 to 60 residues, such as the WW domain protein or the villin headpiece [123, 145, 146, 209]. However, the use of ML methods in studies of large proteins which present complex conformational mechanisms is becoming more and more feasible. In particular, autoencoders or autoencoder-based methods have been recently used in the analysis of

protein dynamics. These applications include for example the use of RAVE and its variations (see Section 1.2.2.C) to RNA-ligand binding in [210] where a reaction coordinate of the RNA-ligand complex is learned, allowing for the computation of the free energy profile of this system using metadynamics. Autoencoders are also used in [211] for the automatic detection of allosteric sites in proteins using MD simulations. Another example is the application of autoencoders coupled with PCA to automatically detect new conformations of proteins [212]. Finally, variational autoencoders are trained to generate new protein conformations [213] or protein variants [214].

Machine learning methods have also been used in the analysis of the dynamics of HSP90. For example, supervised models are used to classify allosteric ligands of HSP90 into inhibitors or activators of the protein [215], and generally to predict the effect of different allosteric modulators on the protein activity [216]. Linear and nonlinear supervised learning is also applied for the estimation of residence times (or conversely dissociation rates) for different ligands [217], and the determination of local protein-ligand interactions which affect the dissociation dynamics [218]. Principal component analysis is also often applied to molecular dynamics simulations of HSP90 to analyze the collective motions of the protein, and to compare the structural differences between apo and holo conformations [203, 219]. To the best of our knowledge, no works were published on the machine learning of collective variables of HSP90.

3.1.4 Aim of this study: machine learning based analysis of HSP90 states and collective variables

In this chapter, we use data-driven methods to analyze conformational changes of HSP90, focusing on the NTD loop of interest. The goals of this study are the following:

- Identify important existing conformational states of HSP90 using the loop of interest of the NTD site. This is done using a clustering analysis over values of the dihedral angles in the residues forming the loop.
- Automatically learn collective variables of the HSP90 N-terminal domain using machine learning models, specifically autoencoders, trained on simulations within the previously identified states.
- Run free energy biasing simulations using the autoencoder CVs in an attempt to sample transitions among the identified states of HSP90, and accurately identify these transitions.The transitions are confirmed by analyzing the structure of the loop of interest through RMSD calculations, dihedral angle analysis, and H-bonds formation analysis.

These three points are naturally handled successively and guide the study presented in this chapter.

3.2 Methodological approach and numerical methods

In this section, we present the methods and protocols used to perform the tasks enumerated in Section 3.1.4. Section 3.2.1 describes the flexible loop based identification of conformational states of the NTD. Simulations started from each of these states are then run under the simulation setup described in Section 3.2.2. The work described in these two sections

(3.2.1 and 3.2.2) was mainly done by Dr. Marc Biancotto. Section 3.2.3 then describes the protocol for choosing an autoencoder structure, including the number and size of layers. Finally, Section 3.2.4 presents the biased simulations setup and methods for their analysis.

3.2.1 Data mining HSP90 structural diversity

As mentioned in Section 3.1.1, a highly flexible loop is located near the ATP binding site of the NTD of HSP90. Many structures of the NTD in both apo and holo forms (i.e. with and without a ligand bound to the active site respectively) are available in the protein databank. These conformations display some specific structural differences in the loop of interest (residues: 105-114). A total of 278 different conformations of the NTD were extracted from external (RCSB PDB) and internal (Sanofi) sources. These structures were aligned and necessary fixes were performed, including atom numbering and naming, determining unresolved residues, managing multiple chains per PDB, protonation [2] and adding missing hydrogen atoms, etc. Then, the dihedral distribution of the loop of interest was used for the clustering of the structures. More precisely, for each conformation, the sine and cosine values of the dihedral angles Φ and Ψ of residues 105 to 114 were computed. The resulting dataset, which contains 278 observations of 40 features (sine and cosine of the 2 dihedrals from each of the 10 residues), was used to perform clustering of the structures using the hierarchical clustering method implemented in `scikit-learn`. The metric used was the Euclidean distance in the $(\sin, \cos)$ 40-dimensional space, so that the distance between two conformations $\alpha_1 = (\Phi_1^1, \Psi_1^1, \ldots, \Phi_1^{10}, \Psi_1^{10})$ and $\alpha_2 = (\Phi_2^1, \Psi_2^1, \ldots, \Phi_2^{10}, \Psi_2^{10})$ is given by

$$\begin{aligned} d(\alpha_1, \alpha_2)^2 = &\sum_{i=1}^{10} (\sin(\Phi_1^i) - \sin(\Phi_2^i))^2 + (\cos(\Phi_1^i) - \cos(\Phi_2^i))^2 \\ &+ (\sin(\Psi_1^i) - \sin(\Psi_2^i))^2 + (\cos(\Psi_1^i) - \cos(\Psi_2^i))^2. \end{aligned}$$

Six separate clusters were identified, indicating the existence of six key states. Two of these states, which we define as states 1 and 2, exist in both apo and holo forms, while the rest of the states have been experimentally resolved in holo conformations only. A representative structure of each cluster was selected as the crystal structure with the highest resolution reported on PDB [3]. The so-obtained representative conformations are:

$1 \rightarrow$ 3T10 [199], loop out, AMPPCP-bound
$2 \rightarrow$ 3T0H [199], loop in, apo conformation
$3 \rightarrow$ 4AWQ [220], holo, bound to tropane derived inhibitor
$4 \rightarrow$ 6EYB [200], semi-helix
$5 \rightarrow$ 2YK9 [221], helix, bound to tricyclic imidazopyridine
$6 \rightarrow$ 3B28 [222], holo, bound to inhibitor CH5015765

For the remainder of the paper, each state is simply referred to by its associated number. Figure 3.3 illustrates the shape of the flexible loop of interest for each state. Note that here, we have reported these states as obtained from PDB. However, the ligand of each conformation presented above was removed before any simulations. All MD described and

[2] Protonation is the adding of a hydrogen cation to an atom or ion

[3] In X-ray crystallography, resolution is the smallest distance between crystal lattice planes that is resolved. It is thus a measure of the quality of the obtained crystal structure.

analyzed in this chapter is thus of unbound NTD. It is also important to recall here that the states were defined based on the loop of interest alone (residue $105-114$), and are thus not necessarily directly representative of the states of the whole HSP90 protein. Indeed, it has been observed (see for example [204]) that the conformations of the lid (and thus of the loop of interest) are not state specific, and that, for example, the energy landscape of the NTD in apo state contains different conformations of the lid and of the loop of interest.

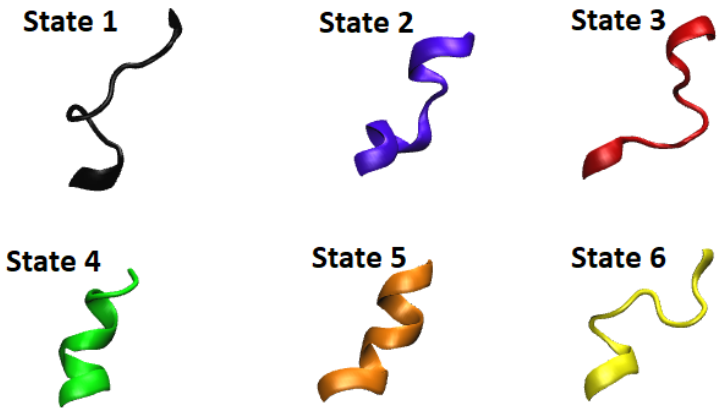


Figure 3.3: Structurally diverse conformations of the loop of interest corresponding to the identified states of the NTD.

3.2.2 Dataset generation

3.2.2.A Preparing the structures

For each of the 6 states, the protein structures were specifically prepared to have the exact same number and naming of atoms (3258 atoms), so that comparisons and structural alignments can be performed between different conformations. All simulations described in this section were performed with Gromacs [223], using the AMBER99SB-ILDN [171] all-atom force field for the description of the protein and the TIP4P [174] water model. All proteins were solvated into a cubic box with an 11 Å buffer around the protein to ensure a sufficient minimum separation of the protein from its periodic images. Na+ counterions were randomly placed in the system to neutralize the total charge while 0.15 M of NaCl salt was also added to resemble physiological conditions.

Before any production runs were performed, an energy minimization and a 5-step restrained MD relaxation protocol was followed, all run under constant pressure, temperature, and number of particles (NPT ensemble). The restraint MD was performed as follows: first, all heavy atoms of the system (i.e. all but hydrogen atoms) were restrained using a force constant of 1000 kJ/mol·nm^2 for 100 ps. Then, only the protein atoms were restrained using a force constant of 1000 kJ/mol·nm^2 for 100 ps. Two more 100 ps restrained MD with the same force constant were performed with restraints only on backbone atoms and then only on C_α carbons. Finally, a flat-bottomed restraint [4] was applied on the C_α carbons, with a

[4] Flat-bottomed restraint is used to restrain particles within a region of the simulation volume.

force constant of 500 kJ/mol·nm^2, starting above 2 Å from the reference structure and for 400 ps.

3.2.2.B Production runs

All production runs were performed in NPT, with a timestep of 2 fs. These runs consist, for each of the 6 states, of 10 independent trajectories of 20 ns each, started from the same configuration within that state, specifically the final conformation obtained from the preproduction restraint MD described in Section 3.2.2.A. Configurations were saved every 2500 steps. Figure 3.4 shows for each state the mean and standard deviation of the RMSDs over the 10 trajectories with respect to their initial conformation. For all the simulations, the RMSD stays below 0.2 nm, with the exception of one simulation started from state 3 (not shown individually here). This trajectory was subsequently visualized in VMD [224] and no abnormal behavior in the structure of the protein was noted.

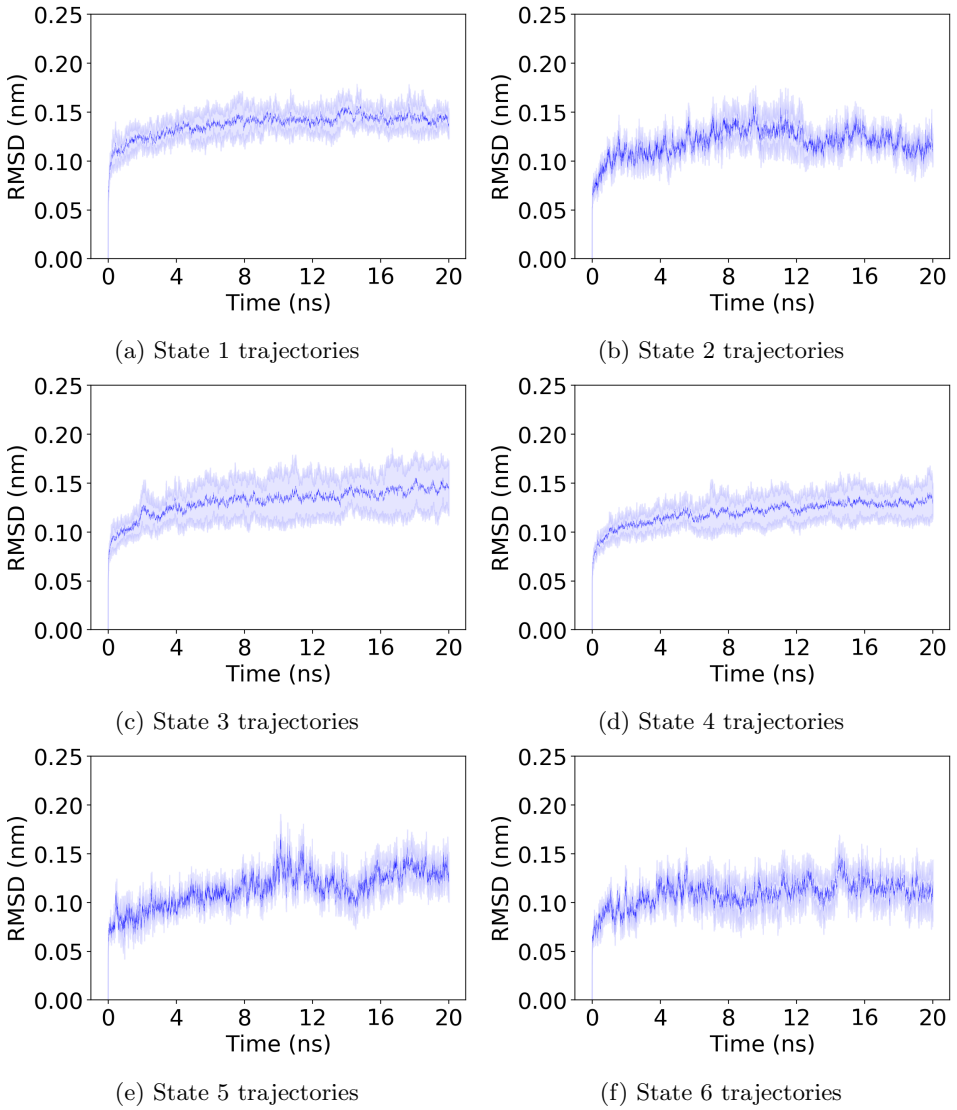
The training dataset used in this work is the concatenation of these 60 independent short MD trajectories. All configurations of this dataset are aligned to the same reference structure, which is the starting conformation of state 1. Only the 207 Cα carbons of the molecule are used in the input representation to avoid any noise created by the flexibility of the protein side chains. Our training dataset is thus composed of a total of $n = 240,000$ observations (configurations) of $D = 3 \times 207 = 621$ coordinates.

It was observed during the simulations described above that while states 4 and 5 were separated in the initial clustering of the different crystal structures of HSP90, state 4 spontaneously and invariably transitions to state 5 (i.e to a structure that is classified as state 5 using the dihedral clustering procedure) during MD simulations. This prompts us to not consider state 4 as a separate state when analyzing our MD simulations. For the remainder of this chapter, the numbering of states is unchanged, the number 4 being simply removed.

3.2.3 Autoencoder architecture

The autoencoder architecture is determined by setting the number, type and size of the layers. By definition, the input and output layers' size is equal to the dataset dimensionality, i.e. $D = 621$ as mentioned in Section 3.2.2.B. Similarly to the autoencoders used in Chapter 2, we use a symmetric architecture, and only fully connected layers (i.e. each neuron of each layer is connected to all the neurons of the previous layer). The number and size of the hidden layers determine the complexity of the model, and thus of the learned collective variables. Generally, when layers are added, more complex representations can be modeled by the autoencoder. However, it should be taken into account that more layers also require a larger dataset for training and can lead to overfitting. Moreover, from a practical point of view, the purpose of the autoencoder collective variable is to run biased sampling. The run time of the biased sampling simulation dramatically increases with more complex collective variables.

All of these points should be considered when selecting the autoencoder architecture. Section 3.2.3.A describes the training setup for all the autoencoders used in this work. In Section 3.2.3.B, we consider two different architectures to check whether adding a layer improves the learned model. The size of the layers is chosen so as to gradually reduce the dimensionality from input to bottleneck. However, the size of the bottleneck layer itself determines the dimensionality of the learned CV, and is thus selected carefully as described in Section 3.2.3.C.



(a) State 1 trajectories

(b) State 2 trajectories

(c) State 3 trajectories

(d) State 4 trajectories

(e) State 5 trajectories

(f) State 6 trajectories

Figure 3.4: Alpha carbon RMSDs mean and standard deviation over the 10 trajectories computed for each state.

3.2.3.A Training

The autoencoders are constructed and trained using the `Keras` [161] library in Python. All autoencoders are trained on 75% of the dataset, leaving 25% for validation. The learning rate used is $\eta = 10^{-4}$ with `Adam` optimization. We use a batch size of 1000 samples, and run training for a maximum of 1000 epochs. Early stopping of the training, i.e. before the 1000 epochs are achieved, is applied when the validation loss does not improve for 40 consecutive epochs. This monitoring of the validation loss is implemented in order to avoid overfitting the model to the training dataset.

3.2.3.B Number of hidden layers

The autoencoder should not contain too many layers because the encoder will later be used as the CV in a biased sampling procedure. By running test simulations using encoders with an increasing number of layers, we determined that having an encoder with more than 2 hidden layers in addition to the input and bottleneck layers is not feasible from a practical point of view, because the run time of the corresponding biased simulations is too high: on the machine used for the simulations performed in this work, and using an encoder of 3 hidden layers in addition to the bottleneck and input layers, one day is needed to run an approximate 0.5 ns of simulation. To assess the necessity of using two hidden layers instead of one in the encoder, we compare the performance of two different autoencoder architectures:

- Structure S_1, where the encoder contains one hidden layer between the input and bottleneck layers: Input (621) $\rightarrow$ Hidden 1 (100) $\rightarrow$ Bottleneck (k) $\rightarrow$ Hidden 2 (100) $\rightarrow$ Output (621).
- Structure S_2, the encoder contains two hidden layers: Input (621) $\rightarrow$ Hidden 1 (150) $\rightarrow$ Hidden 2 (40) $\rightarrow$ Bottleneck (k) $\rightarrow$ Hidden 3 (40) $\rightarrow$ Hidden 4 (150) $\rightarrow$ Output (621).

The numbers between brackets correspond to the size (i.e. number of neurons) of each layer. The parameter k is thus the dimensionality of the bottleneck layer (the final layer of the encoder, i.e. the CV) and will be determined in the next section. In the tests described in this section, we use different values of k, ranging from 1 to 10. For each value of k, two autoencoders with structures S_1 and S_2 are trained on the dataset described in Section 3.2.2. To compare the two autoencoders, we plot the evolution of their training and validation losses throughout training. While this was done for all values of k from 1 to 10, for clarity we only show plots for $k = 1$, $k = 5$ and $k = 10$ in Figure 3.5. The same plots with other values of k have shown similar results. It can be observed from Figure 3.5 that the training of the autoencoders with structure S_1 shows more stability, i.e. the evolution of the validation and training losses is approximately the same. Conversely, structure S_2 autoencoders (apart from the $k = 10$ autoencoder) overfit after ~ 100 epochs. Overfitting is however already handled by the early stopping procedure and is thus not an issue. For each k, the optimal structure S_1 model, i.e. the model with the optimal validation loss, reaches approximately the same validation and training loss as the optimal S_2 model. In particular, for $k = 1$, it can be observed that S_2 seems to outperform S_1 on the training loss, but the corresponding validation loss evolution shows that this actually corresponds to the S_2 model overfitting. The only advantage of structure S_2 is that it finishes training in fewer epochs. However, because S_2 represents larger models, one training epoch takes longer to complete, meaning that this lower number of epochs does not necessarily translate to faster convergence in wall-clock time. More importantly, the most time consuming part of our procedure is by several orders of magnitude the biased simulations, which run approximately 20 times faster with an encoder CV from structure S_1 than one from structure S_2. We thus choose structure S_1 for the autoencoders used in this work.

3.2.3.C Bottleneck layer size

Now that we have determined the number of layers of our model, and the size of the hidden layers, the optimal dimensionality k of the bottleneck layer should be selected. Here, we are looking for the minimal number of variables which are sufficient to represent a maximal

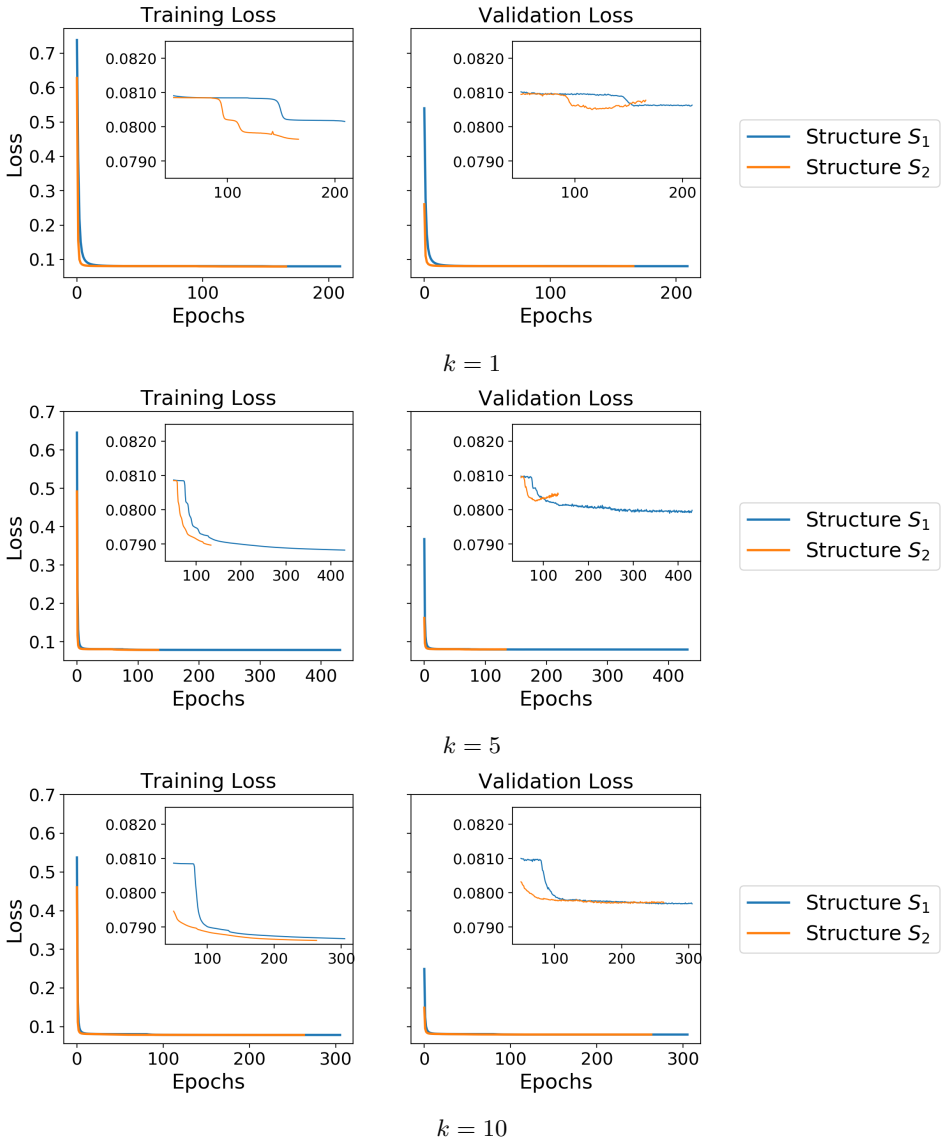


Figure 3.5: Evolution of training and validation losses for two different autoencoder structures and three different values of the bottleneck layer size k.

portion of the patterns and features of the conformational space of the system of interest. Numerically, this can be translated as a trade-off between the subspace dimensionality and the amount of data variance covered by that subspace. In the case of PCA for example, each eigenvalue represents the variance explained by its corresponding principal component. The principal components with the highest eigenvalues are thus kept. More specifically, the optimal dimension is chosen as the number of eigenvalues (ranked in decreasing order) for which the spectral gap (distance to the first eigenvalue) is lower than a certain threshold.

The advantage of PCA is that a single run is sufficient to determine the optimal dimension, and it provides direct quantities to help determine this dimension: the eigenvalues. This is not the case for autoencoders. We instead use the value of the fraction of variance explained (FVE) to determine the dimensionality k, as was done for alanine dipeptide in Chapter 2 (see Remark 4 and Appendix 2.7.4.B). More precisely, a set of 10 autoencoders, with bottleneck layer dimensions ranging from $k = 1$ to $k = 10$, are all trained on the dataset. Then, in order to compare these 10 models, we compute their corresponding FVEs:

$$\mathrm{FVE}(k) = 1 - \frac{\sum_{i=1}^{n} \|x^i - \phi_k(x^i)\|^2}{\sum_{i=1}^{n} \|x^i - \bar{x}\|^2},$$

where x^i are the data samples, $\bar{x} = \frac{1}{n}\sum_{i=1}^{n} x^i$, and $\phi_k(x^i)$ the output by the autoencoder with bottleneck dimension k of the input x^i. The optimal dimension k^* corresponds to a plateau or "knee" in the FVE curve (i.e. FVE plotted against bottleneck dimensions), meaning no considerable improvement in the reconstructed output is obtained by adding another dimension to the bottleneck manifold. Alternatively, the training or validation loss achieved by each autoencoder can also be plotted. Figure 3.6 shows both the training and validation loss curves, as well as the FVE curve computed over the training and validation sets, for k varying from 1 to 10. Both plots in Figure 3.6 show a plateau at $k = 5$ CV coordinates. However, the FVE (respectively loss) values do not increase (respectively decrease) by a significant amount as the bottleneck size k increases, which means that increasing the CV dimensionality does not significantly improve the reconstructed output. The additional dimensions of the CV evidently learn directions that are of much lower variance compared to the initial 1 dimensional bottleneck. A possible explanation is that these directions correspond to learning noise in the training data. However, because the validation curves are similar to the training curves, this hypothesis can be discarded. Additionally, plotting the 1 dimensional bottleneck encoder CV over the training dataset shows that this direction alone is actually not able to differentiate between all the identified states of HSP90. We therefore argue that despite their relatively small variance, the additional dimensions may still be of importance. We thus select the optimal dimensionality as $k^* = 5$ as indicated by the plateau in the plotted curves.

3.2.3.D Final autoencoder structure

In Figure 3.7, we present the final autoencoder structure used in this work. The autoencoder is composed of 4 layers with the following sizes: Input (621) $\rightarrow$ Hidden 1 (100) $\rightarrow$ Bottleneck ($k = 5$) $\rightarrow$ Hidden 2 (100) $\rightarrow$ Output (621). The activation function used for all layers is hyperbolic tangent.

3.2.4 Molecular dynamics simulations

3.2.4.A Simulation setup and parameters

Apart from the simulations described in Section 3.2.2 which served to generate the training dataset, all biased and unbiased simulations subsequently mentioned were run using

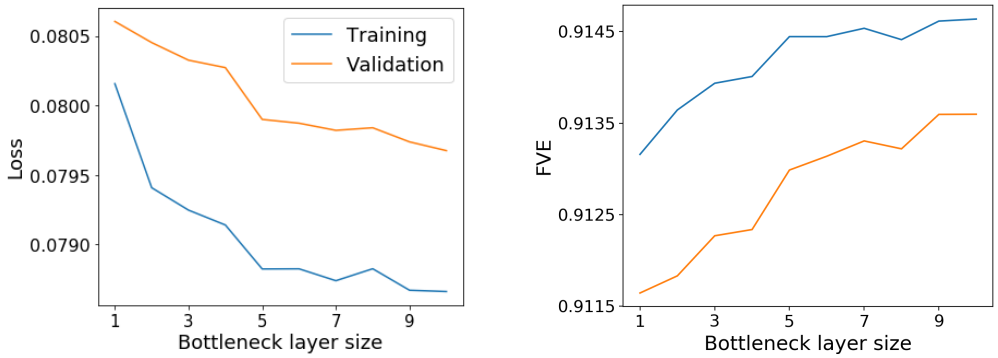


Figure 3.6: Left. Final training (blue) and validation (orange) loss obtained for each model. Right. Fraction of variance explained, over the training (blue), and validation (orange) dataset.

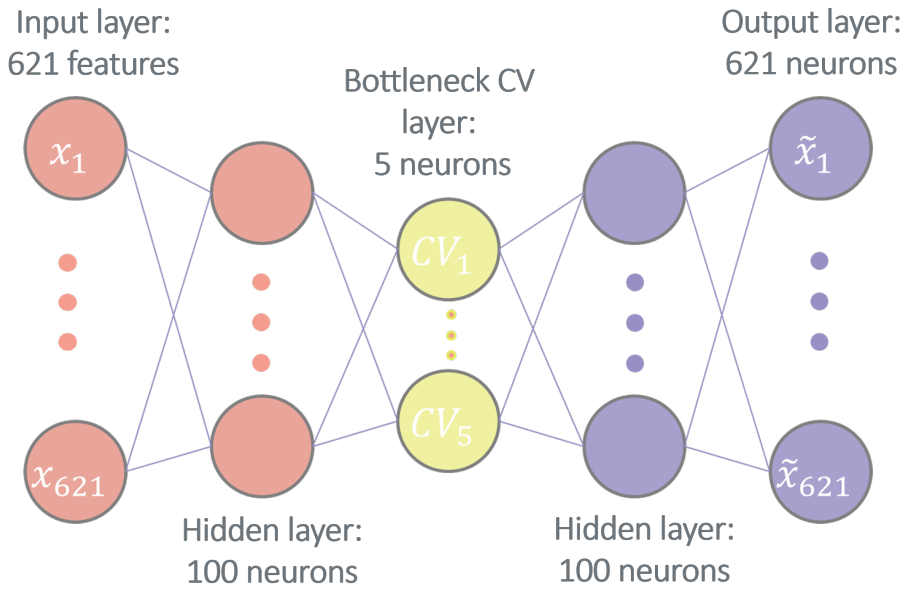


Figure 3.7: Architecture of the autoencoders used in this work.

OpenMM [163] under its Python API. The starting conformation for each state is the the last conformation after the position restraint MD described in Section 3.2.2. The same forcefield (Amber99SB-ILDN) and water model (TIP4P) were used, with a cubic box of 12 Å margin around the protein. Simulations were run in the NVT ensemble, with a Langevin integrator, a collision rate of 1 ps^{-1}, a timestep of 2 fs, and temperature $T = 300$ K. All biased simulations were performed using the extended system adaptive biasing force algorithm implemented in `PLUMED` [164,165]. Each coordinate of the CV was discretized into 50 bins. All other parameters of the ABF algorithm were kept to their default values in `PLUMED` (as was

done for alanine dipeptide in Chapter 2).

3.2.4.B Analyzing the simulations

MD trajectories generated in this work are initially analyzed to ensure that the system remains stable throughout the simulation. For this, the trajectories are visualized in VMD [224], and several quantities (energy, temperature, density, whole protein RMSD, etc) are plotted to ensure that their values are stable in average. Additionally, the trajectories are also analyzed to define the system's state along the trajectory and determine transitions. We consider three indicators for determining the state of a conformation. We next describe each of these analysis tools.

RMSD analysis. All RMSD computations are done using the `mdtraj` library in Python [225]. Only the α carbons are used in the RMSD computations. For each trajectory, RMSDs are computed with respect to each of the 5 states, over the whole protein, and over the 10 residues forming the flexible loop of interest ($105 - 114$). Additional details and results are given in Section 3.3.2.B.

Dihedral angles analysis. The dihedral angles of the residues forming the flexible loop are computed post simulation using `mdtraj`. The dihedral analysis uses the initial dihedral clustering model which was used to define the states. The clustering distances, i.e. the distance of each conformation to the centers of the identified clusters, are used on the unbiased trajectories to delimit the states/clusters from a dynamical (rather than crystal structure based) viewpoint. This makes it possible to detect transitions in the biased trajectories by computing these same distances. Details of the dihedral angle analysis and results are provided in Section 3.3.2.C.

Hydrogen bonds analysis. As an additional analysis step, the Hygrogen bonds (H-bonds) formed within the flexible loop are computed for each state using VMD. More precisely, the short unbiased trajectories are used to compute the frequency of H-bonds for each state. These computations are used to determine a set of the H-bonds specific to each state, making it possible to define a H-bonds-based fingerprint for each state, which can then be used to spot (or rather confirm) transitions in subsequent biased simulations.

3.3 Results

This section presents the results obtained in this work. First, the autoencoder learned collective variable is presented in Section 3.3.1. Second, a specific protocol for identifying transitions in the biased simulations is presented in Section 3.3.2, and illustrated using one of the simulated runs, for which a transition is indeed confirmed. Several other simulations were run, some of which are presented in Section 3.3.2.E.

3.3.1 The autoencoder collective variable

Once the autoencoder is trained, the encoder is used as the learned collective variable. In this section, this CV is analysed and its efficiency at describing and differentiating the 5 determined states of HSP90 is assessed.

3.3.1.A Distribution of the 5 dimensional CV

We start by plotting the values of each of the 5 CV coordinates, which we refer to by CV_i for $1 \leq i \leq 5$. For this, we separate our training dataset into the 5 conformational states to observe how each coordinate of the CV varies from one state to another. The results are shown in Figure 3.8.

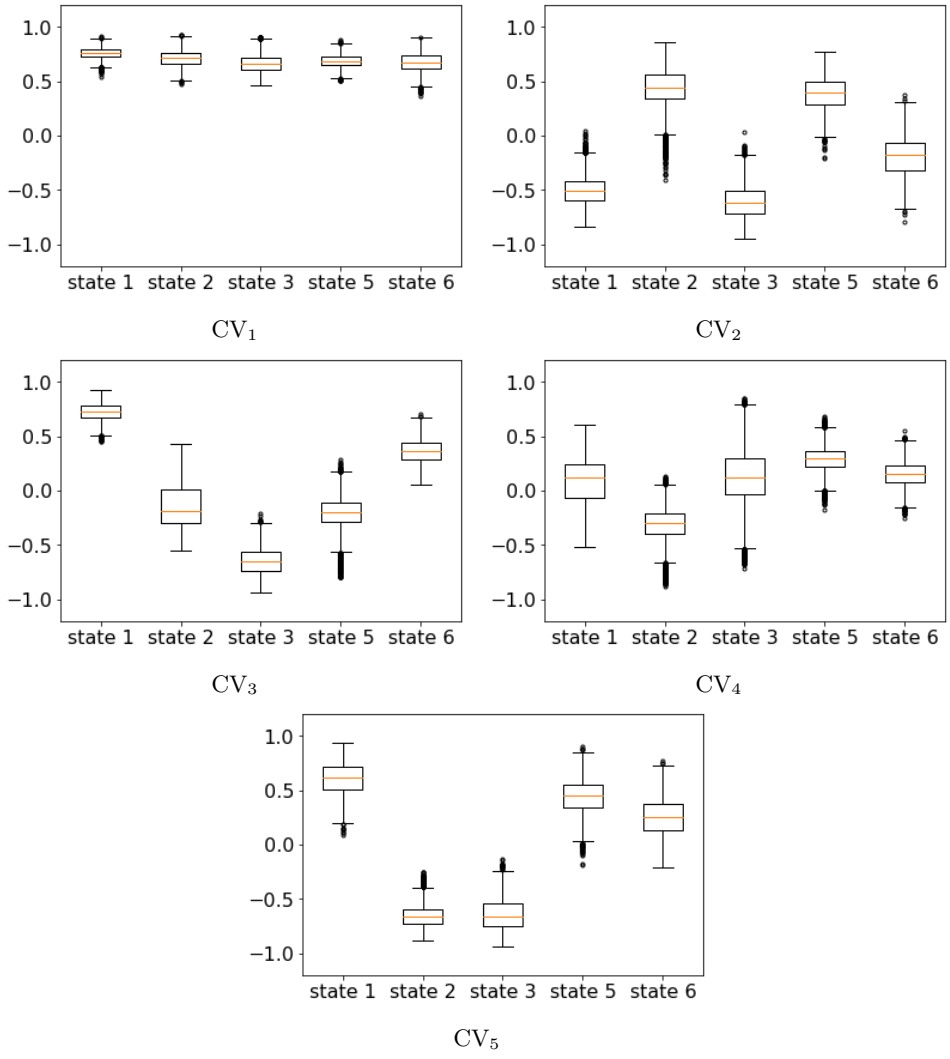


Figure 3.8: Boxplots of the CV variation per state.

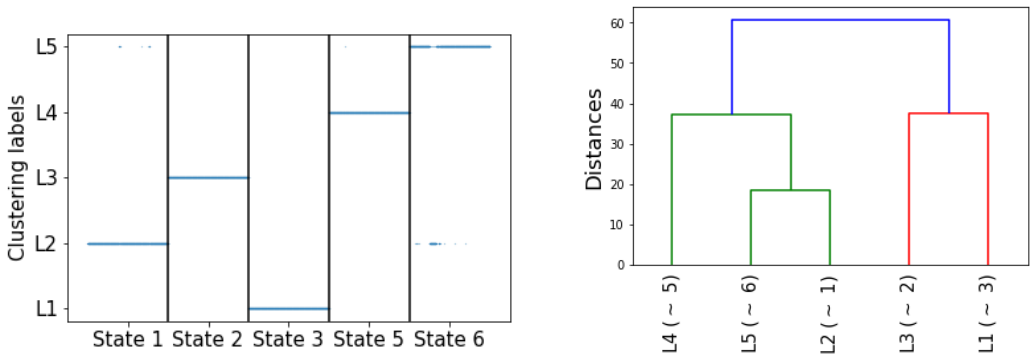
The results indicate that CV_1 does not enable any separation between the 5 identified states of HSP90. Next, the mean values of CV_4 over each state are quite comparable and range within $\sim [-0.25, 0.25]$. This coordinate therefore does not provide a clear separation between states either, contrarily to coordinates CV_2, CV_3 and CV_5.

3.3.1.B Further reduction of the CV dimensionality

Our final goal is to use the autoencoder CV to perform free energy biasing in order to sample transitions between the states of HSP90. In theory, the 5-dimensional CV could be used directly. In practice, the simultaneous biasing in 5 directions and estimation of the free energy over this five dimensional space is prohibitively computationally expensive. Indeed, because adaptive biasing methods such as ABF compute the free energy along the d-dimensional CV, their efficiency significantly decreases for a large value of d. In practice, ABF typically requires a low CV dimension of 1 to 3. Here, we use the boxplots illustrated in Figure 3.8 above, as well as additional analyses and observations described below, to reduce our CV dimensionality to 2, by selecting 2 of the 5 coordinates learned by the autoencoder.

As concluded in Section 3.3.1.A after examining the CV variation per state in Figure 3.8, coordinates CV_1 and CV_4 fail to differentiate significantly between the 5 states and are thus not expected to be helpful for driving transitions among these states. They are therefore eliminated.

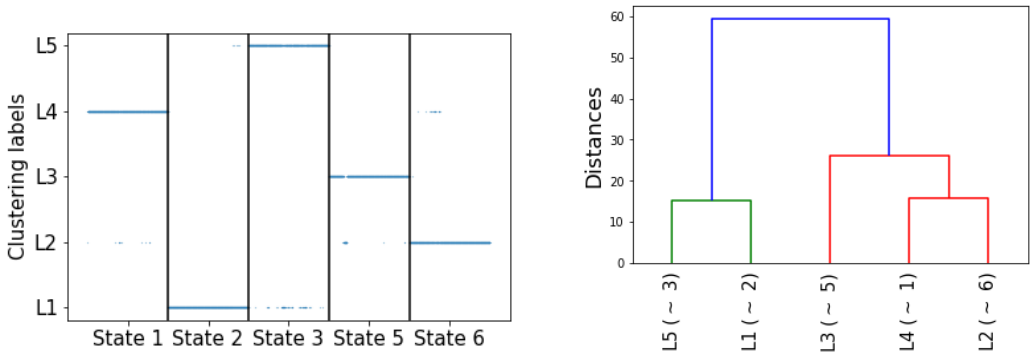
Next, to discriminate between the remaining three directions, we perform hierarchical clustering over each pair of CVs, and select the CV pair whose clustering best separates the 5 states. For this, we use the agglomerative clustering method, with Ward's minimum variance criterion to merge clusters: The method starts with as many clusters as the number of datapoints in our dataset, and successively merges clusters by minimizing the variance of the newly merged clusters. First, the agglomerative clustering is performed over the 5-dimensional CV space. Figure 3.9 illustrates the clusters obtained when stopping the hierarchical clustering at 5 clusters, as well as a dendrogram showing how these 5 clusters are grouped according to their distances from one another. Interestingly, the 5 states have a clear visual correspondence to the 5 clusters (Figure 3.9a). The dendrogram in Figure 3.9b shows that clusters L5 and L2 (i.e states 1 and 6) are the most similar (smallest distance) compared to the the remaining states.



(a) (b)

Figure 3.9: (a) Clustering using 5 clusters over the 5-dimensional CV space. Vertical black lines delimit the points inside each of the 5 conformational states. (b) Dendogram showing how the 5 clusters are iteratively merged into one. Each cluster is represented by its label number (L1 to L5) and the corresponding conformational state is indicated between brackets. The y-axis corresponds to the distances between pairs of clusters.

The same method is now used on all three CV pairs (CV_2, CV_3), (CV_2, CV_5) and (CV_3, CV_5). The optimal clustering is selected as the one which separates the 5 states with the highest accuracy, i.e. the lowest number of mislabeled points. The results indicate that the optimal clustering is obtained when using the pair of coordinates (CV_3, CV_5). For the remainder of the chapter, we refer to this coordinate pair as the autoencoder CV, but keep the indexing CV_3, and CV_5. Figure 3.10 shows the results of the agglomerative clustering applied to (CV_3, CV_5).



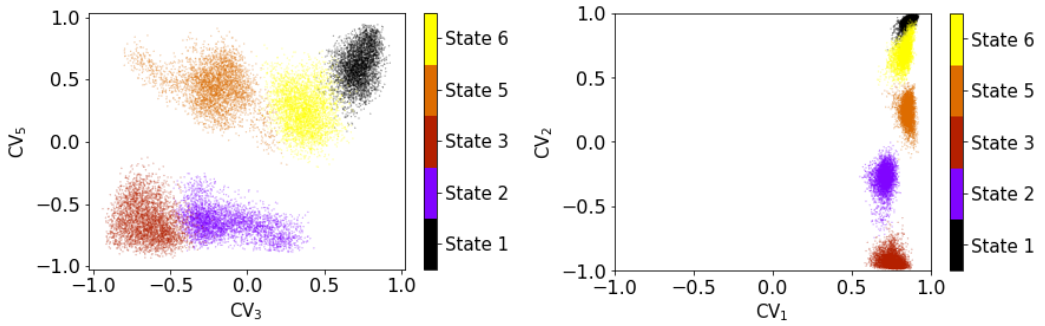
(a) (b)

Figure 3.10: Agglomerative clustering over the 2-dimensional CV space (CV_3, CV_5), instead of the 5 dimensional CV as in Figure 3.9

To gain a clearer idea of the obtained CV space, we plot in Figure 3.11a the autoencoder CV over the dataset of short unbiased trajectories started from each state (the samples from each state are colored using a different color). The plot shows that the autoencoder CV indeed differentiates between the five conformational states, making it a possibly good choice for running a free energy biasing procedure.

Remark 6. *Let us emphasize that while the final CV dimension is* 2*, it is necessary, or at least more optimal, to obtain this* 2*-dimensional CV from the* 5*-bottleneck autoencoder rather than using a* 2*-bottleneck autoencoder directly. We justify this choice by the fact that as an unsupervised model trained for reconstruction of the input, the autoencoder will always have to learn some features which do not necessarily distinguish between the metastable states (as* CV_1 *in Figure 3.8 for instance), but may nonetheless be important for data reconstruction (e.g. features representing a large number of residues of the protein). This is especially true for large proteins whose various states are sometimes only distinguished by motions in relatively small functionally important regions. Training an autoencoder with a large bottleneck size* $k^* = 5$ *makes it possible to learn and separate features representing the actual motions of interest from the features representing high variance directions that are unrelated to these motions. Then, handpicking the* $d < k$ *bottleneck dimensions of interest ensures more efficiency for biasing, compared to directly learning a CV using a* d*-dimensional bottleneck autoencoder.*

To illustrate our point, we show in Figure 3.11b the CV obtained from a 2*-dimensional bottleneck autoencoder. The first coordinate of this CV fails to distinguish between any states, the second coordinate does not make a clear separation between states* 1 *and* 6*. Agglomerative clustering applied to this CV also shows that these* 2 *states are not well separated.*



(a) (b)

Figure 3.11: Scatter plot of the autoencoder CV, each point is colored according to its corresponding state. Left: Coordinates 3 and 5 of the 5-dimensional bottleneck autoencoder. Right: The 2 dimensional bottleneck autoencoder.

3.3.2 Free energy biasing: identifying transitions

As concluded from the analysis of Section 3.3.1, we use the coordinates CV_3 and CV_5 to run ABF simulations using the framework described in Section 3.2.4.A. Multiple simulations were started from each of the 5 states. To analyze the simulations and identify possible transitions, we present in this section a protocol which starts with a first identification of possible transitions in Section 3.3.2.A based on the visited regions of the CV space. Then, the simulations are further analyzed to either confirm or dismiss the previously identified transitions. This is done by first looking into the RMSD evolution of the simulations in Section 3.3.2.B. Then, we present in Section 3.3.2.C an additional set of analysis based on the dihedral angles of the loop of interest of the NTD, i.e. the angles that served to cluster and define the states in the first place, as presented in Section 3.2.1. Finally, an additional analysis step, based on tracking the formation of hydrogen bonds (H-bonds) in the loop of interest, is presented in Section 3.3.2.D.

In this section, we detail the results obtained from only one of our simulations. Note however that results on other biased simulations, for which the same protocol was used to determine transitions, are provided in Section 3.3.2.E.

3.3.2.A A first identification of transitions

The biased simulation selected for illustration is 100ns long and is initiated from conformational state 1. A preliminary step to identify possible transitions in the trajectory is to look into the sampled values of the autoencoder CV, i.e. the biasing CV. These values give first pointers into which regions of the CV space, and thus possibly which distinct protein conformations, are visited during the simulation. Figure 3.12 shows the variations of the CV coordinates as well as the scatter plot of the visited regions. In this biased simulation, the distributions of the CV coordinates seem to indicate:

- A quick transition out of state 1. In fact, the simulation never seems to return to this state.
- A transition to conformational state 3.

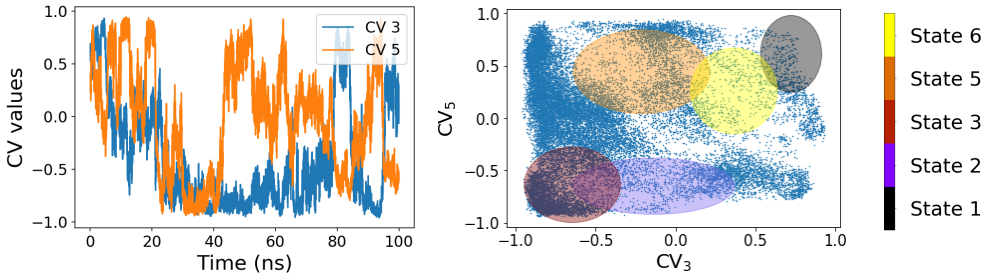


Figure 3.12: Results of the 100 ns biased trajectory initiated from state 1 using (CV_3, CV_5) for biasing. Left: Fluctuations of the two coordinates composing the biasing CV as a function of time. Right: Distribution of visited conformations over the CV space. Ellipses show approximate locations of the 5 conformational states as illustrated in Figure 3.11.

In order to confirm these initial observations, in particular the transition to state 3, additional analysis are performed.

3.3.2.B RMSD with respect to each conformational state

A first analysis is done by plotting the RMSD of the biased trajectory calculated with respect to the initial conformation (i.e. energy minimized, and position restrained conformation) of each of the 5 states, using only the Cα carbons of the specific loop of interest of the NTD. Figure 3.13 shows the obtained RMSD plots. The minimal RMSD is at the beginning of the simulation obtained with respect to the conformational state 1, i.e. the starting state, as expected. However, the minimal RMSD shifts to the one with respect to conformational state 3 after $\sim$ 30 ns of simulation time, until $\sim$ 55 ns of simulation time. These results are a first confirmation of the possible transition from state 1 to state 3 observed from the CV distribution.

3.3.2.C Dihedral angle analysis

The transitions between the 5 states can also be identified by looking into the dihedral angles of the binding site loop. These are the same dihedral angles used to cluster crystal structures of HSP90 into the 5 states as presented in Section 3.2.1, i.e the dihedrals of residues 105 to 114. We first use the clustering model itself on the biased trajectory to check whether any frames from the trajectory are clustered within state (i.e. cluster) 3. We then apply other methods, including dihedral PCA, for further analysis and comparison to unbiased trajectories.

Clustering the biased trajectory. The clustering model trained to obtain the 6 original states (as presented in Section 3.2.1) is used again here. More precisely, the distance of each frame of the biased trajectory to each of the 5 cluster centers corresponding to states 1, 2, 3, 5 and 6 is computed and plotted. Recall that distances to cluster 4 are not plotted as we have already discarded this state. As a point of comparison, the same distances are plotted for the unbiased dataset of trajectories started from each state. The obtained plots are illustrated in Figure 3.14.

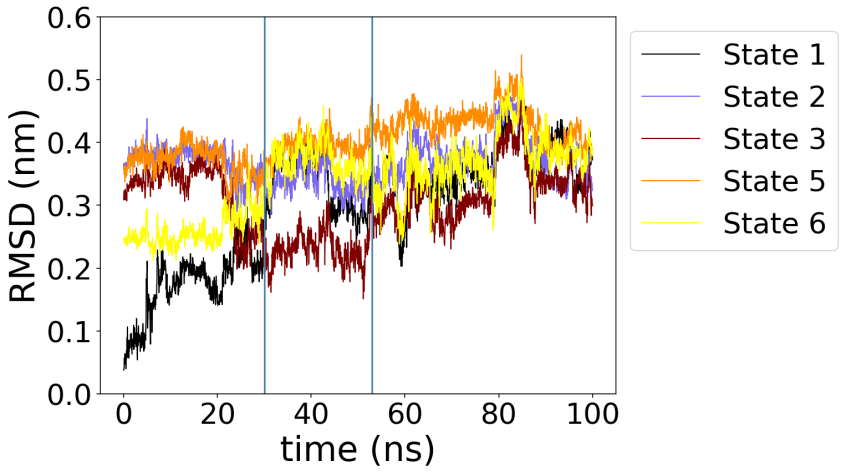


Figure 3.13: RMSD values of the biased trajectory with respect to each conformational state. Vertical blue lines delimit the suspected transition to state 3.

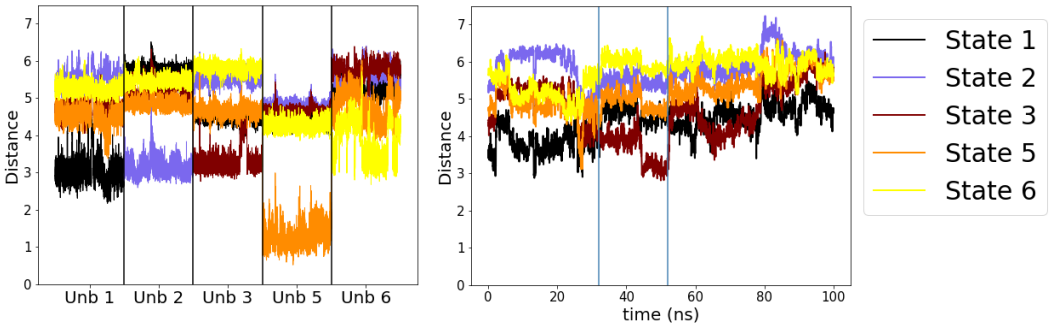


Figure 3.14: Distances to each of the 5 cluster centroids representing the states. Left: Unbiased trajectories started from each of the 5 states. Right: Biased trajectory. Vertical blue lines delimit the suspected transitions to state 3.

The results obtained on the unbiased trajectories show that to label a conformation as in state 3, its distance to the centroid of the corresponding cluster 3 should range between 2.8 and 4, and its distance to the remaining centroids should be higher than 4. The same applies for all other states but state 5, for which the distances are smaller (see left plot of Figure 3.14, column "Unb 5"). The results obtained on the biased trajectory show that the computed distances are compatible with a transition to state 3 between $\sim$ 30 ns and $\sim$ 55 ns, i.e. approximately the same time frame for which a transition was identified using the previous methods.

We also note a small time frame around 25 ns where the conformations are closest to the centroid of state 5 than to any other state. These sampled points are thus automatically classified as state 5 configurations. However, their distances to the centroid 5, ranging between 3 and 4, are actually higher than the typical distance observed in state 5 unbiased trajectories, and which ranges between 1 and 2. This indicates that these conformations

probably do not actually belong to state 5 (or any of the other states determined in the beginning of this work). Importantly, this misclassification emphasizes the importance of looking into the clustering distances, rather than simply trusting the cluster assignments.

Remark 7. *We note that these "pseudo state* 5*" conformations are sampled just before the part of the simulation identified as state* 3*, and may therefore belong to a transient state between* 1 *and* 3*.*

Dihedral PCA. Dihedral PCA can also be used to analyse possible transitions. Here, we use dihedral PCA on the 278 crystal structures to reduce the 20 dihedrals into 2 principal components. The obtained 2D space of the two principal components is plotted in the left side of Figure 3.15, where each point represents one of the crystal structures, colored according to its corresponding state/cluster. Note that we again discard state 4 from the plots. The PCA projection is then applied to the biased trajectory in the right side of Figure 3.15, where each point is colored according to their assumed state, determined as in the previous paragraph by the closest centroid from the clustering.

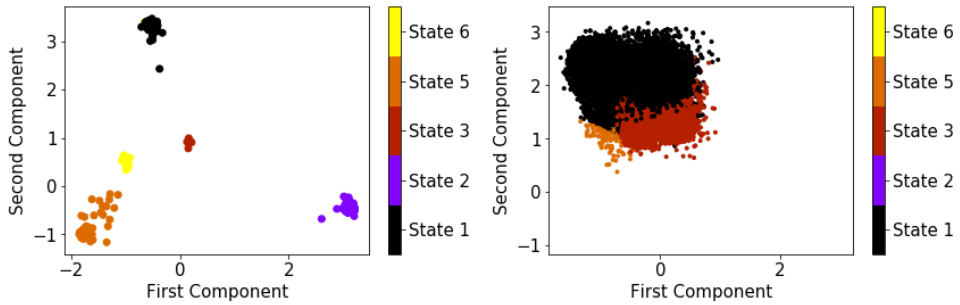
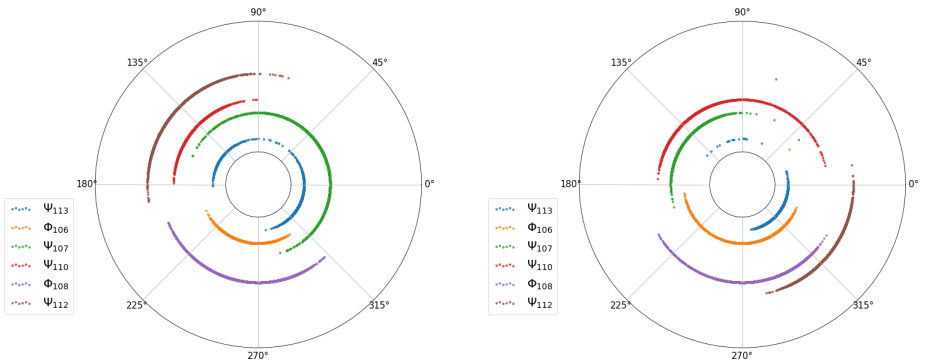


Figure 3.15: Dihedral PCA over the dihedral angles of residues 105 to 114. Left: Dihedral PCA of the crystal structure dihedrals. Points are colored accoridng to their states. Right: Dihedral PCA projection on the biased trajectory. Points are colored according to the cluster assigned using the dihedral clustering model (Figure 3.14)

Recall that the centroid distances in Figure 3.14 indicate that three states are presumably sampled: 1 (the starting state), 3 (the possible transition in the $30-55$ ns timeframe) and 5 (identified around time 25 ns), but with higher than normal centroid distances. We note from Figure 3.15 that while the points identified as states 1 or 3 are correctly located in the PC space, the conformations identified as state 5 are not. This reinforces our conclusion of the previous paragraph that state 5 is not actually visited during this biased simulations. Conversely, the obtained results corroborate the hypothesis of a state 3 transition. Additionally, the location of the pseudo state 5 points, i.e. somewhat between the state 1 and state 3 regions of the PC space, is consistent with the observation made in Remark 7, namely that these conformations may correspond to a transient state between 1 and 3.

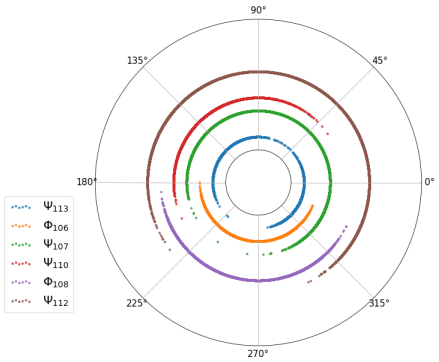
Dihedral fingerprint. As a final analysis of the biased simulation using the dihedral angles, we define a "fingerprint" of each state computed as the range of values taken by each dihedral angle in the loop, on each state. Because there are 10 residues and thus

20 dihedral angles to consider, the obtained fingerprint can be hard to analyze. Here, we use a LASSO classification model on the 278 crystal structures to select a subset of dihedrals that is sufficient to separate the different states. More precisely, we train a simple linear classification model (logistic regression) with ℓ_1-norm regularization. This so-called Lasso regularization ensures sparsity in the trained model, so that only relevant and non redundant features, i.e. dihedrals, are kept. The obtained model selected 6 dihedral angles, whose values can be used to characterize each state. Again, we seek to define states 1 and 3 through the values of their dihedral angles. Figure 3.16 shows the ranges of these 6 dihedral angle values on states 1 and 3, as well as the values visited during the biased simulation. The ranges of the dihedral angles sampled during the biased simulations can be seen as an interpolation between the typical values identified for states 1 and 3, indicating again that a transition between these states has occurred. This is for example illustrated by the values taken by dihedral angle Ψ_{112}, which is colored in brown in Figure 3.16. This angle ranges between 90° to 190° for state 1, and between 280° and 0° for state 3. Its values on the biased simulation range in $[290°, 0] \cup [0°, 200°]$ thus interpolating the values observed in both state 1 and state 3.



(a) Unbiased 1.

(b) Unbiased 3.



(c) Biased simulation.

Figure 3.16: Values of the 6 Lasso selected dihedral angles for (a) state 1, (b) state 3, and (c) the biased trajectory.

3.3.2.D Hydrogen bonds analysis

As a final step analysis of our biased simulation, we observe the formation or breaking of hydrogen bonds between the amino acids of the loop of interest. A hydrogen bond (H-bond) is an electrostatic force of attraction between two atoms:

- a hydrogen atom that is bound to an electronegative atom (usually one of the elements O, N or F), called the donor;
- another electronegative atom which has a lone pair of electrons, called the receiver.

Figure 3.17 shows an example of a H-bond. Structurally, a H-bond can be defined by

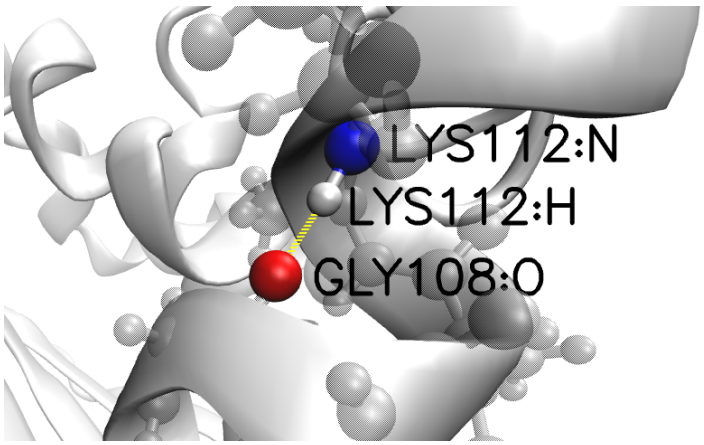


Figure 3.17: Example of a hydrogen bond (yellow dashed line) between donor atom N in residue LYS112 (blue) and acceptor O in residue GLY108 (red).

two important criteria: the bond donor-acceptor distance and the bond acceptor-donor-hydrogen angle, both of which should be smaller than a given threshold. Here, we choose an angle threshold of 25° and a distance threshold of 0.35 nm. These values were chosen as a compromise between the VMD and Gromacs default values.

The following procedure is applied for each of the 5 conformational states: Using the ten 20 ns trajectories of the initial training dataset, we define all hydrogen bonds that appear in the backbone of the protein between residues 100 and 120. A total of 70 hydrogen bonds appear in at least one of the 5 states, for at least one of the 10 short trajectories. For each H-bond h and each conformational state i, we determine a mean $m_{h,i}$ and standard deviation $\sigma_{h,i}$ of the occurrence (percentage of appearance) of each of these bonds (the mean and standard deviation are computed over the 10 trajectories we have). Then for each H-bond h, we compute a general mean m_h and standard deviation σ_h over the means $m_{h,i}$. To rule out bonds that appear similarly often in all states, or that appear very rarely in all bonds, we keep only H-bonds with high enough values of m_h and σ_h (i.e. such that: $m_h > 4\%$ and $\sigma_h > \frac{m_h}{4}$). This leaves us with 43 H-bonds, whose mean occurences and associated standard deviations are plotted in Figure 3.18

We now use these H-bonds occurences to characterize each state. Here, we focus on applying the H-bonds analysis to checking the transition from state 1 to state 3 in the 100 ns biased simulation we have worked on throughout Section 3.3. In order to target

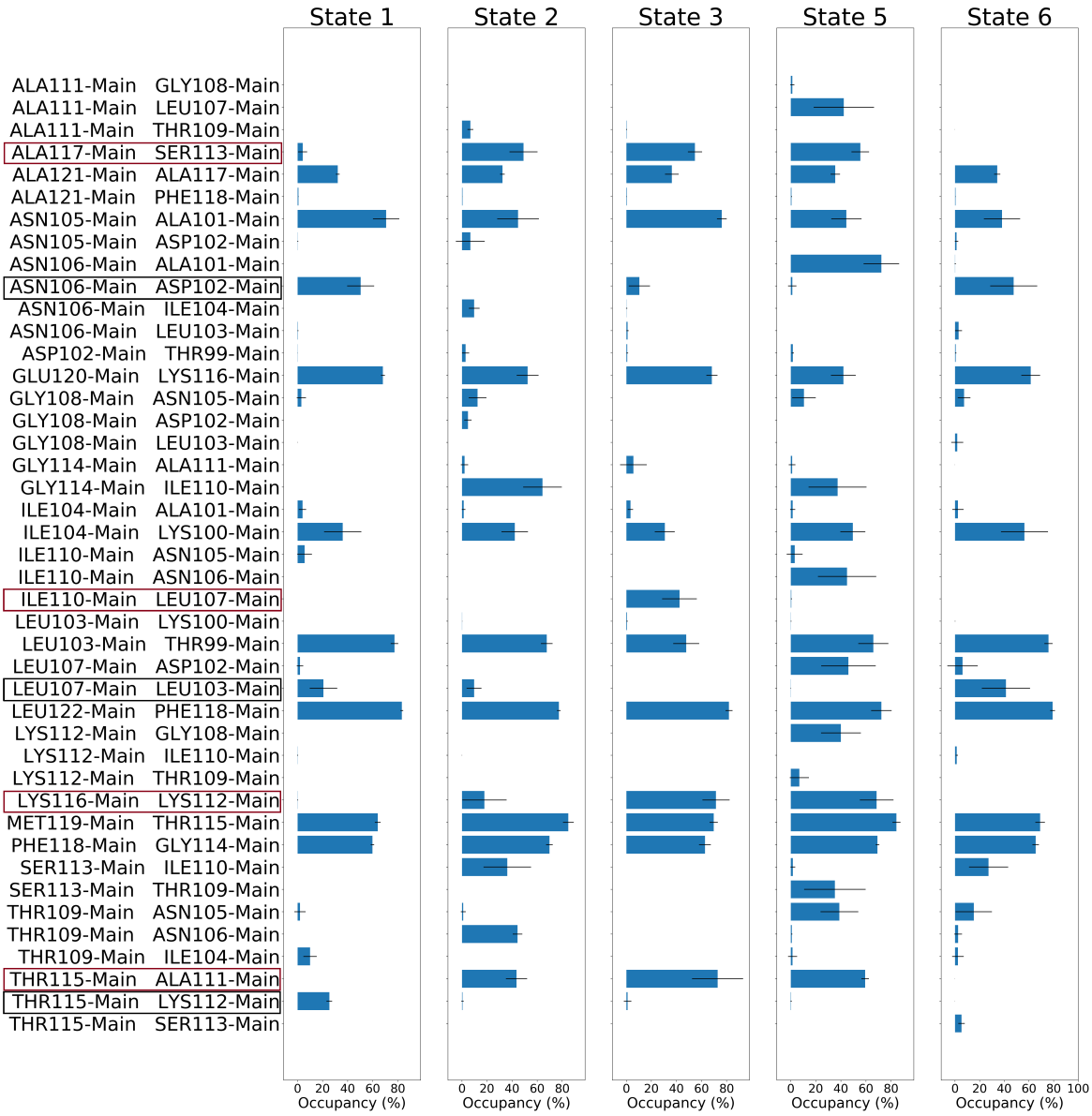


Figure 3.18: H-bonds occurence percentages for all states. State 1 (respectively state 3) H-bonds are framed in black (respectively maroon)

our analysis specifically on transitions between states 1 and 3, we focus on the following H-bonds:

- **LYS116-LYS112**: appears only in state 3 ($\sim 70\%$).
- **THR115-ALA111**: appears only in state 3 ($\sim 70\%$).
- **ILE110-LEU107**: appears only in state 3 ($\sim 40\%$).

- **THR115-LYS112**: appears only in state 1 ($\sim 30\%$).
- **LEU107-LEU103**: appears only in state 1 ($\sim 20\%$).
- **ALA117-SER113**: appears in state 3 ($\sim 50\%$) much more often than in state 1 ($< 5\%$).
- **ASN106-ASP102**: appears in state 1 ($\sim 50\%$) more often than in state 3 ($< 10\%$).

Now that we have narrowed everything down to these 7 H-bonds, we monitor their appearance (formation and breakage) over the biased trajectory, in order to check whether we can detect a transition from state 1 to state 3 using this information. In particular, our focus is on characterizing possible conformations belonging to state 3. The 4 H-bonds specific to this state are illustrated in Figure 3.19.

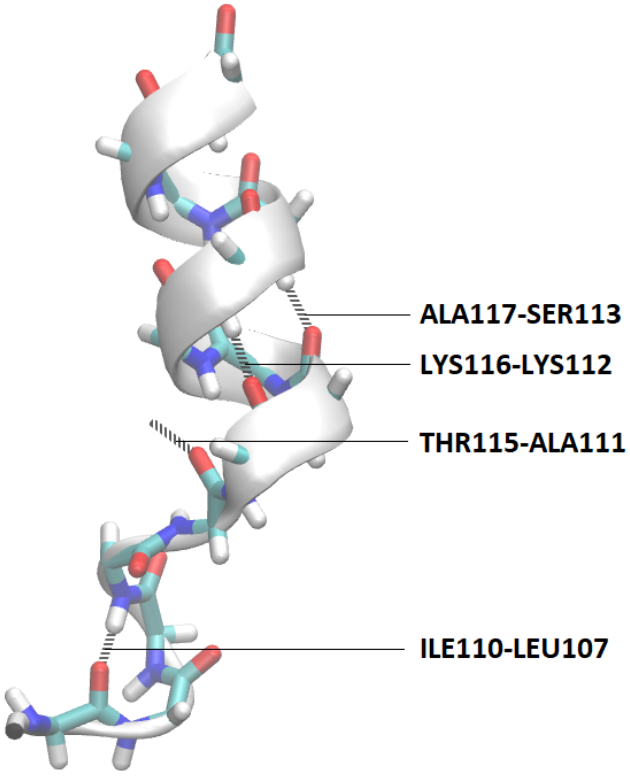
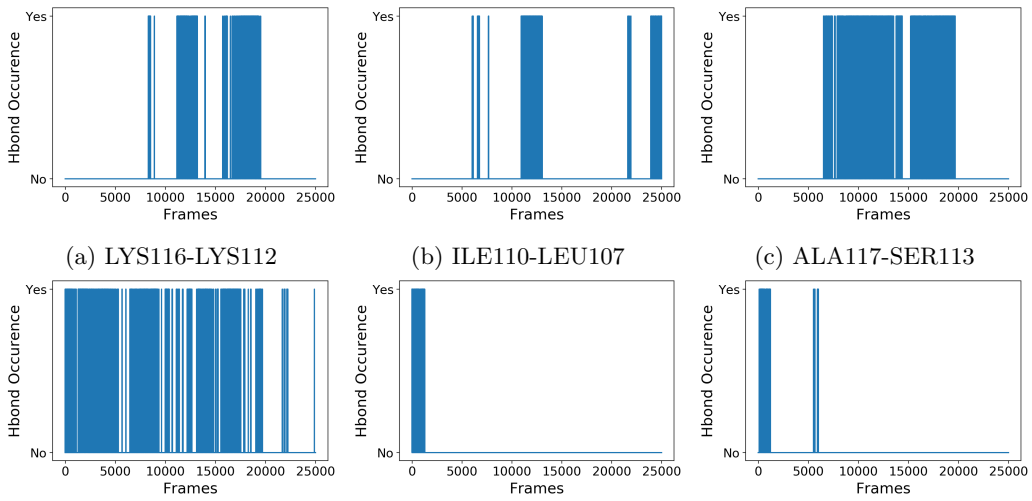


Figure 3.19: Four H-bonds determined as specific to state 3. Only main chain atoms are shown. The H-bond THR115-ALA111 is the only one involving a side chain atom (its hydrogen atom and donor atom, an oxygen atom of THR115, are side chain atoms and thus not shown).

We first note that unfortunately, the H-bond THR115-ALA111 almost does not appear in the biased trajectory (except at one single frame). This H-bond is formed by an oxygen atom in the backbone of ALA111 (receiver) and an oxygen atom in the side chain of THR115. It is the only one of the four state 3 identified H-bonds which includes a side chain atom. The

remaining H-bonds' occurrences in the biased simulation are plotted in Figure 3.20. The obtained results indicate that over the 4 H-bonds linked to state 3, 3 of them are formed during the biased simulation between 25 ns and 60 ns of simulation time. Moreover, of the 3 H-bonds that characterize state 1, only one keeps appearing throughout the whole the simulations, the other two no longer form after the 20th nanosecond of simulation. Overall, the results indicate that the protein exits state 1 and visits a conformational state that is closest to state 3 than any of the 5 other identified states.



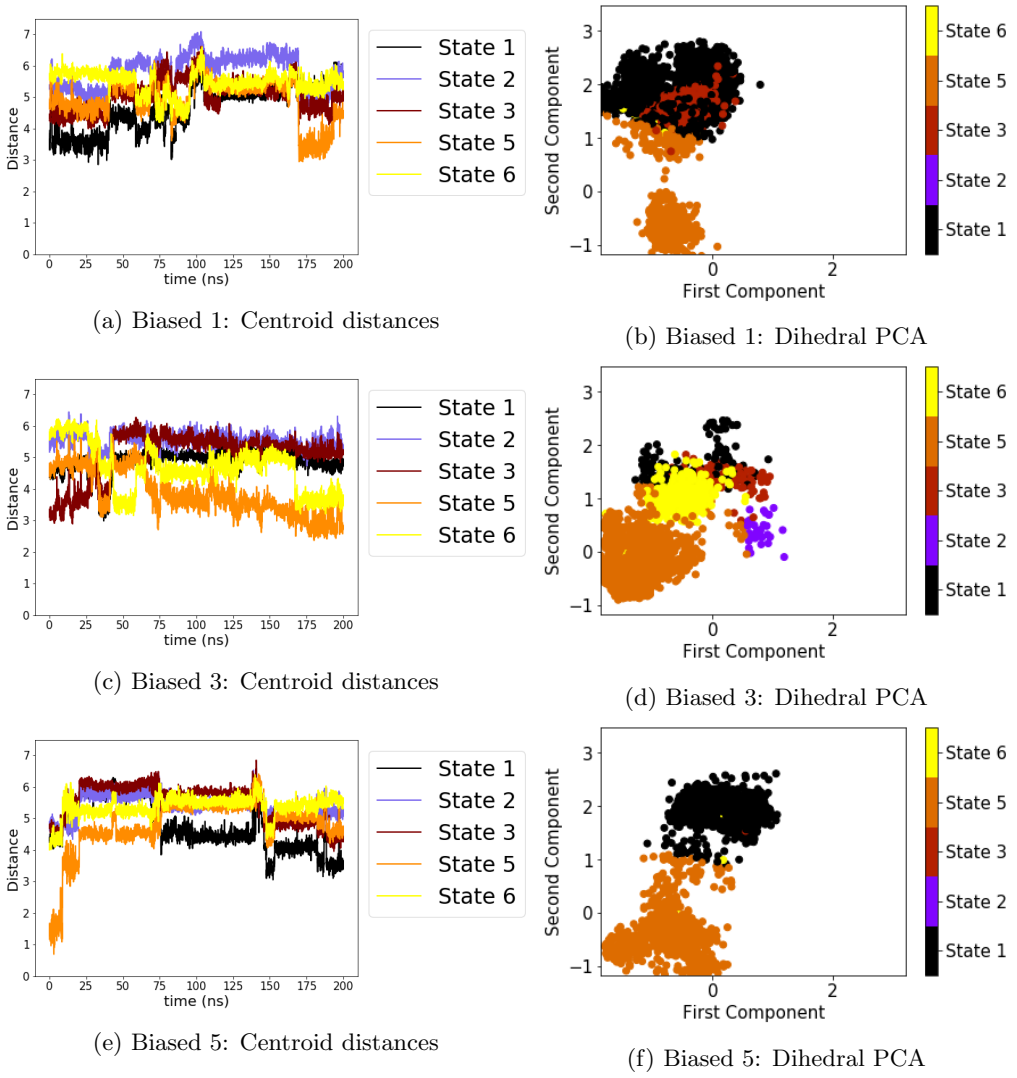
(a) LYS116-LYS112 (b) ILE110-LEU107 (c) ALA117-SER113

(d) THR115-LYS112 (e) LEU107-LEU103 (f) ASN106-ASP102

Figure 3.20: Occurence of the main H-bonds that distinguish conformation 1 and 3. Top: state 3 linked H-bonds. Bottom: state 1 linked H-bonds.

3.3.2.E Additional biased simulations

All the results presented in Section 3.3.2 concern one biased trajectory of 100 ns started from state 1. This simulation was chosen to illustrate the protocol we present here for confirming transitions. However, to ensure that the obtained results were not specific to this one trajectory, additional biased simulations were run and analyzed using the same protocol. Here, we show the results of the dihedral analysis described in Section 3.3.2.C obtained for some of these simulations. We focus on three simulations, which were started from states 1, 3 and 5 and were run for 200 ns each. These three simulations are shown in Figure 3.21 as examples of the biased simulations for which transitions were obtained. More generally, out of 15 biased simulations of 200 ns each started from the different states, 9 achieved identified transitions with high confidence, i.e. confirmed by the whole analysis protocol as was done for the biased simulation presented in Section 3.3.2. 3 of the simulations do not leave the initial state, and 3 leave the initial state but do not visit any other state defined within this work.



(a) Biased 1: Centroid distances

(b) Biased 1: Dihedral PCA

(c) Biased 3: Centroid distances

(d) Biased 3: Dihedral PCA

(e) Biased 5: Centroid distances

(f) Biased 5: Dihedral PCA

Figure 3.21: Dihedral angle analysis of three additional biased simulations. Left: Distances to the centroids of the dihedral angle clustering. Right: Dihedral PCA projections.

3.4 Conclusions and future work

3.4.1 Discussion: machine learning based analysis of HSP90 states and collective variables

The results described in Section 3.3 make it possible to conclude with regard to the initial goals we set in the beginning of this Chapter (Section 3.1). The conformational states of HSP90 were carefully defined using the clustering of a large number of PDB structures. The obtained states are visually different from each other, and display some level of metastability as showcased by the RMSD evolution of unbiased simulations in Figures 3.4 and 3.22. The

autoencoder trained on the dataset of short unbiased simulations allows the construction of a low dimensional CV able to clearly differentiate between the conformational states. Additionally, the results obtained on the 100 ns biased simulation in Section 3.3.2 and the additional biased simulations mentioned in Section 3.3.2.E show that this autoencoder CV can be used to efficiently sample transition paths among the various states. Running biased simulations using more classical choices of CVs, such as some dihedral angles of the loop of interest, did not yield such transitions.

Of course, training the autoencoder on all 5 states was central to obtaining this CV. The entire analysis is in fact based on the definition of the 5 states, which was done using the clustering of the RCSB crystal structures using the flexible loop. The results are thus all guided by this classification choice. While we argue that our classification of the states is well defined, many other choices could have been applied, for example by basing the clustering on the binding site itself rather than the flexible loop adjacent to it. The work presented in this chapter falls within a supervised setting and is in this way different from the results of the unsupervised method, AE-ABF (or more generally FEBILAE) presented in Chapter 2, where the state definitions did not impact the learning of the collective variable. Finally, it is important to recall that the results may also depend on other choices in the machine learning and molecular setup, including the autoencoder structure and training parameters, the MD simulation parameters, namely the choice of the forcefield.

3.4.2 Uncovering transition paths between the conformations of the unbound NTD

The obtained states differ by the conformation of the flexible loop of interest, originally due to their binding (or lack thereof) with different ligands. We chose in this work to simulate HSP90 unbound, i.e. by removing the ligand from the bound states and applying restrained MD to stabilize the obtained systems. In Section 3.5, unbiased simulations as long as 200 ns started from each of these states are shown to be stable as they do not achieve any transitions between the identified states. This indicates that these conformations do exist within the apo form of HSP90. Importantly, this is in accordance with other studies of the NTD, which observe that the apo state visits varying conformations of the NTD lid [204, 226]. While the stability of each of these state for the apo structure should be further examined using more MD simulations, we can conclude in the extent of the present work that the 5 separate conformational states of the flexible loop can all be visited in the apo form, and transitions among these conformations have been achieved by biased sampling of the autoencoder CV. We amphasize again that biased sampling applied with more traditional CVs, namely a selection of the dihedral angles of the loop of interest, did not yield any transitions.

To the best of our knowledge, enhanced sampling of the HSP90 NTD alone in apo form has not been performed. Unbiased sampling of the apo NTD shows some structural motions of the loop of interest, but no clear transitions (e.g. between the helix conformation of state 5 and the loop conformations of other states) is achieved in the μs timescale [226]. Many studies report enhanced sampling simulations of the bound NTD (taken separately or within the whole dimer), where the collective variable used is often the distance between the NTD and the ligand [200, 202, 227]. These simulations often aim to compute the binding affinity of the NTD to the ligands, and consist of short dissociation trajectories to compare a wide array of inhibitors. Some of these studies do differentiate between conformations of the loop of interest [228], by, e.g. distinguishing between ligands which bind to the various conformations (loop or helix). However, the aim of these works is not to drive

transitions among these conformations, and structural motions of the loop of interest during the dissociation MD simulations are not reported/observed. In [207], Umbrella sampling of the whole dimer, using the angle formed by the two monomers, is performed for the apo structure, where the computed free energy surface shows the presence of two low energy states corresponding to the stretched and compact conformations, but the authors report no corresponding important motions in the NTD site itself.

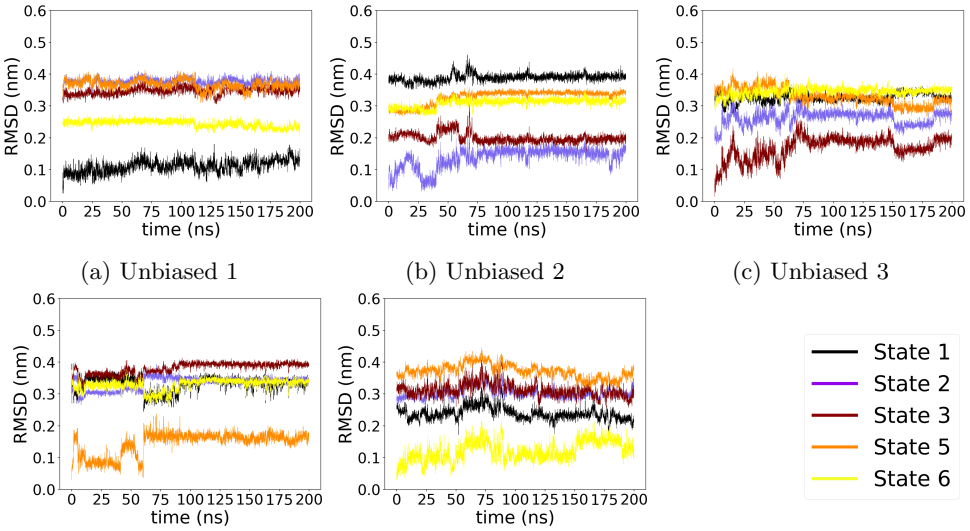
3.4.3 Future work

A natural next step for this work is to compute the free energy landscape of the 2 dimensional autoencoder CV. Because the identified states can somewhat be linked to apo or holo conformations of the HSP90 protein, computing the free energy surface could provide insights into binding mechanisms of the protein. Additionally, as mentioned in Section 3.4.1, while we argue that our choice of state definition, based on the clustering of a large number of crystal structures, is adequate, it is not the only way to obtain the initial states. Free energy computations can help redefine the conformational states of HSP90, and possibly discover new states in the local minima of the energy landscape, which were not included in the initial analysis.

Finally, another possibility is to forgo state definitions altogether and to use only one conformation of HSP90 to build the learning dataset. Such unbiased simulations are expected to be trapped within the initial state or only visit conformations close to it, and so iterative algorithms such as AE-ABF, which was proven to work well on small systems in Chapter 2, can be used. Assessing the efficiency of this method on a larger system such as HSP90 presents an interesting challenge.

3.5 Supplemenatry Results: Additional unbiased simulations

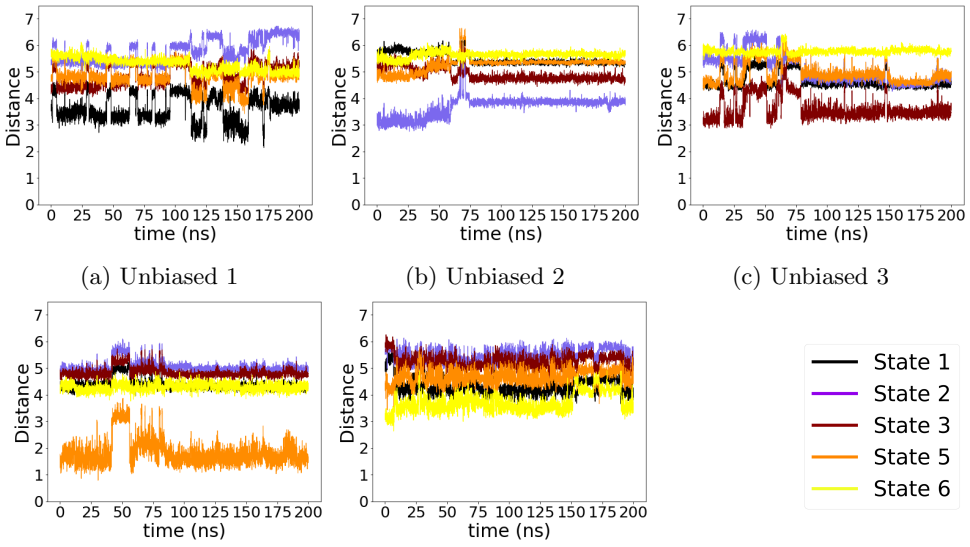
The trajectories used to generate the training dataset, and to characterize the states in order to check for transitions, are short trajectories of 20 ns. We also sampled longer 200 ns unbiased trajectories from each state. This provides a simple sanity check that our states, which were defined using structural clustering only, are metastable. Importantly, these longer simulations also provide a fairer comparison to biased simulations, as they are expected to further explore their respective states. We sample 5 unbiased 200 ns trajectories per state. Figures 3.22 and 3.23 present one such simulation for each state by plotting the RMSD of the loop of interest with respect to each state, and the dihedral clustering centroid distances to each state, computed similarly to what was done for the biased simulation in Section 3.3.2. As can be seen in Figures 3.22 and 3.23, the sampled simulations do not leave their initial states, but visit apparent substates for which the dihedral centroid distances are higher. The existence of these substates may be used to relax the definition of the boundaries of each conformational state. For example, Plot (d) of Figure 3.23 shows the dihedral centroid distances for an unbiased trajectory started from state 5. It can be observed that after ~ 45 ns of simulation time, the distances increase from their usual range of $[1, 2]$ (i.e. the range determined by the shorter unbiased simulations, see Figure 3.14, left, column "Unb 5") to a higher average of $[2.5, 3.5]$. After ~ 85 ns of simulation time, the distances fall back to the usual range.



(a) Unbiased 1 (b) Unbiased 2 (c) Unbiased 3

(d) Unbiased 5 (e) Unbiased 6

Figure 3.22: RMSD of the Cα carbons of the loop of interest over 200-ns unbiased trajectories started from the 5 predefined HSP90 states.



(a) Unbiased 1 (b) Unbiased 2 (c) Unbiased 3

(d) Unbiased 5 (e) Unbiased 6

Figure 3.23: Cluster centroid distances computed for 200-ns unbiased trajectories started from the 5 predefined HSP90 states.

Bibliography

[1] D.E. Shaw, P. Maragakis, K. Lindorff-Larsen, S. Piana, R.O. Dror, M.P. Eastwood, J.A. Bank, J.M. Jumper, J.K. Salmon, Y. Shan, and W. Wriggers. Atomic-level characterization of the structural dynamics of proteins. *Science*, 330(6002):341–346, 2010.

[2] S.A. Hollingsworth and R.O. Dror. Molecular dynamics simulation for all. *Neuron*, 99(6):1129–1143, 2018.

[3] B. Hashemian, D. Millán, and M. Arroyo. Modeling and enhanced sampling of molecular systems with smooth and nonlinear data-driven collective variables. *The Journal of Chemical Physics*, 139(21):214101, 2013.

[4] C. Abrams and G. Bussi. Enhanced sampling in molecular dynamics using metadynamics, replica-exchange, and temperature-acceleration. *Entropy*, 16(1):163–199, 2014.

[5] E. Marinari and G. Parisi. Simulated tempering: a new Monte Carlo scheme. *Europhysics Letters*, 19(6):451, 1992.

[6] R.H. Swendsen and J.S. Wang. Replica Monte Carlo simulation of spin-glasses. *Physical Review Letters*, 57(21):2607, 1986.

[7] Y.M. Rhee and V.S. Pande. Multiplexed-replica exchange molecular dynamics method for protein folding simulation. *Biophysical Journal*, 84(2):775–786, 2003.

[8] N. Nakajima, H. Nakamura, and A. Kidera. Multicanonical ensemble generated by molecular dynamics simulation for enhanced conformational sampling of peptides. *The Journal of Physical Chemistry B*, 101(5):817–824, 1997.

[9] M.R. Sørensen and A.F. Voter. Temperature-accelerated dynamics for simulation of infrequent events. *The Journal of Chemical Physics*, 112(21):9599–9606, 2000.

[10] F. Wang and D.P. Landau. Efficient, multiple-range random walk algorithm to calculate the density of states. *Physical Review Letters*, 86(10):2050, 2001.

[11] A. Laio and M. Parrinello. Escaping free-energy minima. *Proceedings of the National Academy of Sciences*, 99(20):12562–12566, 2002.

[12] A. Laio and F.L. Gervasio. Metadynamics: a method to simulate rare events and reconstruct the free energy in biophysics, chemistry and material science. *Reports on Progress in Physics*, 71(12):126601, 2008.

[13] J. Kästner. Umbrella sampling. *Wiley Interdisciplinary Reviews: Computational Molecular Science*, 1(6):932–942, 2011.

[14] G.M. Torrie and J.P. Valleau. Nonphysical sampling distributions in Monte Carlo free-energy estimation: Umbrella sampling. *Journal of Computational Physics*, 23(2):187–199, 1977.

[15] J. Comer, J.C. Gumbart, J. Hénin, T. Lelièvre, A. Pohorille, and C. Chipot. The adaptive biasing force method: Everything you always wanted to know but were afraid to ask. *The Journal of Physical Chemistry B*, 119(3):1129–1151, 2015.

[16] J. Hénin and C. Chipot. Overcoming free energy barriers using unconstrained molecular dynamics simulations. *The Journal of Chemical Physics*, 121(7):2904–2914, 2004.

[17] E. Darve and A. Pohorille. Calculating free energies using average force. *The Journal of Chemical Physics*, 115(20):9169–9183, 2001.

[18] E. Darve, D. Rodríguez-Gómez, and A. Pohorille. Adaptive biasing force method for scalar and vector free energy calculations. *The Journal of chemical physics*, 128(14):144120, 2008.

[19] Z. Belkacemi, P. Gkeka, T. Lelièvre, and G. Stoltz. Chasing Collective Variables using Autoencoders and biased trajectories. *Journal of Chemical Theory and Computation*, 2021.

[20] P. Gkeka, G. Stoltz, A. Barati Farimani, Z. Belkacemi, M. Ceriotti, J.D. Chodera, A.R. Dinner, A.L. Ferguson, J.B. Maillet, H. Minoux, C. Peter, F. Pietrucci, A. Silveira, A. Tkatchenko, Z. Trstanova, R. Wiewiora, and T. Lelièvre. Machine learning force fields and coarse-grained variables in molecular dynamics: Application to materials and biological systems. *Journal of Chemical Theory and Computation*, 16(8):4757–4775, 2020.

[21] T. Lelièvre and G. Stoltz. Partial differential equations and stochastic methods in molecular dynamics. *Acta Numerica*, 25:681–880, 2016.

[22] M. Rousset, G. Stoltz, and T. Lelievre. *Free Energy Computations: A Mathematical Perspective*. World Scientific, 2010.

[23] M.A. González. Force fields and molecular dynamics simulations. *École thématique de la Société Française de la Neutronique*, 12:169–200, 2011.

[24] D. Frenkel, B. Smit, J. Tobochnik, S.R. McKay, and W. Christian. Understanding molecular simulation. *Computers in Physics*, 11(4):351–354, 1997.

[25] M.P. Allen and D.J. Tildesley. *Computer simulation of liquids*. Oxford university press, 2017.

[26] J.A. McCammon, B.R. Gelin, and M. Karplus. Dynamics of folded proteins. *Nature*, 267(5612):585–590, 1977.

[27] G. Jayachandran, V. Vishal, and V.S. Pande. Using massively parallel simulation and Markovian models to study protein folding: examining the dynamics of the villin headpiece. *The Journal of chemical physics*, 124(16):164902, 2006.

[28] P.L. Freddolino, A.S. Arkhipov, S.B. Larson, A. McPherson, and K. Schulten. Molecular dynamics simulations of the complete satellite tobacco mosaic virus. *Structure*, 14(3):437–449, 2006.

[29] K. Lindorff-Larsen, S. Piana, R.O. Dror, and D.E Shaw. How fast-folding proteins fold. *Science*, 334(6055):517–520, 2011.

[30] J. Huang and A.D. MacKerell Jr. CHARMM36 all-atom additive protein force field: Validation based on comparison to NMR data. *Journal of computational chemistry*, 34(25):2135–2145, 2013.

[31] J. Wang, R.M. Wolf, J.W. Caldwell, P.A. Kollman, and D.A. Case. Development and testing of a general amber force field. *Journal of computational chemistry*, 25(9):1157–1174, 2004.

[32] M. Stroet, K. B Koziara, A.K. Malde, and A.E. Mark. Optimization of empirical force fields by parameter space mapping: A single-step perturbation approach. *Journal of chemical theory and computation*, 13(12):6201–6212, 2017.

[33] A.D. MacKerell Jr, D. Bashford, M.L.D.R. Bellott, R.L. Dunbrack Jr, J.D. Evanseck, M.J. Field, S. Fischer, J. Gao, H. Guo, S. Ha, et al. All-atom empirical potential for molecular modeling and dynamics studies of proteins. *The journal of physical chemistry B*, 102(18):3586–3616, 1998.

[34] W.D. Cornell, P. Cieplak, C.I. Bayly, I.R. Gould, K.M. Merz, D.M. Ferguson, D.C. Spellmeyer, T. Fox, J.W. Caldwell, and P.A. Kollman. A second generation force field for the simulation of proteins, nucleic acids, and organic molecules. *Journal of the American Chemical Society*, 117(19):5179–5197, 1995.

[35] C. Oostenbrink, A. Villa, A.E. Mark, and W.F. Van Gunsteren. A biomolecular force field based on the free enthalpy of hydration and solvation: the GROMOS force-field parameter sets 53A5 and 53A6. *Journal of computational chemistry*, 25(13):1656–1676, 2004.

[36] W.L. Jorgensen, D.S. Maxwell, and J. Tirado-Rives. Development and testing of the OPLS all-atom force field on conformational energetics and properties of organic liquids. *Journal of the American Chemical Society*, 118(45):11225–11236, 1996.

[37] N.L Allinger. Conformational analysis. 130. MM2. A hydrocarbon force field utilizing V1 and V2 torsional terms. *Journal of the American Chemical Society*, 99(25):8127–8134, 1977.

[38] N.L. Allinger, Y.H. Yuh, and J.-H. Lii. Molecular mechanics. The MM3 force field for hydrocarbons. 1. *Journal of the American Chemical Society*, 111(23):8551–8566, 1989.

[39] N.L. Allinger, K. Chen, and J.-H. Lii. An improved force field (MM4) for saturated hydrocarbons. *Journal of computational chemistry*, 17(5-6):642–668, 1996.

[40] T.A. Halgren. Merck molecular force field. I. Basis, form, scope, parameterization, and performance of MMFF94. *Journal of computational chemistry*, 17(5-6):490–519, 1996.

[41] H. Sun. COMPASS: an ab initio force-field optimized for condensed-phase applications overview with details on alkane and benzene compounds. *The Journal of Physical Chemistry B*, 102(38):7338–7364, 1998.

[42] J. Gao, D. Habibollazadeh, and L. Shao. A polarizable intermolecular potential function for simulation of liquid alcohols. *The Journal of Physical Chemistry*, 99(44):16460–16467, 1995.

[43] M. Swart and P.T. Van Duijnen. DRF90: a polarizable force field. *Molecular Simulation*, 32(6):471–484, 2006.

[44] J.W. Ponder, C. Wu, P. Ren, V.S. Pande, J.D. Chodera, M.J. Schnieders, I. Haque, D.L. Mobley, D.S. Lambrecht, R.A. DiStasio Jr, et al. Current status of the AMOEBA polarizable force field. *The journal of physical chemistry B*, 114(8):2549–2564, 2010.

[45] A.C.T. Van Duin, S. Dasgupta, F. Lorant, and W.A. Goddard. ReaxFF: a reactive force field for hydrocarbons. *The Journal of Physical Chemistry A*, 105(41):9396–9409, 2001.

[46] T.P. Senftle, S. Hong, M.M. Islam, S.B. Kylasa, Y. Zheng, Y.K. Shin, C. Junkermeier, R. Engel-Herbert, M.J. Janik, H.M. Aktulga, et al. The ReaxFF reactive force-field: development, applications and future directions. *Computational Materials*, 2(1):1–14, 2016.

[47] J. Behler. Perspective: Machine learning potentials for atomistic simulations. *The Journal of chemical physics*, 145(17):170901, 2016.

[48] S. Chmiela, A. Tkatchenko, H.E. Sauceda, I. Poltavsky, K.T. Schütt, and K.-R. Müller. Machine learning of accurate energy-conserving molecular force fields. *Science advances*, 3(5):e1603015, 2017.

[49] S. Chmiela, H.E. Sauceda, K.-R. Müller, and A. Tkatchenko. Towards exact molecular dynamics simulations with machine-learned force fields. *Nature communications*, 9(1):1–10, 2018.

[50] B. Leimkuhler and C. Matthews. *Molecular Dynamics.* Springer, 2016.

[51] B. Leimkuhler and C. Matthews. Robust and efficient configurational molecular sampling via Langevin dynamics. *The Journal of chemical physics*, 138(17):05B601_1, 2013.

[52] G. Strang. On the construction and comparison of difference schemes. *SIAM journal on numerical analysis*, 5(3):506–517, 1968.

[53] C. Chipot and A. Pohorille. Free energy calculations. *Springer series in chemical physics*, 86:159–184, 2007.

[54] R.W. Zwanzig. High-temperature equation of state by a perturbation method. I. Nonpolar gases. *The Journal of Chemical Physics*, 22(8):1420–1426, 1954.

[55] A.M. Ferrenberg and R.H. Swendsen. Optimized Monte Carlo data analysis. *Computers in Physics*, 3(5):101–104, 1989.

[56] S. Kumar, J.M. Rosenberg, D. Bouzida, R.H. Swendsen, and P.A. Kollman. The weighted histogram analysis method for free-energy calculations on biomolecules. I. The method. *Journal of computational chemistry*, 13(8):1011–1021, 1992.

[57] C.H. Bennett. Efficient estimation of free energy differences from Monte Carlo data. *Journal of Computational Physics*, 22(2):245–268, 1976.

[58] Z. Tan. On a likelihood approach for Monte Carlo integration. *Journal of the American Statistical Association*, 99(468):1027–1036, 2004.

[59] M.R. Shirts and J.D. Chodera. Statistically optimal analysis of samples from multiple equilibrium states. *The Journal of chemical physics*, 129(12):124105, 2008.

[60] J.G. Kirkwood. Statistical mechanics of fluid mixtures. *The Journal of chemical physics*, 3(5):300–313, 1935.

[61] C. Jarzynski. Equilibrium free-energy differences from nonequilibrium measurements: A master-equation approach. *Physical Review E*, 56(5):5018, 1997.

[62] C. Jarzynski. Nonequilibrium equality for free energy differences. *Physical Review Letters*, 78(14):2690, 1997.

[63] M. Rousset and G. Stoltz. Equilibrium sampling from nonequilibrium dynamics. *Journal of Statistical Physics*, 123(6):1251–1272, 2006.

[64] S. Vaikuntanathan and C. Jarzynski. Escorted free energy simulations: Improving convergence by reducing dissipation. *Physical review letters*, 100(19):190601, 2008.

[65] F. Wang and D.P. Landau. Determining the density of states for classical statistical models: A random walk algorithm to produce a flat histogram. *Physical Review E*, 64(5):056101, 2001.

[66] H. Alrachid and T. Lelièvre. Long-time convergence of an adaptive biasing force method: Variance reduction by Helmholtz projection. *The SMAI Journal of Computational Mathematics*, 1:55–82, 2015.

[67] A. Lesage, T. Lelièvre, G. Stoltz, and J. Hénin. Smoothed biasing forces yield unbiased free energies with the extended-system adaptive biasing force method. *Journal of Physical Chemistry B*, 121(15):3676–3685, 2016.

[68] T. Lelièvre, M. Rousset, and G. Stoltz. Long-time convergence of an adaptive biasing force method. *Nonlinearity*, 21(6):1155, 2008.

[69] T. Lelièvre. Two mathematical tools to analyze metastable stochastic processes. In *Numerical Mathematics and Advanced Applications 2011*, pages 791–810. Springer, 2013.

[70] T. Lelievre, M. Rousset, and G. Stoltz. Langevin dynamics with constraints and computation of free energy differences. *Mathematics of computation*, 81(280):2071–2125, 2012.

[71] G. Bussi and A. Laio. Using metadynamics to explore complex free-energy landscapes. *Nature Reviews Physics*, 2(4):200–212, 2020.

[72] W. Zhang, C. Hartmann, and C. Schütte. Effective dynamics along given reaction coordinates, and reaction rate theory. *Faraday discussions*, 195:365–394, 2017.

[73] F. Legoll and T. Lelievre. Effective dynamics using conditional expectations. *Nonlinearity*, 23(9):2131, 2010.

[74] A. Bittracher, R. Banisch, and C. Schütte. Data-driven computation of molecular reaction coordinates. *The Journal of chemical physics*, 149(15):154103, 2018.

[75] A. Bittracher, M. Mollenhauer, P. Koltai, and C. Schütte. Optimal Reaction Coordinates: Variational Characterization and Sparse Computation. *arXiv preprint arXiv:2107.10158*, 2021.

[76] S. Klus, A. Bittracher, I. Schuster, and C. Schütte. A kernel-based approach to molecular conformation analysis. *The Journal of Chemical Physics*, 149(24):244109, 2018.

[77] A. Bittracher, P. Koltai, S. Klus, R. Banisch, M. Dellnitz, and C. Schütte. Transition manifolds of complex metastable systems. *Journal of nonlinear science*, 28(2):471–512, 2018.

[78] B. Peters and B.L. Trout. Obtaining reaction coordinates by likelihood maximization. *The Journal of Chemical Physics*, 125(5):054108, 2006.

[79] A. Ma and A.R. Dinner. Automatic method for identifying reaction coordinates in complex systems. *The Journal of Physical Chemistry B*, 109(14):6769–6779, 2005.

[80] W. Li and A. Ma. Recent developments in methods for identifying reaction coordinates. *Molecular simulation*, 40(10-11):784–793, 2014.

[81] B. Peters. Using the histogram test to quantify reaction coordinate error, 2006.

[82] A. Glielmo, B.E. Husic, A. Rodriguez, C. Clementi, F. Noé, and A. Laio. Unsupervised Learning Methods for Molecular Simulation Data. *Chemical Reviews*, 2021.

[83] H. Sidky, W. Chen, and A.L. Ferguson. Machine learning for collective variable discovery and enhanced sampling in biomolecular simulation. *Molecular Physics*, 118(5):e1737742, 2020.

[84] W. Kabsch. A discussion of the solution for the best rotation to relate two sets of vectors. *Acta Crystallographica Section A*, 34(5):827–828, 1978.

[85] R. Diamond. A note on the rotational superposition problem. *Acta Crystallographica Section A: Foundations of Crystallography*, 44(2):211–216, 1988.

[86] D.L. Theobald. Rapid calculation of RMSDs using a quaternion-based characteristic polynomial. *Acta Crystallographica Section A: Foundations of Crystallography*, 61(4):478–480, 2005.

[87] P. Liu, D.K. Agrafiotis, and D.L. Theobald. Fast determination of the optimal rotational matrix for macromolecular superpositions. *Journal of computational chemistry*, 31(7):1561–1563, 2010.

[88] D.R. Ferro and J. Hermans. A different best rigid-body molecular fit routine. *Acta Crystallographica Section A: Crystal Physics, Diffraction, Theoretical and General Crystallography*, 33(2):345–347, 1977.

[89] A.D. McLachlan. Rapid comparison of protein structures. *Acta Crystallographica Section A: Crystal Physics, Diffraction, Theoretical and General Crystallography*, 38(6):871–873, 1982.

[90] F. Sittel, A. Jain, and G. Stock. Principal component analysis of molecular dynamics: On the use of Cartesian vs. internal coordinates. *The Journal of Chemical Physics*, 141(1):07B605_1, 2014.

[91] F. Sittel and G. Stock. Perspective: Identification of collective variables and metastable states of protein dynamics. *The Journal of chemical physics*, 149(15):150901, 2018.

[92] V. Gapsys and B.L. de Groot. Optimal superpositioning of flexible molecule ensembles. *Biophysical journal*, 104(1):196–207, 2013.

[93] W. Chen and A.L. Ferguson. Molecular enhanced sampling with autoencoders: On-the-fly collective variable discovery and accelerated free energy landscape exploration. *Journal of Computational Chemistry*, 39(25):2079–2102, 2018.

[94] W. Chen, A.R. Tan, and A.L. Ferguson. Collective variable discovery and enhanced sampling using autoencoders: Innovations in network architecture and error function design. *The Journal of Chemical Physics*, 149(7):072–312, 2018.

[95] M. Ernst, F. Sittel, and G. Stock. Contact-and distance-based principal component analysis of protein dynamics. *The Journal of chemical physics*, 143(24):12B640_1, 2015.

[96] K.V. Mardia and P.E. Jupp. *Directional statistics*, volume 494. John Wiley & Sons, 2009.

[97] N.I. Fisher. *Statistical analysis of circular data*. cambridge university press, 1995.

[98] Y. Mu, P.H. Nguyen, and G. Stock. Energy landscape of a small peptide revealed by dihedral angle principal component analysis. *Proteins: Structure, Function, and Bioinformatics*, 58(1):45–52, 2005.

[99] A. Altis, P.H. Nguyen, R. Hegger, and G. Stock. Dihedral angle principal component analysis of molecular dynamics simulations. *The Journal of chemical physics*, 126(24):244111, 2007.

[100] J.M.L. Ribeiro, P. Bravo, Y. Wang, and P. Tiwary. Reweighted autoencoded variational Bayes for enhanced sampling (RAVE). *The Journal of Chemical Physics*, 149(7):072301, 2018.

[101] Z. Shamsi, K.J. Cheng, and D. Shukla. Reinforcement learning based adaptive sampling: REAPing rewards by exploring protein conformational landscapes. *The Journal of Physical Chemistry B*, 122(35):8386–8395, 2018.

[102] T. Ichiye and M. Karplus. Collective motions in proteins: a covariance analysis of atomic fluctuations in molecular dynamics and normal mode simulations. *Proteins: Structure, Function, and Bioinformatics*, 11(3):205–217, 1991.

[103] A. Amadei, A.B.M. Linssen, and H.J.C. Berendsen. Essential dynamics of proteins. *Proteins: Structure, Function, and Bioinformatics*, 17(4):412–425, 1993.

[104] T. Hofmann, B. Schölkopf, and A.J. Smola. Kernel methods in machine learning. *The annals of statistics*, 36(3):1171–1220, 2008.

[105] J.B. Tenenbaum, V. De Silva, and J.C. Langford. A global geometric framework for nonlinear dimensionality reduction. *science*, 290(5500):2319–2323, 2000.

[106] P. Das, M. Moll, H. Stamati, L.E. Kavraki, and C. Clementi. Low-dimensional, free-energy landscapes of protein-folding reactions by nonlinear dimensionality reduction. *Proceedings of the National Academy of Sciences*, 103(26):9885–9890, 2006.

[107] R.R. Coifman, I.G. Kevrekidis, S. Lafon, M. Maggioni, and B. Nadler. Diffusion maps, reduction coordinates, and low dimensional representation of stochastic systems. *Multiscale Modeling & Simulation*, 7(2):842–864, 2008.

[108] B. Nadler, S. Lafon, R.R. Coifman, and I.G. Kevrekidis. Diffusion maps, spectral clustering and reaction coordinates of dynamical systems. *Applied and Computational Harmonic Analysis*, 21(1):113–127, 2006.

[109] Z. Trstanova, B. Leimkuhler, and T. Lelièvre. Local and global perspectives on diffusion maps in the analysis of molecular systems. *Proceedings of the Royal Society A*, 476(2233):20190036, 2020.

[110] G. Hinton and S.T. Roweis. Stochastic neighbor embedding. In *NIPS*, volume 15, pages 833–840. Citeseer, 2002.

[111] L.J.P. Van der Maaten and G.E. Hinton. Visualizing high-dimensional data using t-SNE. *Journal of Machine Learning Research*, 9:2579–2605, 2008.

[112] Moj. Duan, J. Fan, M. Li, L. Han, and S. Huo. Evaluation of dimensionality-reduction methods from peptide folding–unfolding simulations. *Journal of chemical theory and computation*, 9(5):2490–2497, 2013.

[113] G.A. Tribello and P. Gasparotto. Using dimensionality reduction to analyze protein trajectories. *Frontiers in molecular biosciences*, 6:46, 2019.

[114] K.L. Priddy and P.E. Keller. *Artificial neural networks: an introduction*, volume 68. SPIE press, 2005.

[115] M.A. Kramer. Autoassociative neural networks. *Computers & Chemical Engineering*, 16(4):313–328, 1992.

[116] E. Plaut. From principal subspaces to principal components with linear autoencoders. *arXiv preprint arXiv:1804.10253*, 2018.

[117] H. Bourlard and Y. Kamp. Auto-association by multilayer perceptrons and singular value decomposition. *Biological Cybernetics*, 59(4-5):291–294, 1988.

[118] D.P. Kingma and M. Welling. Auto-encoding variational Bayes. *arXiv preprint arXiv:1312.6114*, 2013.

[119] Y. Wang, J.M.L. Ribeiro, and P. Tiwary. Past–future information bottleneck for sampling molecular reaction coordinate simultaneously with thermodynamics and kinetics. *Nature communications*, 10(1):1–8, 2019.

[120] Y. B. Varolgüneş, T. Bereau, and J.F. Rudzinski. Interpretable embeddings from molecular simulations using Gaussian mixture variational autoencoders. *Machine Learning: Science and Technology*, 1(1):015012, 2020.

[121] M. Schöberl, N. Zabaras, and P.-S. Koutsourelakis. Predictive collective variable discovery with deep Bayesian models. *The Journal of Chemical Physics*, 150(2):024109, 2019.

[122] C. Wehmeyer and F. Noé. Time-lagged autoencoders: Deep learning of slow collective variables for molecular kinetics. *The Journal of Chemical Physics*, 148(24):241703, 2018.

[123] C.X. Hernández, H.K. Wayment-Steele, M.M. Sultan, B.E. Husic, and V.S. Pande. Variational encoding of complex dynamics. *Physical Review E*, 97(6):062412, 2018.

[124] Y. Bengio, O. Delalleau, N.L. Roux, J.-F. Paiement, P. Vincent, and M. Ouimet. Learning eigenfunctions links spectral embedding and kernel PCA. *Neural computation*, 16(10):2197–2219, 2004.

[125] B. Hashemian, D. Millán, and M. Arroyo. Modeling and enhanced sampling of molecular systems with smooth and nonlinear data-driven collective variables. *The Journal of chemical physics*, 139(21):12B601_1, 2013.

[126] G. Mishne, U. Shaham, A. Cloninger, and I. Cohen. Diffusion nets. *Applied and Computational Harmonic Analysis*, 47(2):259–285, 2019.

[127] D. Trapl, I. Horvacanin, V. Mareska, F. Ozcelik, G. Unal, and V. Spiwok. Anncolvar: approximation of complex collective variables by artificial neural networks for analysis and biasing of molecular simulations. *Frontiers in Molecular Biosciences*, 6:25, 2019.

[128] S. Klus, F. Nüske, P. Koltai, H. Wu, I. Kevrekidis, C. Schütte, and F. Noé. Data-driven model reduction and transfer operator approximation. *Journal of Nonlinear Science*, 28(3):985–1010, 2018.

[129] H. Wu and F. Noé. Variational approach for learning markov processes from time series data. *Journal of Nonlinear Science*, 30(1):23–66, 2020.

[130] F. Nüske, B.G. Keller, G. Pérez-Hernández, A.S.J.S. Mey, and F. Noé. Variational approach to molecular kinetics. *Journal of Chemical Theory and Computation*, 10(4):1739–1752, 2014.

[131] F. Noé and F. Nüske. A variational approach to modeling slow processes in stochastic dynamical systems. *Multiscale Modeling & Simulation*, 11(2):635–655, 2013.

[132] G. Pérez-Hernández, F. Paul, T. Giorgino, G. De Fabritiis, and F. Noé. Identification of slow molecular order parameters for Markov model construction. *The Journal of chemical physics*, 139(1):07B604_1, 2013.

[133] L. Molgedey and H.G. Schuster. Separation of a mixture of independent signals using time delayed correlations. *Physical Review Letters*, 72(23):3634, 1994.

[134] B.E. Husic and V.S. Pande. Markov state models: From an art to a science. *Journal of the American Chemical Society*, 140(7):2386–2396, 2018.

[135] J.D. Chodera and F. Noé. Markov state models of biomolecular conformational dynamics. *Current Opinion in Structural Biology*, 25:135–144, 2014.

[136] A. Sirur, D. De Sancho, and R.B. Best. Markov state models of protein misfolding. *The Journal of Chemical Physics*, 144(7):075101, 2016.

[137] D. Shukla, C.X. Hernández, J.K. Weber, and V.S. Pande. Markov state models provide insights into dynamic modulation of protein function. *Accounts of Chemical Research*, 48(2):414–422, 2015.

[138] J.-H. Prinz, H. Wu, M. Sarich, B. Keller, M. Senne, M. Held, J.D. Chodera, C. Schütte, and F. Noé. Markov models of molecular kinetics: Generation and validation. *The Journal of chemical physics*, 134(17):174105, 2011.

[139] B. Trendelkamp-Schroer and F. Noé. Efficient Bayesian estimation of Markov model transition matrices with given stationary distribution. *The Journal of chemical physics*, 138(16):04B612, 2013.

[140] F. Noé. Probability distributions of molecular observables computed from Markov models. *The Journal of chemical physics*, 128(24):244103, 2008.

[141] M. Sarich, R. Banisch, C. Hartmann, and C. Schütte. Markov state models for rare events in molecular dynamics. *Entropy*, 16(1):258–286, 2014.

[142] C. Schütte, A. Fischer, W. Huisinga, and P. Deuflhard. A direct approach to conformational dynamics based on hybrid Monte Carlo. *Journal of Computational Physics*, 151(1):146–168, 1999.

[143] G.R. Bowman. Improved coarse-graining of Markov state models via explicit consideration of statistical uncertainty. *The Journal of chemical physics*, 137(13):134111, 2012.

[144] Y. Yao, R.Z. Cui, G.R. Bowman, D.-A. Silva, J. Sun, and X. Huang. Hierarchical Nyström methods for constructing Markov state models for conformational dynamics. *The Journal of chemical physics*, 138(17):05B602_1, 2013.

[145] A. Mardt, L. Pasquali, H. Wu, and F. Noé. VAMPnets for deep learning of molecular kinetics. *Nature Communications*, 9(1):1–11, 2018.

[146] W. Chen, H. Sidky, and A.L. Ferguson. Nonlinear discovery of slow molecular modes using state-free reversible VAMPnets. *The Journal of Chemical Physics*, 150(21):214114, 2019.

[147] L. Bonati, G.M. Piccini, and M. Parrinello. Deep learning the slow modes for rare events sampling. *arXiv preprint arXiv:2107.03943*, 2021.

[148] M.M. Sultan and V.S. Pande. Automated design of collective variables using supervised machine learning. *The Journal of Chemical Physics*, 149(9):094–106, 2018.

[149] D. Mendels, G.M. Piccini, and M. Parrinello. Collective variables from local fluctuations. *The journal of physical chemistry letters*, 9(11):2776–2781, 2018.

[150] L. Bonati, V. Rizzi, and M. Parrinello. Data-driven collective variables for enhanced sampling. *The Journal of Physical Chemistry Letters*, 11(8):2998–3004, 2020.

[151] M. Crawshaw. Multi-task learning with deep neural networks: A survey. *arXiv preprint arXiv:2009.09796*, 2020.

[152] M. Frassek, A. Arjun, and P.G. Bolhuis. An extended autoencoder model for reaction coordinate discovery in rare event molecular dynamics datasets. *The Journal of Chemical Physics*, 155(6):064103, 2021.

[153] V. Mironov, Y. Alexeev, V.K. Mulligan, and D.G. Fedorov. A systematic study of minima in alanine dipeptide. *Journal of Computational Chemistry*, 40(2):297–309, 2019.

[154] H. Wu, F. Nüske, F. Paul, S. Klus, P. Koltai, and F. Noé. Variational Koopman models: slow collective variables and molecular kinetics from short off-equilibrium simulations. *The Journal of Chemical Physics*, 146(15):154104, 2017.

[155] M.M. Sultan, G. Kiss, D. Shukla, and V.S. Pande. Automatic selection of order parameters in the analysis of large scale molecular dynamics simulations. *Journal of Chemical Theory and Computation*, 10(12):5217–5223, 2014.

[156] J. Rogal, E. Schneider, and M.E. Tuckerman. Neural-network-based path collective variables for enhanced sampling of phase transformations. *Physical Review Letters*, 123(24):245701, 2019.

[157] L. Felsberger and P.-S. Koutsourelakis. Physics-constrained, data-driven discovery of coarse-grained dynamics. *arXiv preprint arXiv:1802.03824*, 2018. `https://arxiv.org/abs/1802.03824` Accessed December 4th 2021.

[158] P. Ravindra, Z. Smith, and P. Tiwary. Automatic mutual information noise omission (AMINO): generating order parameters for molecular systems. *Molecular Systems Design & Engineering*, 5(1):339–348, 2020.

[159] S. Park, M.K. Sener, D. Lu, and K. Schulten. Reaction paths based on mean first-passage times. *The Journal of Chemical Physics*, 119(3):1313–1319, 2003.

[160] T. Lelièvre, M. Rousset, and G. Stoltz. Computation of free energy profiles with parallel adaptive dynamics. *The Journal of Chemical Physics*, 126(13):134111, 2007.

[161] F. Chollet. Keras, 2015.

[162] M. Abadi, A. Agarwal, P. Barham, E. Brevdo, Z. Zhifeng Chen, C. Citro, G.S. Corrado, A. Davis, J. Dean, M. Devin, S. Ghemawat, I. Goodfellow, A. Harp, G. Irving, M. Isard, Y. Jia, R. Jozefowicz, L. Kaiser, M. Kudlur, J. Levenberg, D. Mané, R. Monga, S. Moore, D. Murray, C. Ola, M. Schuster, J. Shlens, B. Steiner, I. Sutskever, K. Talwar, P. Tucker, V. Vanhoucke, V. Vasudevan, F. Viégas, O. Vinyals, P. Warden, M. Wattenberg, M. Wicke, Y. Yu, and X. Zheng. TensorFlow: Large-scale machine learning on heterogeneous systems, 2015. Software available from tensorflow.org.

[163] P. Eastman, J. Swails, J.D. Chodera, R.T. McGibbon, Y. Zhao, K.A. Beauchamp, L.P. Wang, A.C. Simmonett, M.P. Harrigan, C.D. Stern, R.P. Wiewiora, B.R. Brooks, and V.S. Pande. OpenMM 7: Rapid development of high performance algorithms for molecular dynamics. *PLOS Computational Biology*, 13(7):1–17, 2017.

[164] M. Bonomi, G. Bussi, C. Camilloni, G.A. Tribello, P. Banáš, A. Barducci, M. Bernetti, P.G. Bolhuis, S. Bottaro, and D. Branduardi. Promoting transparency and reproducibility in enhanced molecular simulations. *Nature Methods*, 16(8):670–673, 2019.

[165] G.A. Tribello, M. Bonomi, D. Branduardi, C. Camilloni, and G. Bussi. PLUMED 2: New feathers for an old bird. *Computer Physics Communications*, 185(2):604–613, 2014.

[166] H. Chen, H. Fu, X. Shao, C. Chipot, and W. Cai. ELF: an extended-Lagrangian free energy calculation module for multiple molecular dynamics engines. *J. Chem. Inf. Model.*, 58(7):1315–1318, 2018.

[167] openmm-plumed: OpenMM plugin to interface with Plumed. `https://github.com/openmm/openmm-plumed`.

[168] G. Fiorin, M.L. Klein, and J. Hénin. Using collective variables to drive molecular dynamics simulations. *Molecular Physics*, 111(22-23):3345–3362, 2013.

[169] D.P. Kingma and J. Ba. Adam: A method for stochastic optimization. *3rd International Conference for Learning Representations*, 2014.

[170] N. Morgan and H. Bourlard. Generalization and parameter estimation in feedforward nets: Some experiments. In D. S. Touretzky, editor, *Advances in Neural Information Processing Systems 2*, pages 630–637. Morgan-Kaufmann, 1990.

[171] V. Hornak, R. Abel, A. Okur, B. Strockbine, A. Roitberg, and C. Simmerling. Comparison of multiple Amber force fields and development of improved protein backbone parameters. *Proteins: Struct., Funct., Bioinf.*, 65(3):712–725, 2006.

[172] D. Satoh, K. Shimizu, S. Nakamura, and T. Terada. Folding free-energy landscape of a 10-residue mini-protein, chignolin. *FEBS Lett.*, 580(14):3422–3426, 2006.

[173] K. Lindorff-Larsen, S. Piana, K. Palmo, P. Maragakis, J.L. Klepeis, R.O. Dror, and D.E. Shaw. Improved side-chain torsion potentials for the Amber ff99SB protein force field. *Proteins: Struct., Funct., Bioinf.*, 78(8):1950–1958, 2010.

[174] W.L. Jorgensen, J. Chandrasekhar, J.D. Madura, R.W. Impey, and M.L. Klein. Comparison of simple potential functions for simulating liquid water. *J. Chem. Phys.*, 79(2):926–935, 1983.

[175] B. Hess, H. Bekker, H.J.C. Berendsen, and J.G.E.M. Fraaije. LINCS: a linear constraint solver for molecular simulations. *Journal of Computational Chemistry*, 18(12):1463–1472, 1997.

[176] P.G. Bolhuis, C. Dellago, and D. Chandler. Reaction coordinates of biomolecular isomerization. *Proc. Natl. Acad. Sci. U.S.A.*, 97(11):5877–5882, 2000.

[177] B. Hashemian and M. Arroyo. Topological obstructions in the way of data-driven collective variables. *J. Chem. Phys.*, 142(4):044102, 2015.

[178] S. Honda, K. Yamasaki, Y. Sawada, and H. Morii. 10 residue folded peptide designed by segment statistics. *Structure*, 12(8):1507–1518, 2004.

[179] D. Picard. Heat-shock protein 90, a chaperone for folding and regulation. *Cellular and Molecular Life Sciences CMLS*, 59(10):1640–1648, 2002.

[180] K. Richter, P. Muschler, O. Hainzl, and J. Buchner. Coordinated ATP hydrolysis by the Hsp90 dimer. *Journal of Biological Chemistry*, 276(36):33689–33696, 2001.

[181] J.D. Chavez, D.K. Schweppe, J.K. Eng, and J.E. Bruce. In vivo conformational dynamics of Hsp90 and its interactors. *Cell chemical biology*, 23(6):716–726, 2016.

[182] J.C. Young, I. Moarefi, and F.U. Hartl. Hsp90: a specialized but essential protein-folding tool. *The Journal of cell biology*, 154(2):267, 2001.

[183] L. Whitesell and S.L. Lindquist. HSP90 and the chaperoning of cancer. *Nature Reviews Cancer*, 5(10):761–772, 2005.

[184] R.E. Lackie, A. Maciejewski, V.G. Ostapchenko, J. Marques-Lopes, W.-Y. Choy, M.L. Duennwald, V.F. Prado, and M.A.M. Prado. The Hsp70/Hsp90 chaperone machinery in neurodegenerative diseases. *Frontiers in neuroscience*, 11:254, 2017.

[185] R.A. Watkins, C. Evans-Molina, J.K. Terrell, K.H. Day, L. Guindon, I.A. Restrepo, R.G. Mirmira, J.S. Blum, and L.A. DiMeglio. Proinsulin and heat shock protein 90 as biomarkers of beta-cell stress in the early period after onset of type 1 diabetes. *Translational Research*, 168:96–106, 2016.

[186] T. Didenko, A.M.S. Duarte, G.E. Karagöz, and S.G.D. Rüdiger. Hsp90 structure and function studied by NMR spectroscopy. *Biochimica et Biophysica Acta (BBA)-Molecular Cell Research*, 1823(3):636–647, 2012.

[187] S.E. Jackson. Hsp90: structure and function. *Molecular chaperones*, pages 155–240, 2012.

[188] S. Tsutsumi, M. Mollapour, C. Prodromou, C.-T. Lee, B. Panaretou, S. Yoshida, M.P. Mayer, and L.M. Neckers. Charged linker sequence modulates eukaryotic heat shock protein 90 (Hsp90) chaperone activity. *Proceedings of the National Academy of Sciences*, 109(8):2937–2942, 2012.

[189] W. Ye, M. Götz, S. Celiksóy, L. Túting, C. Ratzke, J. Prasad, J. Ricken, S.V. Wegner, R. Ahijado-Guzmán, T. Hugel, et al. Conformational dynamics of a single protein monitored for 24 h at video rate. *Nano letters*, 18(10):6633–6637, 2018.

[190] L.H. Pearl and C. Prodromou. Structure and mechanism of the Hsp90 molecular chaperone machinery. *Annu. Rev. Biochem.*, 75:271–294, 2006.

[191] C. Prodromou, B. Panaretou, S. Chohan, G. Siligardi, R. O'Brien, J.E. Ladbury, S.M. Roe, P.W. Piper, and L.H. Pearl. The ATPase cycle of Hsp90 drives a molecular 'clamp'via transient dimerization of the N-terminal domains. *The EMBO journal*, 19(16):4383–4392, 2000.

[192] C. Prodromou, S.M. R., R. O'Brien, J.E. Ladbury, P.W. Piper, and L.H. Pearl. Identification and structural characterization of the ATP/ADP-binding site in the Hsp90 molecular chaperone. *Cell*, 90(1):65–75, 1997.

[193] S.M. Roe, C. Prodromou, R. O'Brien, J.E. Ladbury, P.W. Piper, and L.H. Pearl. Structural basis for inhibition of the Hsp90 molecular chaperone by the antitumor antibiotics radicicol and geldanamycin. *Journal of medicinal chemistry*, 42(2):260–266, 1999.

[194] C.E. Stebbins, A.A. Russo, C. Schneider, N. Rosen, F.U. Hartl, and N.P. Pavletich. Crystal structure of an Hsp90–geldanamycin complex: targeting of a protein chaperone by an antitumor agent. *Cell*, 89(2):239–250, 1997.

[195] D. Chen, A. Shen, J. Li, F. Shi, W. Chen, J. Ren, H. Liu, Y. Xu, X. Wang, X. Yang, et al. Discovery of potent N-(isoxazol-5-yl) amides as HSP90 inhibitors. *European journal of medicinal chemistry*, 87:765–781, 2014.

[196] K.A. Krukenberg, T.O. Street, L.A. Lavery, and D.A. Agard. Conformational dynamics of the molecular chaperone Hsp90. *Quarterly reviews of biophysics*, 44(2):229–255, 2011.

[197] M.M. Ali, S.M. Roe, C.K. Vaughan, P. Meyer, B. Panaretou, P.W. Piper, C. Prodromou, and L.H. Pearl. Crystal structure of an Hsp90–nucleotide–p23/Sba1 closed chaperone complex. *Nature*, 440(7087):1013–1017, 2006.

[198] H. Zhang, C. Zhou, W. Chen, Y. Xu, Y. Shi, Y. Wen, and N. Zhang. A dynamic view of ATP-coupled functioning cycle of Hsp90 N-terminal domain. *Scientific reports*, 5(1):1–9, 2015.

[199] J. Li, L. Sun, C. Xu, F. Yu, H. Zhou, J. Zhao, Y.and Zhang, J. Cai, C. Mao, L. Tang, et al. Structure insights into mechanisms of ATP hydrolysis and the activation of human heat-shock protein 90. *Acta Biochim Biophys Sin*, 44(4):300–306, 2012.

[200] D.B. Kokh, M. Amaral, J. Bomke, U. Grädler, D. Musil, H.-P. Buchstaller, M.K. Dreyer, M. Frech, M. Lowinski, et al. Estimation of drug-target residence times by τ-random acceleration molecular dynamics simulations. *Journal of chemical theory and computation*, 14(7):3859–3869, 2018.

[201] E. Moroni, G. Morra, and G. Colombo. Molecular dynamics simulations of Hsp90 with an eye to inhibitor design. *Pharmaceuticals*, 5(9):944–962, 2012.

[202] K. Kawaguchi, H. Saito, S. Okazaki, and H. Nagao. Molecular dynamics study on the free energy profile for dissociation of ADP from N-terminal domain of Hsp90. *Chemical Physics Letters*, 588:226–230, 2013.

[203] F. Yan, X. Liu, S. Zhang, Q. Zhang, and J. Chen. Understanding conformational diversity of heat shock protein 90 (HSP90) and binding features of inhibitors to HSP90 via molecular dynamics simulations. *Chemical Biology & Drug Design*, 95(1):87–103, 2020.

[204] G. Colombo, G. Morra, M. Meli, and G. Verkhivker. Understanding ligand-based modulation of the Hsp90 molecular chaperone dynamics at atomic resolution. *Proceedings of the National Academy of Sciences*, 105(23):7976–7981, 2008.

[205] G. Morra, M.A.C. Neves, C.J. Plescia, S. Tsustsumi, L. Neckers, G. Verkhivker, D.C. Altieri, and G. Colombo. Dynamics-based discovery of allosteric inhibitors: Selection of new ligands for the C-terminal domain of Hsp90. *Journal of chemical theory and computation*, 6(9):2978–2989, 2010.

[206] G. Vettoretti, E. Moroni, S. Sattin, J. Tao, D.A. Agard, A. Bernardi, and G. Colombo. Molecular dynamics simulations reveal the mechanisms of allosteric activation of Hsp90 by designed ligands. *Scientific reports*, 6(1):1–13, 2016.

[207] M. Simunovic and G.A. Voth. Molecular and thermodynamic insights into the conformational transitions of Hsp90. *Biophysical journal*, 103(2):284–292, 2012.

[208] B. Sohmen, C. Beck, T. Seydel, I. Hoffmann, B. Hermann, M. Nüesch, M. Grimaldo, F. Schreiber, S. Wolf, F. Roosen-Runge, et al. Nanosecond structural dynamics of the chaperone Hsp90. *arXiv preprint arXiv:2110.10483*, 2021.

[209] M. Ghorbani, S. Prasad, J.B. Klauda, and B.R. Brooks. Variational embedding of protein folding simulations using Gaussian mixture variational autoencoders. *The Journal of Chemical Physics*, 155(19):194108, 2021.

[210] Y. Wang, S. Parmar, J.S. Schneekloth, and P. Tiwary. Interrogating RNA-small molecule interactions with structure probing and AI augmented-molecular simulations. *bioRxiv*, 2021.

[211] Y. Tsuchiya, K. Taneishi, and Y. Yonezawa. Autoencoder-based detection of dynamic allostery triggered by ligand binding based on molecular dynamics. *Journal of chemical information and modeling*, 59(9):4043–4051, 2019.

[212] Y. Jin, L.O. Johannissen, and S. Hay. Predicting new protein conformations from molecular dynamics simulation conformational landscapes and machine learning. *Proteins: Structure, Function, and Bioinformatics*, 89(8):915–921, 2021.

[213] M.T. Degiacomi. Coupling molecular dynamics and deep learning to mine protein conformational space. *Structure*, 27(6):1034–1040, 2019.

[214] A. Hawkins-Hooker, F. Depardieu, S. Baur, G. Couairon, A. Chen, and D. Bikard. Generating functional protein variants with variational autoencoders. *PLoS computational biology*, 17(2):e1008736, 2021.

[215] F. Marchetti, E.a Moroni, A. Pandini, and G. Colombo. Machine learning prediction of allosteric drug activity from molecular dynamics. *The journal of physical chemistry letters*, 12(15):3724–3732, 2021.

[216] M. Ferraro, E. Moroni, E. Ippoliti, S. Rinaldi, C. Sanchez-Martin, A. Rasola, L.F. Pavarino, and G. Colombo. Machine learning of allosteric effects: the analysis of ligand-induced dynamics to predict functional effects in TRAP1. *The Journal of Physical Chemistry B*, 125(1):101–114, 2020.

[217] N. Amangeldiuly, D. Karlov, and M.V. Fedorov. Baseline model for predicting protein–ligand unbinding kinetics through machine learning. *Journal of Chemical Information and Modeling*, 60(12):5946–5956, 2020.

[218] D.B. Kokh, T. Kaufmann, B. Kister, and R.C. Wade. Machine learning analysis of τRAMD trajectories to decipher molecular determinants of drug-target residence times. *Frontiers in molecular biosciences*, 6:36, 2019.

[219] A. Dixit and G.M. Verkhivker. Probing molecular mechanisms of the Hsp90 chaperone: Biophysical modeling identifies key regulators of functional dynamics. *PloS one*, 7(5):e37605, 2012.

[220] J. Bussenius, C.M. Blazey, N. Aay, N.K. Anand, A. Arcalas, T. Baik, O.J. Bowles, C.A. Buhr, S. Costanzo, J.K. Curtis, et al. Discovery of XL888: a novel tropane-derived small molecule inhibitor of HSP90. *Bioorganic & medicinal chemistry letters*, 22(17):5396–5404, 2012.

[221] F. Vallée, C. Carrez, F. Pilorge, A. Dupuy, A. Parent, L. Bertin, F. Thompson, P. Ferrari, F. Fassy, A. Lamberton, et al. Tricyclic series of heat shock protein 90 (Hsp90) inhibitors part I: discovery of tricyclic imidazo [4, 5-c] pyridines as potent inhibitors of the Hsp90 molecular chaperone. *Journal of medicinal chemistry*, 54(20):7206–7219, 2011.

[222] T. Miura, T.A Fukami, K. Hasegawa, N. Ono, A. Suda, H. Shindo, D.-O. Yoon, S.-J. Kim, Y.-J. Na, Y. Aoki, et al. Lead generation of heat shock protein 90 inhibitors by a combination of fragment-based approach, virtual screening, and structure-based drug design. *Bioorganic & medicinal chemistry letters*, 21(19):5778–5783, 2011.

[223] H. Bekker, H.J.C. Berendsen, E.J. Dijkstra, S. Achterop, R. Vondrumen, D. Vanderspoel, A. Sijbers, H. Keegstra, and M.K.R. Renardus. Gromacs-a parallel computer for molecular-dynamics simulations. In *4th International Conference on Computational Physics (PC 92)*, pages 252–256. World Scientific Publishing, 1993.

[224] William H., Andrew D., and Klaus S. VMD – Visual Molecular Dynamics. *Journal of Molecular Graphics*, 14:33–38, 1996.

[225] R.T. McGibbon, K.A. Beauchamp, M.P. Harrigan, C. Klein, J.M. Swails, C.X. Hernández, C.R. Schwantes, L.-P. Wang, T.J. Lane, and V.S. Pande. MDTraj: A modern open library for the analysis of molecular dynamics trajectories. *Biophysical Journal*, 109(8):1528 – 1532, 2015.

[226] M. Amaral, D.B. Kokh, J. Bomke, A. Wegener, H.P. Buchstaller, H.M. Eggenweiler, P. Matias, C. Sirrenberg, R.C. Wade, and M.J.N.C. Frech. Protein conformational flexibility modulates kinetics and thermodynamics of drug binding. *Nature communications*, 8(1):1–14, 2017.

[227] S.T. Ngo, K.B. Vu, L.M. Bui, and V.V. Vu. Effective estimation of ligand-binding affinity using biased sampling method. *ACS omega*, 4(2):3887–3893, 2019.

[228] A. Nunes-Alves, D.B. Kokh, and R.C. Wade. Recent progress in molecular simulation methods for drug binding kinetics. *Current Opinion in Structural Biology*, 64:126–133, 2020.

Appendix A

Machine learning force fields and coarse-grained variables in molecular dynamics: application to materials and biological systems

This supplementary article was initially published in [20] as a review paper following a CECAM meeting co-organized by Sanofi, Ecole des Ponts ParisTech, and Sorbonne University.

Machine learning force fields and coarse-grained variables in molecular dynamics: application to materials and biological systems

Paraskevi Gkeka,[*,†] Gabriel Stoltz,[*,‡,¶] Amir Barati Farimani,[§] Zineb Belkacemi,[†,‡] Michele Ceriotti,[‖] John Damon Chodera,[⊥] Aaron R. Dinner,[#] Andrew L. Ferguson,[@] Jean-Bernard Maillet,[△] Hervé Minoux,[▽] Christine Peter,[††] Fabio Pietrucci,[‡‡] Ana Silveira,[⊥] Alexandre Tkatchenko,[¶¶] Zofia Trstanova,[§§] Rafal Wiewiora,[⊥] and Tony Lelièvre[*,‡,¶]

†*Structure Design and Informatics, Sanofi R&D, 91385 Chilly-Mazarin, France*

‡*Ecole des Ponts ParisTech, France*

¶*Matherials project-team, Inria Paris, France*

§*Carnegie Mellon University, USA*

‖*Laboratory of Computational Science and Modelling, Institute of Materials, École Polytechnique Fédérale de Lausanne, Switzerland*

⊥*Sloan Kettering Institute, USA*

#*Department of Chemistry, The University of Chicago, Chicago, Illinois 60637, USA*

@*Pritzker School of Molecular Engineering, 5640 South Ellis Avenue, University of Chicago, Chicago, Illinois 60637, USA*

△*CEA-DAM, DIF, France*

▽*Structure Design and Informatics, Sanofi R&D, 94403 Vitry-sur-Seine, France*

††*University of Konstanz, Germany*

‡‡*Sorbonne Université, UMR CNRS 7590, MNHN, Institut de Minéralogie, de Physique des Matériaux et de Cosmochimie, 75005 Paris, France*

¶¶*Department of Physics and Materials Science, University of Luxembourg, L-1511 Luxembourg*

§§*School of Mathematics, The University of Edinburgh, UK*

Abstract

Machine learning encompasses a set of tools and algorithms which are now becoming popular in almost all scientific and technological fields. This is true for molecular dynamics as well, where machine learning offers promises of extracting valuable information from the enormous amounts of data generated by simulation of complex systems. We provide here a review of our current understanding of goals, benefits, and limitations of machine learning techniques for computational studies on atomistic systems, focusing on the construction of empirical force fields from ab-initio databases and the determination of reaction coordinates for free energy computation and enhanced sampling.

1 Introduction

The atomistic representation of physical systems offers a precise description of matter. Simplified models based on coarse-grained (CG) representations offer an alternative that can significantly aid in the understanding of the physical properties of the systems under consideration. Such representations can also be used as a surrogate model for enhanced sampling methods (e.g. sampling large conformational changes using reduced models).

Both in the case of biochemical systems as well as in materials, a CG description can be based on distance metrics for structural clustering,[1] as well as on reaction coordinates: for instance, the conformational changes of a complex molecule can be modeled by a few key functions of the atomic positions, while a phase transition can be described by a change of the average atomic coordination or box shape. In condensed matter physics, atomic descriptors are employed to summarize the key features of atomic configurations in order to predict forces and energies.[2,3]

In the past, reaction coordinates were defined using empirical methods and chemical intuition, while more systematic approaches were employed for the definition of atomic descriptors.[4,5] During the last decade, the return and rise of Machine Learning (ML) techniques

have initiated many efforts focusing on automating the definition of reaction coordinates or descriptors that are able to successfully describe the underlying atomic systems.[6–9] The employed methods, both supervised and unsupervised, vary. The most commonly used methods for the identification of reaction coordinates include Principal Component Analysis (PCA),[10] diffusion maps,[11,12] and auto-encoders.[13–16] For atomic descriptors, common choices are based on a judicious use of adjacency matrices and their generalizations, or on a large set of feature vectors based on a set of basis functions.

We are witnessing many current attempts for automatically devising intuition-free collective variables, in particular for drug discovery applications.[13,17] Although the initially very high hopes raised by numerical potentials are now mitigated, there have been quite a few systematic studies on the quality of the descriptors obtained by these approaches.[18,19]

A recent CECAM (Center Européen de Calcul Atomique et Moléculaire) discussion meeting[1] brought together a diverse audience of 29 participants from various scientific fields, including chemistry, drug design, condensed matter physics, materials science, and mathematics, to exchange about state-of-the-art techniques for automatically building coarse-grained information on molecular systems. In particular, we believe that the viewpoint and experience of condensed matter physicists in devising atomic descriptors could prove useful insights in devising reaction coordinates in a more systematic way. Mathematics offer, in this framework, a common language for the discussion. One distinctive feature of this CECAM meeting is that the emphasis was on the technical details of the underlying numerical methods.

In the current review, we discuss the following highlights of the meeting:

- **Machine learning force fields and Potential of Mean Force.** ML techniques have been recently employed in the development of force field (FF) parameters based on quantum-mechanical calculations. More generally, ML techniques can be used to define a surrogate model of any quantity that could be obtained from a quantum chem-

[1] See the conference website `https://cermics-lab.enpc.fr/cecam_ml_md/`

ical calculation, as a function of atomic coordinates (e.g. NMR chemical shieldings, IR dipole moments, ...), making it possible to obtain an accurate estimate of experimental observables. Such models are beginning to find merit due to their accuracy and versatility. In Section 2, we review the factors that play an important role in the accuracy and transferability of a force field. Specifically, we report the importance of the input database and the choice of the regression method for the force field construction. The use of prior physico-chemical knowledge in this construction of ML potentials is also discussed.

- **Dimensionality reduction and identification of meaningful collective variables.** Another important issue discussed during the CECAM meeting is the dimensionality reduction and the identification of meaningful CVs using ML techniques (see Section 3). We considered the case when this identification relies on a database which covers the full configuration space of the system under study (obtained for instance by high temperature sampling, steered molecular dynamics, etc), and the case when the data is restricted to a metastable state. Once a reaction coordinate is found, the question of devising a good effective model along this coordinate can also be addressed using machine learning techniques: either approximate free energies (for example by potentials involving only 2, 3 or 4 body interactions), or approximate the terms in the effective dynamics, namely the drift, diffusion coefficient, metric tensor and memory terms, for example using projections *à la* Mori-Zwanzig.

- **Applications of machine learning techniques in biological systems and drug discovery.** In Section 4, we discuss some "real world" applications, where MD simulations coupled with ML techniques enable us to understand the biological complexity at the atomic and molecular levels and provide us with interesting insights about the thermodynamic and mechanistic behaviour of biological processes. In particular, we highlight some examples of ML approaches applied in clustering and construction of

Markov state models, we describe how ML methods facilitate enhanced sampling protocols through the use of efficient CVs and we mention some possible applications in the drug discovery process. These examples illustrate the current state and potential of the field of ML in the study of biological systems and drug discovery.

We close the review with some perspectives in Section 5.

2 Machine learning force fields and Potential of Mean Force

Interactions between atoms are often modeled using empirical potentials with some prescribed functional forms, as suggested by physical considerations. This provides computationally cheap (with a cost scaling linearly with the number of atoms) but somewhat inaccurate potentials. On the contrary, ab-initio approaches provide more reliable, less uncertain force fields, at the expense however of a large computational cost (typically scaling as the number of electrons to the power 3). The promise of machine learning for force field computations is to predict forces and energies with accuracy arbitrary close to the level of ab-initio approaches,[20] but with a much smaller computational cost and scaling as a function of the number of atoms. Ideally, these force fields should be able to describe chemical reactions. This is typically done in practice by setting up a database of configurations with associated forces and energies, summarizing atomic configurations through some descriptors of the local environment, and predicting the forces and energies from these descriptors through a function which has been trained by some (nonlinear) regression procedure to provide good results on the database. The resulting potential is called a "numerical potential".

There are three different factors to discuss the success of ML methods, whose relative importance depend on the aims of the user: accuracy, computational cost, and transferability. The latter concept means that a numerical potential computed for a given material in a given thermodynamic range, can be used outside the fitting domain – for instance because it is

used for other materials and systems than the ones it was trained on, and/or in a different thermodynamic range than the one considered for the configurations in the database.

We first discuss in this section elements on the choice of the database, see Section 2.1. We next present various choices for the descriptors and for associated ML regression methods, see Section 2.2. We then discuss in Section 2.3 how to incorporate physical insights in order to improve ML techniques, and we give some perspectives in Section 2.4. We end the section by mentioning how ML approaches can also be used to derive CG potentials, see Section 2.5: in this perspective, empirical force fields for all atom models are seen as the reference (they are the counterpart of ab-initio databases in this context), and effective force fields describing the interaction of coarse-grained variables are sought.

2.1 Setting up a database

One of the key factors that affects the accuracy and transferability of a force field is the database used for its construction. This database defines the envelope of confidence (applicability domain) for the potential as the subsequent regression method is efficient in interpolation. It is often the case that a numerical potential has a poor transferability. Therefore, for condensed matter systems, the database should sample the region of interest, i.e., the thermodynamic conditions where the potential is going to be used. However, this representative part of the configurational space covers only a small fraction of the overall available space. Hence, a systematic exploration is impossible, and physical intuition is often used to constrain the search of new interesting configurations for learning. This makes the construction of the database a rather laborious process. A first application of 'active learning' in this process, also still hand made, is proposed by Artrith and Behler in Ref. 21: two different neural networks are optimized on the same database and, in case their predictions on a new configuration differ too much this configuration should be included in the database. Active learning, based on outlier detection (i.e., definition of a metric to detect parameters corresponding to some extrapolation) is now routinely employed during the database con-

struction.[22] In this way, force field accuracy can be improved during the training procedure[23] and the domain of applicability could be extended.[24] The bottom line is that 'on the fly' learning[25] enables to perform optimization and prediction at the same time.[26] Typically, a trade-off has to be found between the transferability of a potential (its robustness to changes in the database) and its accuracy.

The representation of the database should also be meaningful: finding a proper space for this representation allows to define an envelope of confidence for the potential. When the potential is used, each new configuration can rapidly be plotted in this space to check if it belongs to the database envelope (applicability domain), i.e., if the potential is used in interpolation or in extrapolation. It then becomes a useful criterion for outlier detection.

What is globally accepted is that the methods should systematically be validated on test data, different from the training data. In any case, one should be very careful about the quality of the model for extrapolation.

2.2 Descriptors and regression methods

We present in this section the technical approaches to fit a potential on a database. We distinguish the representation of the atomic configurations through descriptors, and the subsequent regression allowing to fit the parameters of the chosen model. Typically, a very simple descriptor, based on physical/chemical intuition or moment estimates for atomic densities, should be combined with a complex regression such as a neural network; on the other hand, more educated descriptors, for instance based on convolutional neural networks and a scattering transform,[27] can be fed into quite simple (bi)linear regression models.

2.2.1 Representing atomic configurations

It is almost never appropriate to use the Cartesian coordinates of atoms in a structure as the input of a machine-learning scheme,[28] because Cartesian coordinates do not conform with the invariance of the target properties, e.g. permutation of the indices of identical

atoms, rigid translations, rotations and reflections. For this reason, several different schemes have been devised to map atomic configurations onto vectors of features that fulfil these symmetry requirements. Usually, it is desirable for this mapping to be differentiable and smooth, particularly in applications where one needs to compute forces as the derivative of a machine-learning potential or CG force field.

One can roughly partition methods to represent atomic configurations into two classes. *Descriptors* are often highly simplified representations of a structure, usually of much smaller dimensionality than the number of degrees of freedom and incorporating some degree of chemical intuition, or a heuristic understanding of the behavior of the system being studied. Cheminformatics schemes to characterise the connectivity of a molecule, such as SMILES[29] strings, are useful when dealing with databases of organic compounds. Steinhardt parameters[30] are often used to characterize the coordination of liquids and solids. Backbone dihedral angles, or more complex indicators of secondary structure[31] can be utilized to discard information on the side chains of polypeptides. The dimensionality reduction that is intrinsic to this family of methods typically induce loss of information, which may be desirable (when it discards irrelevant details) or problematic: in the latter case, it is often more effective to use a more complete description and then proceed with an automatic dimensionality reduction algorithm, some of which will be discussed in Section 3.

Representations, on the other hand, attempt to provide a complete description of a configuration. This family of features is typically used when building regression models for energy and properties. Most of the time (particularly for condensed-phase applications, but often also for isolated molecules) representations are not built for an entire structure, but are instead used to describe atom-centered environments. This is advantageous, because - by representing a structure as a collection of compact groups of atoms, and assuming that the overall property can be computed as a sum of local contributions - it becomes possible to train models that can be easily transferred between systems of different sizes, and from simple to more complex configurations. Many of these systematic representations

- including e.g., SOAP (bi)spectrum,[32] Behler-Parrinello symmetry functions,[33] moment tensor potentials,[18] FCHL kernels[34] - can be seen as projections on different basis of n-body correlation functions,[35] and offer a systematic and completely general way to describe atomic configurations, that can be applied equally well to condensed phases, gas-phase molecules and polypeptides.[36]

2.2.2 Choosing the regression method

Once the atomic descriptor has been chosen, the choice of the regression method to determine the force field is crucial and greatly depends on the system under study.[37] A distinction should be made between learning based on neural networks, and other regression methods based on kernels or (bi)linear methods. Training neural networks is a complex non-convex optimization problem in very high dimension (generally thousands of parameters are needed to parameterize the networks under consideration). Already the computation of the gradient of the objective function is non trivial and relies on clever numerical tricks, such as back-propagation. Kernel-based methods or (bi)linear regression techniques lead, on the other hand, to much better behaved optimization problems, which can even be solved analytically through some matrix inversion on the Euler equation defining the minimizer.

The choice of the regression method also determines whether error estimators are available. For example a variance can be associated with a prediction when a kernel method is used, whereas error quantification is harder using neural networks. Moreover, the robustness of the potential depends on the regression method and its associated regularization (used to alleviate overfitting issues). A simple (bi)linear method may be less accurate but more robust. It may also be sufficient if the descriptors already provide enough information on the system, as is the case for the descriptors obtained via convolutional neural networks in Ref. 27.

In principle, both neural network (NN) and nonlinear kernel regression models are sufficiently sophisticated to obtain a trustworthy representation of scalar potential-energy sur-

faces (PES) or vector force fields of arbitrary complexity. However, in practice, choices have to be made for the similarity measure between atomic configurations (in both kernel regression methods and NN) or for the architecture of the neural network. The optimal choices are not the same for different systems, i.e., descriptors/parameters that work well for solids are not easily transferable to biological molecules and vice versa. Hence, many ML developments are currently specific to either organic molecules or materials. That being said, there is currently a growing interest in understanding the advantages and limitations of the different existing approaches[18,27,32,33,38–41] and developing truly general frameworks for learning complex PES or force fields that work seamlessly for both organic and inorganic matter.

2.2.3 Current methods and their performances

We list some key methods in Table 1. The first successful ML approaches were developed to describe PES of defectless materials and their surfaces[32,33,38] with the goal to enable efficient and accurate Molecular dynamics (MD) of large supercells of elementary or binary materials. The Behler-Parrinello NN approach[33] or the kernel-based GAP approach of Csanyi[32] are both able to achieve accuracies of 1-2 meV/atom for some solids (C, Si, Cu, TiO2, among others). There are several key differences between these two methods, the main ones being the NN vs kernel approach and the different similarity measures between atomic configurations. Both approaches typically require on the order of tens to hundreds of thousands reference calculations at the DFT level for constructing the training dataset, in order to achieve 1-2 meV/atom accuracy. Recently, PES-fitting methods based on deep networks have also been developed.[41,42] These approaches often do not require any *a priori* definition of the similarity measure; they are instead able to learn the similarity measure from the training data.

Constructing ML models for organic molecules is a field that faces somewhat different challenges compared to ML models for solids and materials. While DFT calculations are often deemed to provide sufficiently accurate reference data for solids, this is not the

Table 1: Summary of some key learning methods for force field (FF) development.

Method	**Short description**	**Ref.**
Kernel-based Gaussian approximation potentials (GAP)	Combines a structural descriptor and a kernel establishing the link between structure and energy	32
Behler-Parrinello NN	Feed-forward NNs for each atom. The potential energy is constructed as the sum of local atomic energies	33,38
Deep NN (DTNN)	No a priori similarity definition needed, similarity is learned	41,42
Permutationally-invariant polynomials (PIP)	Uses polynomials of Morse variables in fitting PES	39,43
Gradient-domain ML (GDML)	Learns an explicit FF and obtains the PES via integration	7,40

case for organic molecules. The "gold standard" is coupled cluster CCSD(T) computations. Quantum-chemical CCSD(T) calculations are however computationally expensive and it is only possible to carry hundreds of such calculations even for simple molecules such as aspirin. Early successful nonlinear PES models were based on permutationally-invariant polynomials (PIP).[39] More recent developments include the so-called gradient-domain machine learning (GDML) approach[7,40] for constructing molecular force fields. The GDML approach learns an explicit force field and obtains the PES via integration, instead of the more conventional approach to learning a PES and then taking its gradient to drive MD. This has two advantages: (i) the usage of an explicit Hessian kernel that provides the maximum flexibility, minimizes noise and prevents artifacts between forces and energies in the learning process; (ii) a significant gain in data efficiency, since globally accurate force fields for small molecules (accuracy of 0.2 kcal/mol and 1 kcal/mol/Å) can now be constructed using only a few hundred molecular conformations for training. This data efficiency currently enables

the construction of essentially exact force fields for molecules with up to 30-40 atoms.[7]

2.3 Synergy between physics, chemistry, mathematics and ML approaches

ML approaches used to construct accurate PES and force fields have already been successful and have enabled simulations of molecules and materials that were previously considered impossible. Ultimately, it would be worthwhile to achieve an optimal balance between physics-based models and ML approaches to enable not only faster and more accurate simulations, but also obtain insights into interactions of complex quantum-mechanical molecules and materials. For example, the GAP, Behler-Parrinello, GDML, and PIP approaches discussed above already incorporate translational, rotational, and permutational symmetries of molecules and materials in their internal representation of atomic interactions. Such symmetries were also made precise in the mathematical literature.[18] In addition, by learning simultaneously energy and forces such that the latter are (minus) the gradient of the former, all of these methods enforce exactly energy conservation.

However, many more physical symmetries can and should be incorporated in ML approaches. For example, exact constraints are known for asymptotic forms of atomic interaction potentials. Also, some analytic and empirical results are known for series expansions of interatomic potentials. Finally, there are mathematical results which provide rigorous statements on the behavior of the potential energy functions in terms of the locality of the interactions.[19] The incorporation of such prior knowledge could improve the efficiency and accuracy of ML potentials and ultimately also lead to novel analysis tools that offer new insights into the complex nature of atomic interactions.[44]

It is also worth noting that electronic interactions in complex molecules and materials can be rather long-ranged. For example, electrostatic interactions and plasmon-like electronic fluctuations in molecules and nanostructures can lead to interatomic potentials extending to at least 20-30 nanometers.[45,46] Most current ML models explicitly or implicitly cut off

interactions at an interatomic distance of 5-6 Å. Hence, by construction, these ML approaches are not able to capture interactions extending over larger length scales. For this reason, it is ultimately necessary to couple ML approaches that excel at capturing complex short-range chemical bonding with explicit physics-based approaches to non-covalent interactions. It is important to note that such physics-based models can also employ ML approaches to learn short-range interaction parameters based on datasets of electrostatic moments and polarizabilities. The recently developed IPML approach lies the foundation for unifying ML force fields and physics-based interatomic potentials.[47] An alternative approach based on the definition of structure representations that incorporate long-range correlations with the correct asymptotic behavior[48] can simplify the simultaneous description of the multiple length scales contributing to molecular interactions.

2.4 Perspectives for ML approaches to the determination of force fields

We gather in this section some mathematical and numerical perspectives, as well as open problems, on ML methods for force fields:

- A first perspective is the use of ML to learn the difference between already acceptable empirical force fields and DFT models, as some form of preconditioning. Such an approach greatly depends on the regression method. For example, for kernel methods, it has been shown that a potential can be built on top of pre-existing two-body and three-body classical potentials, improving the overall accuracy.[49,50] On the contrary, fitting differences between a good classical potential and an ab-initio potential with a linear regression yields very poor results, since the difference is small (almost noisy) and rugged (not smooth). It is observed that a simpler starting guess, such as the Ziegler–Biersack–Littmark potential,[51] yields better results, since this increases the numerical stability and improves the accuracy.

- A question related to the robustness of these learning techniques is whether it would make sense to optimize potentials on a Pareto curve, where various properties of interest are weighted in different manners in the cost function. Indeed, the optimization is usually performed on a multi-objective cost function (including energy, force, stress, and sometimes bond distances, ...). The so-obtained potential is a result of the user arbitrary choice of the weighting parameters – infinitely many 'optimal' potentials can be obtained depending on the choice of the weights. The naturally rising question here is: is it possible to have a unified way of defining cost functions?

- An important practical concern is the sensitivity of the learnt parameters relatively upon the data (for instance depending on the fraction of elements used for training vs. testing).

- Another more theoretical question is: What is the numerical stability induced by machine learning potentials on the time integration of Hamiltonian dynamics and its variations? Indeed, some preliminary results suggest that machine learning potentials may be smoother than current empirical potentials.

- For reasons which remain to elucidate, predicting intensive (as opposed to extensive) properties seems to be very challenging.

2.5 Bottom-up coarse-graining force fields: From PES to FES

A classical particle-based coarse grained (CG) simulation model, where several atoms are grouped together, can be viewed as a reduction of the dimensionality of the classical phase space (see Figure 1). It requires the determination of an effective Hamiltonian that allows the model to explore the phase space in the same way as an atomistic simulation would. Thus, in the so-called bottom up coarse-graining strategies, the interactions in the CG model are devised such that an accurate representation of a (known) atomistic sampling of the configurational phase space (mapped to the CG representation) is achieved. These methods use

the underlying multidimensional potential of mean force (PMF) derived from the atomistic simulation data as parameterization target, i.e., they try to reproduce a (typically high-dimensional) free-energy surface (FES) as opposed to a PES. Naturally, this is of particular relevance to the simulation of soft matter problems such as liquid state systems, soft materials and biological systems, where entropic effects, disorder and heterogeneity dominate the overall properties of the system.

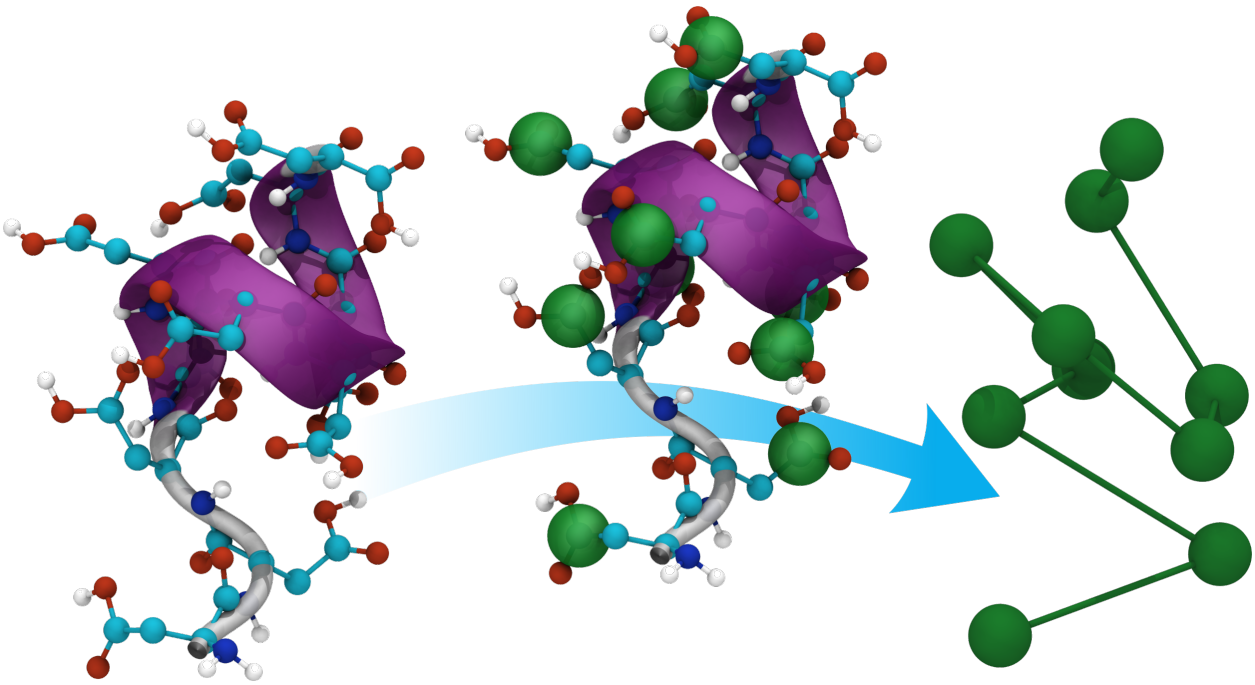


Figure 1: Particle-based coarse-graining: high dimensional free energy surfaces (FES) can be extract from atomistic data and used as a basis for CG models.[52,53]

Free energies and potentials of mean force are not a direct output of a MD simulation. They can be calculated by Boltzmann inversion of a (high-dimensional) probability density distribution obtained from sampling configurations in phase space or from mean forces acting on the interaction sites in the CG representation. In the past, several bottom-up coarse-graining methods have been derived which - while all aiming for an effective Hamiltonian that approximates a multidimensional PMF/FES - differ in terms of both the actual parameterization target (multidimensional PMFs/probability density distributions, structure functions as low-dimensional representations of these PMFs; mean forces in the direction of

selected CVs or relative entropies) and the type of CG interactions which are typically represented by low-dimensional potentials, i.e., pair interactions, or three-body interactions).[54–58] Since these coarse-graining methods derive interactions from atomistic reference simulations, they are intrinsically data driven. Consequently, ML-based approaches yield new types of reference atomistic data and new types of CG interactions and parameterization methods. On the one hand, ML methods can be used to determine dimensionality-reduced representations of the phase space and to derive or validate CG models by matching the sampling of a (relatively complex) FES as opposed to low-dimensional target functions/properties. On the other hand, ML methods can also be employed to identify suitable CVs that describe the states and the dynamics of a system, which can then either be directly used in the CG potentials or be employed to identify optimal CG representations and learn CG interactions. This is discussed at length in Section 3.

Following the methodology of inferring all-atom potential energy functions from corresponding quantum mechanical data, John and Csanyi have extended the Gaussian Approximation Potential (GAP-CG) approach to coarse-graining of simple liquid systems.[59] In this case, the many-body PMF is described via local multibody terms, based on local descriptors and multidimensional functions which are determined by Gaussian process regression from atomistic training data (instantaneous collective forces or mean forces). In a similar vein, Zhang et al. developed a scheme, called the Deep Coarse-Grained Potential (DeePCG), which uses a NN to construct a many-body CG potential for liquid water.[60] The network is trained with atomistic data in a manner similar to the force matching in the multi-scale coarse-graining method,[61] and in such a way that it preserves the natural symmetries of the system. While the described two methods are related to the force-matching type of bottom-up coarse-graining and use ML to significantly extend the complexity of the CG interactions, Lemke and Peter follow a different strategy.[52] A NN is used to extract high-dimensional FES from atomistic MD simulation trajectories. The NN is trained to predict conformational free energies by creating a classification problem between real MD confor-

mations and fake conformations of a known distribution. With such a classification based procedure it is possible to train the NN to return probability densities without requiring any binning or normalization – which circumvents the problem of binning in high dimensional space.[62] By using the NN probability densities directly in a Monte Carlo type of sampling of conformations, a (relatively) high-dimensional FES is thus used as effective CG Hamiltonian. This NN network model was successfully tested for several homo-oligopeptides.[53] By employing a convolutional NN architecture, the NN model could be simultaneously trained on data of different chain lengths and could even make meaningful predictions for polymers with chain lengths different from the ones in the training data. Thus, such an approach is promising for the simulation of polymer systems where naturally training data are restricted to chain lengths that are shorter than the intended polymers.

Coarse-graining of potential energy functions into free energy type interactions has a well founded statistical interpretation. A difficult question is however whether some dynamical properties are also preserved in this coarse-graining process, and to which extent.

3 Dimensionality reduction and identification of collective variables

The objective of this section is to discuss various techniques to identify collective variables. After some general considerations in Section 3.1, we first present the main two ideas to build collective variables in Section 3.2, namely looking for high-variance or slow degrees of freedom. We then discuss how this can be used to enhance the sampling of the canonical ensemble on the example of diffusion maps in Section 3.3, before discussing dynamical aspects in Sections 3.4 and 3.5.

3.1 General considerations

Molecular systems are characterized by the fact that their long-time dynamical behavior is typically governed by a small number of emergent collective variables (CVs).[63–65] These collective modes arise from cooperative couplings between the constituent atoms induced by interatomic forces (e.g., covalent bonds, electrostatics, van der Waals interactions) and possibly external fields (e.g., electric fields, hydrodynamic flows), and which render the effective dimensionality of the system far lower than that of the full-dimensional phase space in which the system Hamiltonian and equations of motion are formulated.[64,65] In a dynamical systems sense, the long-time evolution of the system is restrained to a low-dimensional attractor or intrinsic manifold and its dynamics over these time scales may be described within the Mori-Zwanzig projection operator formalism as evolving within a subspace of slow collective variables to which the remaining degrees of freedom are effectively slaved.[64]

Traditional unbiased MD is not able to efficiently explore the whole kinetic landscape with time scales spanning over orders of magnitude, from picoseconds to milliseconds. In this scenario, one relies on extensive simulations together with some clever strategy to escape metastable states. Such a strategy can only be devised if one is able to identify what defines a "long-lived" state, which is equivalent to discovering meaningful collective variables (CVs) or reaction coordinates.[66]

The methods described below aim at finding these CVs or states. As will become clear later, depending on the objective, the focus may be different: gain insight/intuition on the system, bias to exit metastable states, compute a free energy profile, set up a coarse-grained dynamics simulation, cluster/classify configurations, etc.

3.2 Data-driven discovery of high-variance and slow collective variables

The inherently multi-body and emergent nature of the CVs means that they are exceedingly challenging to intuit for all but the most trivial systems, and data-driven techniques present a powerful means to systematically estimate them from molecular simulation data. The origins of this data-driven approach can be traced back to pioneering work in the early 1990's by Toshiko Ichiye and Martin Karplus,[67] Angel Garcia[68] and Andrea Amadei, Antonius Linssen and Herman Berendsen[69] who applied PCA to molecular simulations of protein folding. Since that time there has been an explosion of interest in the use of data science and machine learning techniques to estimate CVs from molecular simulation data and the subsequent use of these CVs to inform new understanding, perform molecular design, and guide enhanced sampling.

Data-driven CV discovery typically employs unsupervised learning techniques that seek low-dimensional parameterizations of the geometry of the data in the high-dimensional phase space of atomic coordinates.[70] This procedure can usually be cast as an optimization problem that maximizes some objective function, or equivalently minimizes some loss function, over the data. The techniques can be categorized into linear and nonlinear methods. Linear techniques are restricted to discovering CVs that are linear combinations of the input features, whereas nonlinear techniques can discover more general nonlinear functional relations. The more powerful and general nonlinear techniques are typically better suited to the estimation of the complex emergent CVs in molecular systems, but linear techniques should not be discounted since they are typically more robust, interpretable, and less data hungry, and can also admit nonlinearities through feature engineering or the kernel trick.[71] The importance of the choice of features in which the molecular system is represented to the CV discovery tool should not be underestimated. Feature sets that contain and foreground the important molecular behaviors and respect fundamental symmetries (e.g., translation, rotation, permutation) can be critical to the success of CV discovery (particularly in the

case of linear techniques), whereas poor choices that mask or discard essential information or contain spurious symmetries can easily produce poor performance. What constitutes a good choice of feature set is strongly system dependent and is typically reliant on some combination of intuition, experience, and exploratory trial-and-improvement. We refer for example to Ref. 72 for a discussion on the importance of the choice of the representation of the data.

Although the details and specifics differ, most CV discovery techniques can be placed in one of two categories: those that seek high-variance CVs and those that seek slow CVs (see Figure 2).

High variance CVs maximally preserve the configurational variance in the high-dimensional data upon projection into the low-dimensional space spanned by these CVs. Slow (i.e., maximally autocorrelated) CVs define a low-dimensional space that maximally preserves the

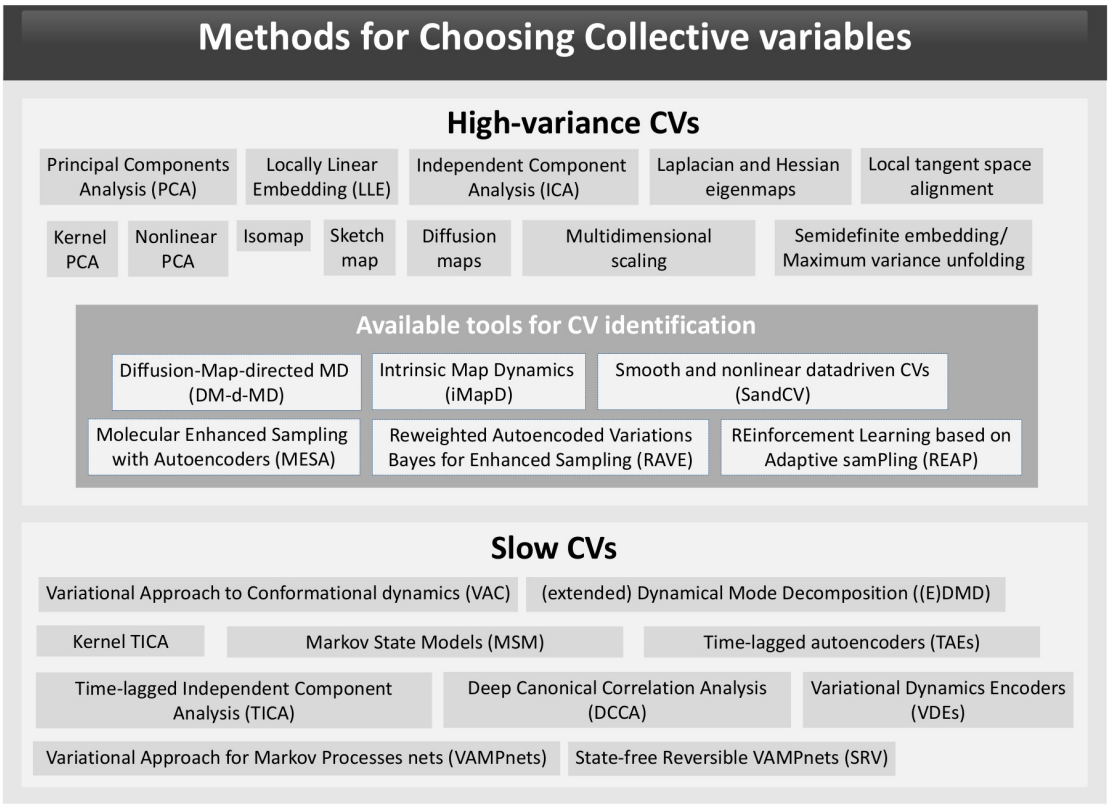


Figure 2: Representative methods for CV identification. All related citations are in the main text.

long-time kinetics of the system. Frequently the slow and high-variance collective modes are related, but this is not always the case. Importantly, the estimation of slow CVs requires data arranged in time series (e.g., MD trajectories) whereas the estimation of high-variance CVs can be applied to data sampled without temporal ordering (e.g., Monte Carlo trajectories). Notice however that methods exist to recover dynamical information according to some artificial dynamics (e.g. reversible purely diffusive dynamics) upon non-time ordered data to render it amenable to temporal analysis techniques.[73]

Let us also mention that recent advances in deep reinforcement learning (DRL) in robotics opens up new avenues for deploying DRL to atomic and molecular systems. In all DRL algorithms, a reward function, state and action space should be defined. In atomic systems, state space can be atomic coordinate, action space can be the movement of atoms, and reward can be defined as energy. DRL can be suitable replacement for finding transition paths and can potentially be used to strengthen the string or nudged-elastic-band method.[74,75]

Before giving more details about the high-variance and slow CVs, let us mention that a widespread definition of an optimal *scalar-valued* reaction coordinate in the rare event-field is the committor function, i.e., in a system with two metastable states, the probability that a given atomic configuration will evolve towards the products before reaching the reactants. Such probability can in principle be estimated by generating a huge number of MD simulations from each configuration of interest: even if such a procedure cannot be applied in practice to the whole configuration space, the committor represents an ideal reaction coordinate in some sense (we refer the reader to Ref. 76 or 77 (p.126) for example) and provides tests and optimization strategies for candidate CVs .[5,17,76,78–80]

3.2.1 High-variance CV estimation

The best known high-variance CV estimation technique is PCA,[10] also known as the Karhunen-Loève transform,[81–84] or proper orthogonal decomposition.[85,86] This approach discovers an orthogonal transformation of the input data to define a hyperplane approximation that pre-

serves most of the variance in the data. Popular nonlinear techniques for high-variance CV estimation include kernel and nonlinear PCA,[87–90] independent component analysis (ICA),[91] multidimensional scaling,[92] sketch map[93] locally linear embedding (LLE) ,[94,95] Isomap,[96–98] local tangent space alignment,[99] semidefinite embedding / maximum variance unfolding,[100] Laplacian and Hessian eigenmaps,[101,102] and diffusion maps .[11,103] These approaches differ in their mathematical details, but can be broadly conceived of as nonlinear analogs of principal component analysis that pass curvilinear manifolds through the data to define nonlinear projections into a low-dimensional subspace spanned by the learned CVs. Specialized techniques for molecular simulations that integrate iterative high-variance CV discovery and accelerated sampling of configurational space have been developed in recent years.[13–15,104–114]

The techniques described above can be coupled with enhanced sampling methods, which use the uncovered CV's to help the system leave metastable states. In this case, one actually relies on CV estimates based on partial sampling.[73] Let us describe a few methods in that direction.

Diffusion-map-directed MD (DM-d-MD) uses diffusion maps to identify CVs spanning the range of explored system configurations and then initializes new simulations at the frontiers of this domain to drive sampling of new system configurations.[113,114] Intrinsic map dynamics (iMapD) employs diffusion maps to construct a nonlinear embedding of the high dimensional simulation trajectory and then uses boundary detection algorithms with a local principal components analysis to extrapolate into new regions of phase space at which to seed new simulations.[105] The Smooth And Nonlinear Data-driven Collective Variables (SandCV) approach identifies nonlinear CVs using Isomap, expands them within basis functions centered on a small number of landmark points, and then passes this parameterization to the adaptive biasing force accelerated sampling technique to drive sampling along these coordinates.[109] Molecular enhanced sampling with autoencoders (MESA) employs autoencoding neural networks to discover nonlinear CVs for enhanced sampling without the need for approximate basis function expansions.[13,14] Reweighted Autoencoded Variational Bayes

for Enhanced Sampling (RAVE) employs variational autoencoders to discover nonlinear CVs that are compared at the level of their probability distributions with an ensemble of physical candidate variables to identify physical coordinates for accelerated sampling.[15] Recently, Tiwary and co-workers extended their approach using the past–future information bottleneck principle on a novel deep neural network (linear encoder–stochastic decoder model).[115] Interestingly, as the authors mention, the addition of a linear encoder part helps preserving the interpretability of the CV. REinforcement learning based Adaptive samPling (REAP) employs reinforcement learning to identify the dynamically-varying relative importance in driving exploration of configurational space of each CV within a candidate set and then adaptively seeds new simulations from configurations with high reward functions.[104]

3.2.2 Slow CV estimation

The identification of slow CVs is valuable and informative from many perspectives. From a mechanistic perspective, these CVs reveal the collective modes that dictate the metastable states of the system and the transitions between them. From a design perspective, they can offer a blueprint for the structural, thermodynamic, and dynamic properties of the system. From an enhanced sampling perspective, they provide good variables in which one can apply biases to accelerate barrier crossing and improve exploration of configurational phase space.

A number of approaches have been proposed to analyze MD time series to estimate slow CVs. The theoretical basis for these techniques is founded in the variational principle of conformational dynamics (VAC),[116] or in the (extended) dynamical mode decomposition ((E)DMD)[117,118] that, respectively, frame the recovery of the slow CVs as a variational optimization or regression problem.[16,119] Shortly, VAC estimates the slowest modes as linear combinations of *a priori* defined basis functions of the input coordinates. In Time-lagged independent component analysis (TICA) these basis functions are the coordinates themselves.[116,120–126] In Markov state models, the slow CVs are approximated in a basis of indicator functions defined over the data[119,127] (see also the recent special issue Ref. 128 for

the latest developments on Markov state models). Perron cluster analysis can be used to reduce the large number of states uncovered by clustering methods along the trajectory, to a few metastable states, see Ref. 129–131. Combining TICA with the kernel trick yields kernel TICA (kTICA) that is capable of approximating the slow CVs with nonlinear functions of the input features.[116,132] Deep canonical correlation analysis (DCCA),[133] the variational approach for Markov processes nets (VAMPnets),[134] and state-free reversible VAMPnets (SRV)[135] all employ Siamese neural networks to learn nonlinear featurizations of the input coordinates as basis functions with which to approximate the slow CVs. Time-lagged autoencoders (TAEs) employ time-delayed autoencoding neural networks to learn slow CVs into which the molecular trajectory can be projected (i.e., encoded) and also used to predict the system state at the next time increment (i.e., decoded).[16] Variational dynamics encoders (VDEs) are similar to TAEs but employ a variational as opposed to traditional autoencoding architecture that introduces stochasticity into the decoding of the learned CVs.[136,137] In a very recent study, Bonati et al. take a step further their initial Variationally Enhanced Sampling (VES) method[139] by representing the biasing potential using a neural network, which makes it unnecessary to resort to CVs.[138]

Enhanced sampling can be conducted in the learned slow CVs in a similar manner to that in the high-variance CVs, but the application of artificial biasing potentials perturbs the true system dynamics and subsequent applications of slow CV estimation techniques to the biased data must compensate for this effect.[140–142] Moreover, it should be noted that, even though in some cases such as the study of biomolecular systems, we are interested in rare events and slow CVs are optimal, there are cases where the identified slow CVs have implied timescales that are beyond the phenomenon-relevant scales. In this scenario, a non-optimal solution would be to correct the kinetic model afterwards by removing undesired modes. As Husic and Noé pointed out, such a strategy might become impractical when evaluating multiple candidate models with nonequivalent modes. As a more general and automatic solution, they propose to use deflation techniques to eliminate the leading slow CVs when

these do not correspond to the kinetic processes of interest (e.g., folding).[143]

3.3 Enhanced sampling using local and global diffusion maps

Using the illustrative example of diffusions maps, we discuss in this section how to use the proposed reaction coordinate to enhance sampling and somehow perform some extrapolation procedure. Diffusion maps are a dimensionality reduction technique which allows for identifying the slowly-evolving principal modes of high-dimensional molecular systems.[11,12] It does so by computing an approximation of a Fokker-Planck operator on the trajectory point-cloud sampled from a probability distribution (typically the Boltzmann-Gibbs distribution corresponding to prescribed temperature). The construction is based on a normalized graph Laplacian matrix. In an appropriate limit, the matrix converges to the generator of overdamped Langevin dynamics. The spectral decomposition of the diffusion map matrix thus yields an approximation of the continuous spectral problem on the point-cloud[144] and leads to natural CVs.

Since the first appearance of diffusion maps,[11] several improvements have been proposed including local scaling,[145] variable bandwidth kernels[146] and target measure maps (TMDmap).[147] The latter scheme extends diffusion maps on point-clouds obtained from a surrogate distribution, ideally one that is easier to sample from. Based on the idea of importance sampling, it can be used on biased trajectories, and improves the accuracy and application of diffusion maps in high dimensions.[147]

Several algorithms have used diffusion maps to learn the CVs adaptively and thus enhance the dynamics in the learned slowest dynamics.[13,105,113,114] These methods are based on iterative procedures whereby diffusion maps are employed as a tool to gradually uncover the intrinsic geometry of the local states and drive the sampling toward unexplored domains of the state space, either through sequential restarting[114] or pushing[105] the trajectory from the border of the point-cloud in the direction given by the reduced coordinates. All these methods try to gather local information about the metastable states to drive global sam-

pling. In,[73] the authors focused on the construction of diffusion maps within a metastable state by formalizing the concept of a local equilibrium based on the *quasi-stationary distribution*.[148] This local equilibrium guarantees the convergence of the diffusion map within the metastable state. Moreover, the work provides the analytic form of the operator obtained when metastable trajectories are used within diffusion maps.

Finally, since the collective variables provided by diffusion maps are only defined on the sampled point cloud, one must apply extrapolation approaches. These might be very noisy and, more importantly, lose their meaning outside the convex hull of the point cloud. As a remedy, diffusion maps could be used as a tool to select collective variables from a database of physical reaction coordinates, similarly to,[17] providing more physical insight into the abstract collective variables. This approach would allow to evaluate the CV outside the point cloud and provide more physical meaning into the abstract collective variables.

The local-global perspective has motivated a method allowing on-the-fly identification of metastable states as an ensemble of configurations along a trajectory, for which the diffusion map spectrum converges. Secondly, an enhanced sampling algorithm based on QSD and diffusion maps has been proposed. For the latter, the main idea is a sample from the QSD allowing to build high-quality local CVs (within the metastable state) by considering the most correlated physical CVs to the diffusion coordinates. Once the best local CVs have been identified, one can use existing methods as metadynamics to enhance the sampling, effectively driving the dynamics to exit the metastable state. The authors in[73] demonstrate this idea on a toy-model example showing improved sampling over the standard approach.

Diffusion maps can also be used to a compute the committor function,[149] which provides dynamical information about the connection between two metastable states and can be used as a reaction coordinate. Markov state models (MSM) can in principle be used to compute committor probabilities,[150] but high dimensionality makes grid-based methods intractable. Similar work in this direction was done by.[149,151,152] Diffusion-maps, especially the TMDmap,[147] can be used for committor computations in high dimensions. The low

computational complexity aids in the analysis of molecular trajectories and helps to unravel the dynamical behaviour at various temperatures.

As a future work, the quality of the diffusion map approximation could be improved by introducing more sophisticated kernels or point-cloud approximations similarly to.[149] Also, diffusion maps could be extended to the approximation of generators of the underdamped Langevin dynamics.

3.4 Extracting dynamical information from trajectory data

Once good CVs or metastable states have been identified, these can be used to extract dynamical information. Let us describe in this section the approach followed by Thiede *et al.*,[151] which is based on a Galerkin projection of the infinitesimal generator.

The approach in[151] builds on the MSM and related frameworks.[116,118,129,153–158] Dynamical statistics of interest are cast as solutions to equations involving the generator, i.e., the operator that describes the evolution of functions of the dynamics over infinitesimal times. Although the full generator cannot be determined in general, the equations can be solved by a Galerkin approximation. In this approximation, the dynamical statistic of interest is expanded in terms of a basis, and its generator equation is reduced to a linear form. The contributing matrix elements (inner products of basis elements and the generator) can be estimated from short MD trajectories. A key challenge is to generate basis sets consistent with the boundary conditions. Thiede *et al.*[151] considered two basis sets: indicator functions that reprise MSMs and diffusion maps.[11] The latter showed promise for capturing smoothly varying dynamical statistics, such as committors and mean first-passage times with fewer basis functions, but the efficiency of a given basis is likely to be problem specific. Because the dynamical Galerkin approximation framework generalizes the notion of transition between states, the sampled configurations can be replaced by short trajectory segments. This allows treating memory that arises from incomplete description of the system by delay embedding.[159,160] This is an appealing alternative to extending the lag time in an MSM because

it does not sacrifice time resolution. Going forward, it will be interesting to investigate whether variational methods akin to those for elucidating time scales[116,134] can be developed to permit representation of the dynamical statistics in terms of nonlinear functions.

3.5 Tackling both Markovian and non-Markovian cases: Free energy, friction and mass profiles extracted from short MD trajectories using Langevin models

In principle, the high-dimensional dynamics of a system composed by many atoms, when projected onto one (or a few) CV, can be modeled by a generalized Langevin equation.[161,162] Such stochastic differential equations contain several ingredients: a mass, a drift term corresponding to the mean force (gradient of the free energy landscape), a friction and a noise. Projecting on a low-dimensional space yields, in general, non-Markovian dynamics, except in the presence of time scale separation between CVs and bath coordinates and at coarse time resolution.[161]

Clearly, the construction of optimal Langevin models along meaningful reaction coordinates is appealing from several viewpoints.[163] On one side, the complex many-body dynamics is approximated by an equation that preserves physical intuition and is cheap to integrate. On the other side, exact kinetic rates - free from transition state theory approximations - between metastable states can be accessed more easily, by exploiting brute-force Langevin simulations or more elaborate methods.[164] Generalized Langevin models include by construction memory effects in the selected physically-measurable variables, effects that are missing in standard Markov state models. Notice, however, that there are approaches to include memory effects also in discrete state models.[165]

For all these reasons, several algorithms have been developed to recast MD data into low-dimensional Langevin models.[166–177] Usually, with these techniques, the terms of the Langevin equation are estimated employing very long equilibrium MD trajectories that er-

godically sample the whole relevant free energy landscape. Of course such data are seldom available in complex applications featuring rare events, strongly limiting the scope to the case of barriers smaller than a few k_BT. Tackling the more general case of limited sampling and non-equilibrium MD trajectories is much more involved.[178]

A possible and simple solution to this challenge - especially in the context of rare events - has been proposed in Ref. 179: the parameters of a generalized Langevin equation are optimized by minimizing the error between MD and Langevin probability distributions $P(x, \dot{x}, t)$ along the reaction coordinate x. Such out-of-equilibrium distributions are estimated from a set of short unbiased trajectories initiated close to a barrier top (with random thermal velocities) and allowed to relax into the adjacent free energy minima, in the spirit of committor analysis (a preliminary exploration of putative transition state structures can be nowadays performed at a moderate cost using, e.g., the prejudice-free techniques of Ref. 180–182).

Employing both benchmark models and solvated proline dipeptide as a test case, numerical evidence indicates that $\sim$100 short trajectories (of few picoseconds in the typical case of a small solute in water) encode all the information needed to reconstruct free energy, friction, and mass profiles.[179] This approach, suitable also for high barriers of tens of k_BT and non-Markovian dynamics, provides the thermodynamics and kinetics of activated processes in a conceptually direct way, employing only standard unbiased MD, at a competitive cost with respect to existing enhanced sampling methods. Furthermore, the systematic construction of Langevin models for different choices of CVs starting from the same initial data could help in reaction coordinate optimization.

4 Application of machine learning techniques in biological systems and drug discovery

Two of biology's biggest challenges are the prediction of protein structure based on its amino acid sequence, i.e., protein folding, as well as the dynamical conformational changes of the

three-dimensional structure of proteins, i.e., protein dynamics. Beyond the actual problem of protein folding, which was recently set at a different basis after the breakthrough from AlphaFold and the impressive one million time faster Artificial Intelligence (AI) solution by AlQuraishi,[183] the prediction of protein dynamics and mechanism of action is possible through the use of MD simulations.

Recent advances in computer hardware and algorithms have led to simulations of protein dynamics of size and time lengths that are intrinsic to biological processes. Dynamics of protein plasticity and drug binding/unbinding mechanisms are a few of the key processes that we would ideally like to capture through these large scale simulations. However, the analysis and interpretation of the large amount of data that are produced by these simulations is complex and should be carefully considered.[184]

As discussed in Section 3.2, despite the ever-growing time and length scales of simulations, unbiased MD is not able to explore the whole kinetic landscape of complex systems and carefully chosen, meaningful CVs can be used to represent the free energy surface of these systems in order to reveal the regions of low energy, i.e., stable and metastable states, as well as the barriers, i.e., transition states, between these regions.[168,174,185] ML approaches have recently started being used for the discovery of meaningful CVs,[14,15,134,186,187] while iterative schemes where CVs are being updated based on new simulation data provide promising results for challenging systems.[186,188,189]

In this section, we first present an example of dimensionality reduction for building a Markov State Model for the study of lysine methyltransferase SETD8 (see Section 4.1). We next present some biological examples were adaptive MD/ML techniques can help gain access to non-crystallographic conformational states of disease-related proteins for drug discovery purposes (see Section 4.2). In Section 4.2.1, we discuss the possibility of conformational-specific targeting of proteins using their metastable states as target conformations, while in Section 4.2.2 we give some examples were ML techniques applied in MD simulations can provide information about potential allosteric binding sites or protein activation mechanisms

upon ligand binding.

4.1 Selection of efficient collective variables for MSMs: the example of SETD8

Conformational changes in proteins span from thermal fluctuations of side chains and motions of active loops to major rearrangement of sub-domains, including unfolding and refolding processes.[190] The ability to unveil the mechanisms underlying protein function requires quantifying the importance of these motions for the process of interest or, in other words, obtaining a representative ensemble of conformations.

Besides the relevance for devising enhanced sampling strategies, the discovery of CVs is decisive when analyzing simulation data sets by using, for instance, Markov State Models. In this context, the conformational study of the protein methyltransferase SETD8, an epigenetic enzyme essential in the regulation of the cell cycle, was discussed in.[188]

SETD8 is characterized by a dynamically rich behavior, which has proven to be essential in enzymatic catalysis.[191] In[188] the authors combined experiments and simulation in an attempt to span the up-to-that-time unexplored configurational space of SETD8. Several new X-ray structures were obtained by trapping conformations with small-molecule ligands.[192] These, in turn, were used to build hypothetical structures by manually combining fragments observed in experiments.

The set of initial configurations was used to seed independent MD simulations in explicit solvent, resulting in an extensive simulation database. The search of reaction coordinates was done in different spaces of residue-residue distances, logistic distances, and backbone dihedrals. These CVs, usually referred to as "features" in the MSMs literature, are arbitrary choices, that have been traditionally based on human intuition and heuristics.[193] This is arguably the "achilles heel" of MSMs and has prompted the development of ML approaches to bypass human intervention.[16,134]

Given that MSMs seek to approximate the slowest kinetic processes, it is essential to

build such a model on top of data reflecting time scale separation.[187] To this end, a common approach is to apply dimensionality reduction techniques, such as tICA or PCA, to the preselected set of features.[120,125,193,194] In an MSM analysis of ultra-long simulations of 12 small proteins, Husic et al.[193] showed tICA to consistently outperform PCA in producing higher scoring (i.e., slower) MSMs that better approximate the true slow timescales of the system dynamics. This is because PCA emphasizes large (high variance) motions, which can be fast, while de-emphasizing, i.e., grouping together, rare motions. Still, a crucial limitation of both methods is that, by construction, they yield a linear combination of features, which can fail in capturing inherently nonlinear processes. This has prompted the development of nonlinear approaches, including variations of tICA and autoencoders.[136,195] To avoid the issues discussed so far, others have opted to skip the dimensionality reduction stage by using structural properties, such as RMSD[196,197] and contact maps.[195] The stage regarding data representation ends with clustering the conformational snapshots into discrete states using unsupervised ML protocols, such as the k-centers and k-means methods.[198]

Given the multiple subjective decisions involved in selecting features and algorithms to represent the database, MSMs building must be allied with validation strategies. In this context, Husic *et al.*[193] emphasize the importance of using a kinetically-motivated dimensionality reduction and cross-validation strategies to avoid over fitting. The study of SETD8[188] uses both structural and kinetic criteria, and 50:50 shuffle-split cross-validation scheme with random divisions of the data into training and test sets (see Figure 3). As a result of such an extensive validation, the specific study successfully quantified an ensemble of kinetically relevant macrostates which, in addition, were validated with experiments.

4.2 Machine learning-driven MD simulations in drug discovery

The discovery of a new drug is a long, multi-step and expensive process. Any tool that can speed up any of the steps involved would have big implications down the entire drug discovery chain. Artificial intelligence is expected to significantly shape the future of many aspects of

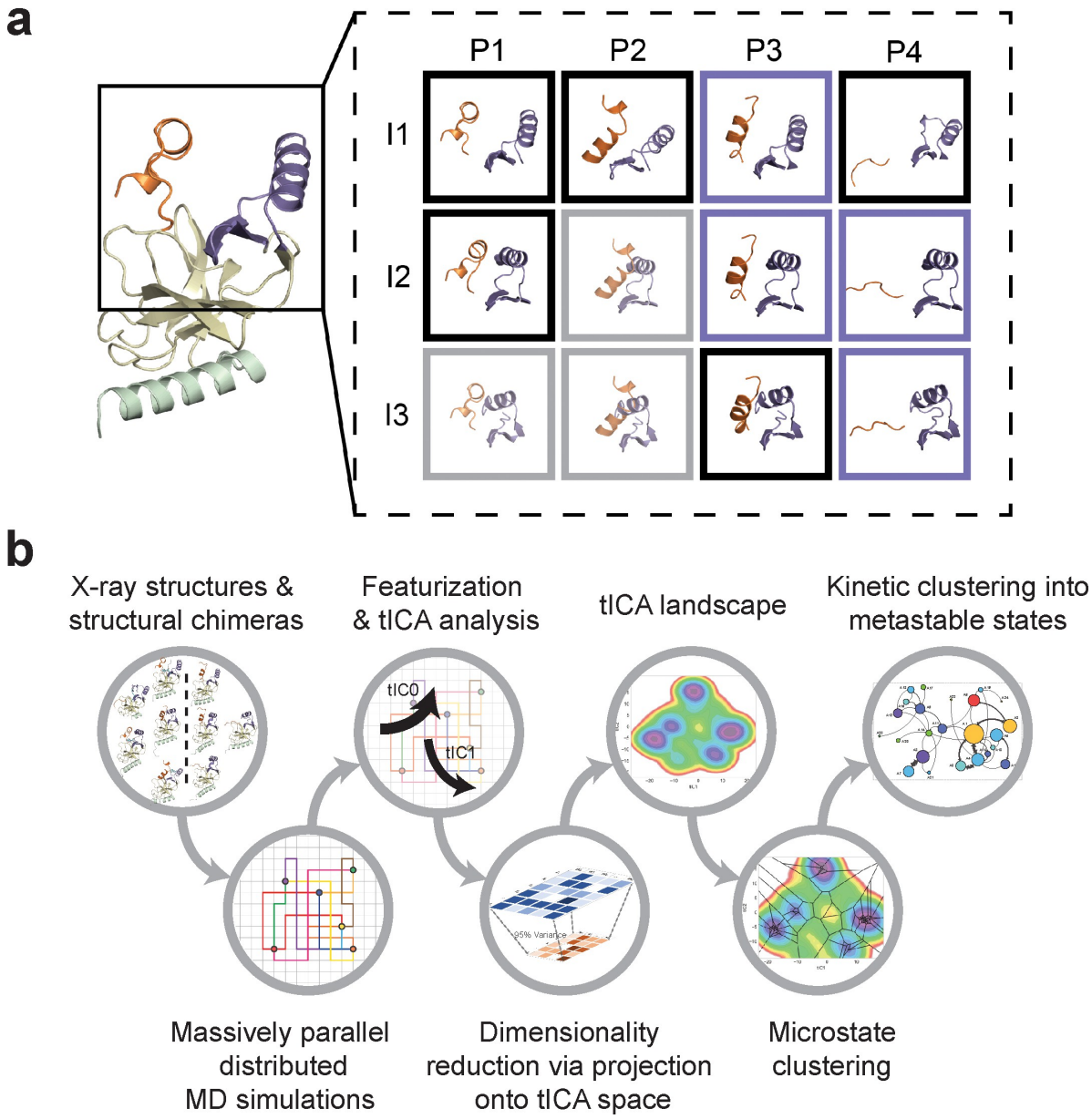


Figure 3: Construction of conformational landscapes of apo- and SAM-bound SETD8 through diversely seeded, parallel molecular dynamics simulations and Markov state models.(a) Combinatorial construction of structural chimeras using crystallographically-derived conformations. (b) Workflow for dynamic conformational landscapes construction using MSM. For more information we refer the reader to the original publication 188. (Image source: Ref. 188. Use permitted under the Creative Commons Attribution License CC BY 4.0., https://creativecommons.org/licenses/by/4.0/).

drug discovery during the forthcoming decades. It is already used to design evidence-based treatment plans for cancer patients, instantly analyze results from medical tests to escalate to the appropriate specialist immediately, and most recently to conduct scientific research

for early-stage drug discovery.

Proteins, the most common drug targets, are dynamic molecular machineries whose function is intimately linked to their conformations. Destabilization of the subtle equilibrium of protein conformations can lead to severe pathologies, like in the well-known cases of KRAS G12X oncogenic mutations and prion disease. In this context, knowledge of the conformational landscape of targeted proteins would provide an outstanding advantage for the design of novel and original compounds stabilizing specific conformations of the protein.[199]

Experimentally, the protein conformational space is often limited to few conformations that have been prone to crystallize. The use of GPUs and massive computational resources has enabled for the *in silico* alternative, MD simulations, to gain an important place in the first steps of drug discovery. Nevertheless, MD is limited to a few hundreds of microseconds of simulation, which limits the conformational space exploration.

New molecular modeling approaches combining MD simulations and ML techniques can help gain access to these non-crystallographic conformational states of a target protein. This knowledge would allow focusing on specific conformations of the protein in order to alter or restore its function. ML techniques can enable us to identify patterns in simulation data, build models that explain the different conformational states of a target and predict potential target-specific solutions for their druggability.[13,15,186,187,189,200–203]

As discussed in Section 3.1, good CVs can guide enhanced sampling MD simulations in order to gain insights into long timescale dynamics of biomolecular systems. The difficulty of the identification of such CVs and in most cases the complexity of their definition has limited the number of available software for this purpose. PLUMED is an open-source, community-developed library that has been widely used in enhanced-sampling simulations of complex biological systems in combination with many MD engines, e.g., Amber, GROMACS, NAMD, and OpenMM.[204–208] Most importantly, PLUMED can be interfaced with the host code using an API, accessible from multiple languages, including C++ and Python). This last functionality is important for adaptive protocols used for the identification of optimal

CVs using iterative learning algorithms based on well developed ML libraries like Keras,[209] TensorFlow,[210] PyTorch[211] and Fastai.[212] The MSM Builder package provides the user with software tools for predictive modeling of long timescale dynamics of biomolecular systems using statistical modeling to analyze physical simulations.[213] Other tools that can be employed in MD/ML studies include among others MDTraj,[214] ColVar module for VMD,[200] OpenPathSampling.[215]

In all the above-mentioned methods, an identified set of meaningful CVs is needed in order to perform enhanced sampling simulations. In their recent study, Noé et al. introduce a novel approach where Boltzmann generators are used to learn to generate unbiased equilibrium samples from different metastable states.[216] This approach opens new directions in the exploration of bio-molecular simulations as the latent spaces learned by Boltzmann generators can be combined with existing sampling methods in order to address rare event-sampling problems in complex systems.

4.2.1 Conformational-specific targeting of proteins using cryptic binding sites

Drugs are traditionally designed to bind to the primary active site of their biological targets in order to induce a therapeutic effect. However, the high similarity between the orthosteric pockets among most of the protein families, leads in several cases to adverse effects. A new emerging direction in drug discovery is the use of alternative, transient, non-orthosteric binding sites that are not apparent in the protein's known crystallographic conformations and where small molecules can bind and modulate the biological target's function.

By binding to non-orthosteric sites of proteins, allosteric inhibitors can also exhibit a better selectivity vs proteins from the same family, as illustrated by SAR156497, a highly selective inhibitor of Aurora kinases.[217] Well known drugs on the market work through this kind of mechanism of action (e.g., Lapatinib or Imatinib), but this mechanism was described *a posteriori*. Moreover, there are approved allosteric modulator drugs such as Cinacalcet for the treatment of hyperparathyroidism and Maraviroc for the treatment of AIDS, as well

as many candidates at different stages of development.[218,219] Another aspect in targeting non-orthosteric pockets in drug discovery relies on the fact that allosteric inhibitors will not compete with endogenous ligands for binding, which can be critical when such endogenous ligands have very strong affinity for their protein.

One of the successful efforts in this direction is the example of PI3Kα, where a novel non-orthosteric pocket was identified using molecular dynamics (MD) simulations.[220,221] In,[220] the authors used Functional Mode Analysis[222] and identified two dominant motions of PI3Kα that influence both the active and allosteric pockets and are distinct between the wild-type protein and its oncogenic counterpart. Current work aims at extending this approach to other protein targets, where neural networks are employed in order to establish the link between oncogenic mutations and the protein's mode of action, with an ultimate goal to identify druggable mutant-specific conformations.

Beyond single protein conformations, multimeric protein assembly also appears as a challenging area where ML could play a role in drug discovery. The recent example on TNFα for instance shows the importance of how subtle changes in protein conformation can translate into a distorded trimeric assembly of TNFα, impacting downstream signaling of TNFR1. Small compounds stabilizing this asymmetrical TNFα trimer can then be designed to treat or prevent TNFα-related diseases.[223]

4.2.2 Compound-specific effect of binding

Another promising direction in the drug discovery process is the compound-specific effect of protein binding.[224,225] For example, a small organic compound can be used to boost the enzymatic activity of a protein enzyme or evaluate allosteric binders by the stabilization of its active conformation. In finding allosteric binding sites, ML algorithms such as k-means and Markov Models can significantly help in reducing the dimensions of drug binding events. The connections between statistical mechanics principles, such as Boltzmann Machines, and the discovery of the binding sites in proteins can be insightful. As an example, one can run

thousands of small trajectories of drug binding and unbinding events and learn the reaction coordinates using tICA (time-independent Component Analysis) in order to find the possible allosteric binding sites.[224] These trajectories can be generated using different initial seeds (both different locations and orientations) and may range from 50 ns to 500 ns.

In the activation pathway of many proteins such as G Protein Coupled Receptors (GPCRs), the conformational changes are subtle and are limited to the sequential motion of residue switches triggering a signal from ligand to intracellular motifs. Finding these intricate motions in high dimensional space requires ML techniques to reduce the system's dimensions.[225] Among these methods, variational autoencoders (VAE) and tICA (sparse or kernel) can be used to achieve learning and finding the reaction coordinates for such complex proteins.

5 Concluding remarks and perspective

Let us conclude this review by presenting some global perspectives on the interactions between machine learning approaches and molecular simulation, which are common to all the situations we discussed – from devising numerical potentials based on ab-initio reference data to the identification of collective variables in actual simulation of biological proteins.

First, we have seen that the aims of the coarse-graining procedures may be very different in nature. From the material presented in this review, one can identify three major purposes: (1) *a modeling objective*: using machine learning techniques to improve models, for instance by better representing force fields and potential energy surfaces; (2) *a numerical objective*: improving the efficiency of numerical methods, for instance by devising good collective variables to be used in conjunction with enhanced sampling techniques, such as free energy biased sampling techniques; (3) *a data analysis objective*: providing an efficient post-processing tool, as for instance a Markov state model to interpret the raw simulation data from molecular dynamics and identify states of interest.

Concerning the choice of the learning methods, some common trends are shared by all

methods, namely ensuring that one has access to a sufficiently rich database (sufficient variability of configurations for force fields, long reactive trajectories to identify CVs) and representing correctly the data (starting possibly with some putative CVs/descriptors, and then using some regression from there to sparsify/optimally combine these initial guesses). The precise choice of the learning method and the reduced model to work with, however, depend very much on the goal and priority of the user, and the system under consideration. The priority can be *the accuracy* (being as precise and as close as possible to some reference model, e.g., all-atom results when coarse-graining, or reproducing DFT energies when constructing numerical potentials), *the transferability* (learning how to coarse-grain small systems and extending the method to larger ones, learning energies at a given temperature and using the potential at another one) or the CPU/GPU *computational cost*.

In this context, the method to be used for dimensionality reduction with minimal information loss greatly depends on the objective of each study. Linear methods generally demand less computational power and their accuracy can be more easily assessed than for nonlinear methods, thanks to built-in error estimators. Their results also admit a statistical interpretation in many situations. Nonlinear methods on the other hand usually have a better approximation power and can tackle more complicated problems. In some cases, they are effective only for specific data-sets and fail to generalize over real world data, i.e., they may be system/problem-dependent, even at the heavy computational cost needed to accommodate non-linearity. There are, however, cases where it is useful to rely on nonlinear functions to map a high-dimensional space into a meaningful reduced dimensional space such as when studying complex protein dynamics where the leading structural or kinetic collective variables are typically complex nonlinear functions of the atomic coordinates.

When using black box learning techniques, based for example on neural networks, a problem which is often raised is the *interpretability* of the result. This is discussed for example in[80] which attempts to reconcile machine learning models (specifically a neural network approach to optimal reaction coordinates) with physical insight by means of symbolic

regression techniques, also known as genetic programming. Such techniques appear very promising for the future, being able to distill fundamental natural laws from numerical data.[226]

Another important element is the *reproducibility* of the results: one should favor approaches which are easy enough to cross-check and to repeat on various architectures. This also requires the researchers to ensure that the coarse-graining technique they propose yield robust results. For example, the results should not depend on the initial weights in a neural network, or on the sampled point used as inputs. Finally, this includes considering well established databases, or making databases available to other users/developers; and also relying on standard and well maintained packages when using external libraries.

One idea which would help setting up common benchmarks and/or agreeing on common aims/priorities would be to organize some competition or prediction contest, which should ideally be simple enough so that even small groups can participate since this requires agreeing on common goals. Setting up the rules of such a competition would already be quite an achievement. Another important idea would be to emphasize transferability in all approaches, and more systematically work with some databases of some sort and then test on different databases.

Finally, the authors envisage that ML approaches similar to the ones presented herein could be extended to non-classical MD quantum dynamics simulations. The combination of affordable ab-initio-quality ML models and accelerated quantum dynamics techniques is making once-prohibitive simulations feasible, as demonstrated by recent work using ML to probe quantum statistics[227,228] and dynamics.[229–231] We expect that the combination with more sophisticated sampling techniques shall extend even further the range of biological and materials systems that are amenable to molecular dynamics simulations.

Acknowledgement

This review paper was written following a CECAM (Centre Européen de Calcul Atomique et Moléculaire) discussion meeting, hosted at the Sanofi Campus of Gentilly. The authors thank the CECAM as well as Sanofi for making this event possible. Moreover, the PG, GS and TL thank Dr. Marc Bianciotto for proof reading and feedback.

References

(1) Zhang, Y.-Y.; Niu, H.; Piccini, G.; Mendels, D.; Parrinello, M. Improving collective variables: The case of crystallization. *J. Chem. Phys.* **2019**, *150*, 094509.

(2) Behler, J. Atom-centered symmetry functions for constructing high-dimensional neural network potentials. *J. Chem. Phys.* **2011**, *134*, 074106.

(3) Bartók, A. P.; Kondor, R.; Csányi, G. On representing chemical environments. *Phys. Rev. B* **2013**, *87*, 184115.

(4) Wales, D. J. Perspective: Insight into reaction coordinates and dynamics from the potential energy landscape. *J. Chem. Phys.* **2015**, *142*, 130901.

(5) Peters, B. Reaction Coordinates and Mechanistic Hypothesis Tests. *Annu. Rev. Phys. Chem.* **2016**, *67*, 669–690.

(6) McGibbon, R. T.; Husic, B. E.; Pande, V. S. Identification of simple reaction coordinates from complex dynamics. *J. Chem. Phys.* **2016**, *146*, 044109.

(7) Chmiela, S.; Sauceda, H. E.; Müller, K.-R.; Tkatchenko, A. Towards exact molecular dynamics simulations with machine-learned force fields. *Nat. Commun.* **2018**, *9*, 3887.

(8) Wang, J.; Olsson, S.; Wehmeyer, C.; Pérez, A.; Charron, N. E.; de Fabritiis, G.; Noé, F.; Clementi, C. Machine Learning of Coarse-Grained Molecular Dynamics Force Fields. *ACS Cent. Sci.* **2019**, *5*, 755–767.

(9) Häse, F.; Fernández Galván, I.; Aspuru-Guzik, A.; Lindh, R.; Vacher, M. How machine learning can assist the interpretation of ab initio molecular dynamics simulations and conceptual understanding of chemistry. *Chem. Sci.* **2019**, *10*, 2298–2307.

(10) Jolliffe, I. *Principal Component Analysis*; Wiley Online Library, 2002.

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

(12) Coifman, R. R.; Kevrekidis, I. G.; Lafon, S.; Maggioni, M.; Nadler, B. Diffusion maps, reduction coordinates, and low dimensional representation of stochastic systems. *Multiscale Model. Simul.* **2008**, *7*, 842–864.

(13) 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.

(14) 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.

(15) Ribeiro, J. M. L.; Bravo, P.; Wang, Y.; Tiwary, P. Reweighted autoencoded variational Bayes for enhanced sampling (RAVE). *J. Chem. Phys.* **2018**, *149*, 072301.

(16) Wehmeyer, C.; Noé, F. Time-lagged autoencoders: Deep learning of slow collective variables for molecular kinetics. *J. Chem. Phys.* **2018**, *148*, 241703.

(17) Ma, A.; Dinner, A. R. Automatic method for identifying reaction coordinates in complex systems. *J. Phys. Chem. B* **2005**, *109*, 6769–6779.

(18) Shapeev, A. V. Moment Tensor Potentials: A Class of Systematically Improvable Interatomic Potentials. *Multiscale Model. Simul.* **2016**, *14*, 1153–1173.

(19) Chen, H.; Lu, J.; Ortner, C. Thermodynamic limit of crystal defects with finite temperature tight binding. *Arch. Ration. Mech. Anal.* **2018**, *230*, 701–733.

(20) Lunghi, A.; Sanvito, S. A unified picture of the covalent bond within quantum-accurate force fields: From organic molecules to metallic complexes' reactivity. *Sci. Adv.* **2019**, *5*, eaaw2210.

(21) Artrith, N.; Behler, J. High-dimensional neural network potentials for metal surfaces: A prototype study for copper. *Phys. Rev. B* **2012**, *85*, 045439.

(22) Podryabinkin, E. V.; Tikhonov, E. V.; Shapeev, A. V.; Oganov, A. R. Accelerating crystal structure prediction by machine-learning interatomic potentials with active learning. *Phys. Rev. B* **2019**, *99*, 064114.

(23) Gubaev, K.; Podryabinkin, E. V.; Hart, G. L.; Shapeev, A. V. Accelerating high-throughput searches for new alloys with active learning of interatomic potentials. *Comput. Mater. Sci.* **2019**, *156*, 148–156.

(24) Huan, T. D.; Batra, R.; Chapman, J.; Kim, C.; Chandrasekaran, A.; Ramprasad, R. Iterative-Learning Strategy for the Development of Application-Specific Atomistic Force Fields. *J. Phys. Chem. C* **2019**, *123*, 20715–20722.

(25) Jinnouchi, R.; Karsai, F.; Kresse, G. On-the-fly machine learning force field generation: Application to melting points. *Phys. Rev. B* **2019**, *100*, 014105.

(26) Deringer, V. L.; Proserpio, D. M.; Csányi, G.; Pickard, C. J. Data-driven learning and prediction of inorganic crystal structures. *Faraday Discuss.* **2018**, *211*, 45–59.

(27) Eickenberg, M.; Exarchakis, G.; Hirn, M.; Mallat, S.; Thiry, L. Solid harmonic wavelet scattering for predictions of molecule properties. *J. Chem. Phys.* **2018**, *148*, 241732.

(28) Ferré, G.; Haut, T.; Barros, K. Learning molecular energies using localized graph kernels. *J. Chem. Phys.* **2017**, *146*, 114107.

(29) Weininger, D. SMILES, a chemical language and information system. 1. Introduction to methodology and encoding rules. *J. Chem. Inf. Model.* **1988**, *28*, 31–36.

(30) Steinhardt, P. J.; Nelson, D. R.; Ronchetti, M. Bond-orientational order in liquids and glasses. *Phys. Rev. B* **1983**, *28*, 784–805.

(31) Pietrucci, F.; Laio, A. A Collective Variable for the Efficient Exploration of Protein Beta-Sheet Structures: Application to SH3 and GB1. *J. Chem. Theory Comput.* **2009**, *5*, 2197–2201.

(32) 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.

(33) Behler, J.; Parrinello, M. Generalized neural-network representation of high-dimensional potential-energy surfaces. *Phys. Rev. Lett.* **2007**, *98*, 146401.

(34) Faber, F. A.; Christensen, A. S.; Huang, B.; von Lilienfeld, O. A. Alchemical and structural distribution based representation for universal quantum machine learning. *J. Chem. Phys.* **2018**, *148*, 241717.

(35) Willatt, M. J.; Musil, F.; Ceriotti, M. Atom-density representations for machine learning. *J. Chem. Phys.* **2019**, *150*, 154110.

(36) Bartók, A. P.; De, S.; Poelking, C.; Bernstein, N.; Kermode, J. R.; Csányi, G.; Ceriotti, M. Machine learning unifies the modeling of materials and molecules. *Sci. Adv.* **2017**, *3*, e1701816.

(37) Zuo, Y.; Chen, C.; Li, X.; Deng, Z.; Chen, Y.; Behler, J.; Csányi, G.; Shapeev, A. V.; Thompson, A. P.; Wood, M. A.; Ong, S. P. Performance and Cost Assessment of Machine Learning Interatomic Potentials. *J. Phys. Chem. A* **2020**, *124*, 731–745.

(38) Behler, J. Perspective: Machine Learning Potentials for Atomistic Simulations. *J. Chem. Phys.* **2016**, *145*, 170901.

(39) Braams, B. J.; Bowman, J. M. Permutationally invariant potential energy surfaces in high dimensionality. *Int. Rev. Phys. Chem.* **2009**, *28*, 577–606.

(40) Chmiela, S.; Tkatchenko, A.; Sauceda, H. E.; Poltavsky, I.; Schütt, K. T.; Müller, K.-R. Machine learning of accurate energy-conserving molecular force fields. *Sci. Adv.* **2017**, *3*, e1603015.

(41) 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.

(42) Schütt, K. T.; Sauceda, H. E.; Kindermans, P.-J.; Tkatchenko, A.; Müller, K.-R. SchNet - A deep learning architecture for molecules and materials. *J. Chem. Phys.* **2018**, *148*, 241722.

(43) Qu, C.; Bowman, J. M. A fragmented, permutationally invariant polynomial approach for potential energy surfaces of large molecules: Application to N-methyl acetamide. *J. Chem. Phys.* **2019**, *150*, 141101.

(44) Deringer, V. L.; Csányi, G. Machine learning based interatomic potential for amorphous carbon. *Phys. Rev. B* **2017**, *95*, 094203.

(45) Ambrosetti, A.; Ferri, N.; DiStasio, R.; Tkatchenko, A. Wavelike charge density fluctuations and van der Waals interactions at the nanoscale. *Science* **2016**, *351*, 1171–1176.

(46) Hermann, J.; DiStasio, R.; Tkatchenko, A. First-Principles Models for van der Waals Interactions in Molecules and Materials: Concepts, Theory, and Applications. *Chem. Rev.* **2017**, *117*, 4714–4758.

(47) Bereau, T.; DiStasio Jr, R.; Tkatchenko, A.; Von Lilienfeld, O. Non-covalent interactions across organic and biological subsets of chemical space: Physics-based potentials parametrized from machine learning. *J. Chem. Phys.* **2018**, *148*, 241706.

(48) Grisafi, A.; Ceriotti, M. Incorporating long-range physics in atomic-scale machine learning. *J. Chem. Phys.* **2019**, *151*, 204105.

(49) Glielmo, A.; Zeni, C.; Vita, A. D. Efficient nonparametricn-body force fields from machine learning. *Phys. Rev. B* **2018**, *97*.

(50) Veit, M.; Jain, S. K.; Bonakala, S.; Rudra, I.; Hohl, D.; Csányi, G. Equation of State of Fluid Methane from First Principles with Machine Learning Potentials. *J. Chem. Theory Comput.* **2019**, *15*, 2574–2586.

(51) Ziegler, J. F.; Biersack, J. P. In *Treatise on Heavy-Ion Science*; Bromley, D. A., Ed.; Springer US: Boston, MA, 1985; Vol. 6:Astrophysics, Chemistry, and Condensed Matter; pp 93–129.

(52) Lemke, T.; Peter, C. Neural Network Based Prediction of Conformational Free Energies - A New Route toward Coarse-Grained Simulation Models. *J. Chem. Theory Comput.* **2017**, *13*, 6213–6221.

(53) Hunkler, S.; Lemke, T.; Peter, C.; Kukharenko, O. Back-mapping based sampling: Coarse grained free energy landscapes as a guideline for atomistic exploration. *J. Chem. Phys.* **2019**, *151*, 154102.

(54) Peter, C.; Kremer, K. Multiscale simulation of soft matter systems - from the atomistic to the coarse-grained level and back. *Soft Matter* **2009**, *5*, 4357–4366.

(55) Rudzinski, J. F.; Noid, W. G. Coarse-graining entropy, forces, and structures. *J. Chem. Phys.* **2011**, *135*, 214101.

(56) Noid, W. G. Perspective: Coarse-grained models for biomolecular systems. *J. Chem. Phys.* **2013**, *139*, 090901.

(57) Potestio, R.; Peter, C.; Kremer, K. Computer Simulations of Soft Matter: Linking the Scales. *Entropy* **2014**, *16*, 4199–4245.

(58) Shell, M. S. Coarse-graining with the relative entropy. *Adv. Chem. Phys.* **2016**, 395–441.

(59) John, S. T.; Csányi, G. Many-Body Coarse-Grained Interactions Using Gaussian Approximation Potentials. *J. Phys. Chem. B* **2017**, *121*, 10934–10949.

(60) Zhang, L.; Han, J.; Wang, H.; Car, R.; E, W. DeePCG: Constructing coarse-grained models via deep neural networks. *J. Chem. Phys.* **2018**, *149*, 034101.

(61) Noid, W. G.; Chu, J.-W.; Ayton, G. S.; Krishna, V.; Izvekov, S.; Voth, G. A.; Das, A.; Andersen, H. C. The multiscale coarse-graining method. I. A rigorous bridge between atomistic and coarse-grained models. *J. Chem. Phys.* **2008**, *128*, 244114.

(62) Garrido, L.; Juste, A. On the determination of probability density functions by using Neural Networks. *Comput. Phys. Commun.* **1998**, *115*, 25–31.

(63) Ferguson, A. L.; Panagiotopoulos, A. Z.; Debenedetti, P. G.; Kevrekidis, I. G. Systematic determination of order parameters for chain dynamics using diffusion maps. *P. Natl. Acad. Sci. USA* **2010**, *107*, 13597–13602.

(64) Ferguson, A. L.; Panagiotopoulos, A. Z.; Kevrekidis, I. G.; Debenedetti, P. G. Nonlinear dimensionality reduction in molecular simulation: The diffusion map approach. *Chem. Phys. Lett.* **2011**, *509*, 1–11.

(65) Wang, J.; Ferguson, A. Nonlinear machine learning in simulations of soft and biological materials. *Mol. Simul.* **2018**, *44*, 1090–1107.

(66) Pietrucci, F. Strategies for the exploration of free energy landscapes: unity in diversity and challenges ahead. *Reviews in Physics* **2017**, *2*, 32–45.

(67) Ichiye, T.; Karplus, M. Collective motions in proteins: a covariance analysis of atomic fluctuations in molecular dynamics and normal mode simulations. *Proteins: Struct., Funct., Bioinf.* **1991**, *11*, 205–217.

(68) García, A. E. Large-amplitude nonlinear motions in proteins. *Phys. Rev. Lett.* **1992**, *68*, 2696.

(69) Amadei, A.; Linssen, A.; Berendsen, H. Essential dynamics of proteins. *Proteins* **1993**, *17*, 412–425.

(70) Ferguson, A. L. Machine learning and data science in soft materials engineering. *J. Phys. Condens. Matter* **2017**, *30*, 043002.

(71) Schölkopf, B. The Kernel Trick for Distances. Proceedings of the 13th International Conference on Neural Information Processing Systems. Cambridge, MA, USA, 2000; p 283–289.

(72) Sittel, F.; Stock, G. Perspective: Identification of collective variables and metastable states of protein dynamics. *J. Chem. Phys.* **2018**, *149*, 150901.

(73) Trstanova, Z.; Leimkuhler, B.; Lelièvre, T. Local and Global Perspectives on Diffusion Maps in the Analysis of Molecular Systems. *Proc. R. Soc. A* **2020**, *476*, 20190036.

(74) Jónsson, H.; Mills, G.; Jacobsen, K. In *Classical and Quantum Dynamics in Condensed Phase Simulations*; Berne, B., Ciccotti, G., Coker, D., Eds.; World Scientific, 1998; pp 385–404.

(75) E, W.; Weiqing, R.; Vanden-Eijnden, E. String method for the study of rare events. *Phys. Rev. B* **2002**, *66*, 52301.

(76) Weinan, E.; Vanden-Eijnden, E. Transition-path theory and path-finding algorithms for the study of rare events. *Annu. Rev. Phys. Chem.* **2010**, *61*, 391–420.

(77) Lelièvre, T.; Stoltz, G. Partial differential equations and stochastic methods in molecular dynamics. *Acta Numer.* **2016**, *25*, 681–880.

(78) Bolhuis, P. G.; Chandler, D.; Dellago, C.; Geissler, P. L. Transition path sampling: Throwing ropes over rough mountain passes, in the dark. *Annu. Rev. Phys. Chem.* **2002**, *53*, 291–318.

(79) Banushkina, P. V.; Krivov, S. V. Optimal reaction coordinates. *WIREs: Comput. Mol. Sci.* **2016**, *6*, 748–763.

(80) Jung, H.; Covino, R.; Hummer, G. Artificial Intelligence Assists Discovery of Reaction Coordinates and Mechanisms from Molecular Dynamics Simulations. 2019; arXiv:1901.04595 [physics.chem-ph], https://arxiv.org/abs/1901.04595 (accessed June 17, 2020).

(81) Loève, M. *Probability Theory: Foundations, Random Sequences*; Van Nostrand, 1955.

(82) Sirovich, L. Turbulence and the dynamics of coherent structures. I. Coherent structures. *Q. Appl. Math.* **1987**, *45*, 561–571.

(83) Sirovich, L. Turbulence and the dynamics of coherent structures. II. Symmetries and transformations. *Q. Appl. Math.* **1987**, *45*, 573–582.

(84) Park, H.; Cho, D. The use of the Karhunen-Loeve decomposition for the modeling of distributed parameter systems. *Chem. Eng. Sci.* **1996**, *51*, 81–98.

(85) Chatterjee, A. An introduction to the proper orthogonal decomposition. *Current Science* **2000**, *78*, 808–817.

(86) Liang, Y.; Lee, H.; Lim, S.; Lin, W.; Lee, K.; Wu, C. Proper orthogonal decomposition and its applications—Part I: Theory. *J. Sound Vib.* **2002**, *252*, 527–544.

(87) Schölkopf, B.; Smola, A.; Müller, K.-R. Kernel principal component analysis. Artificial Neural Networks — ICANN'97: 7th International Conference Lausanne, Switzerland, October 8–10, 1997 Proceeedings. Berlin Heidelberg, 1997; pp 583–588.

(88) Kramer, M. A. Nonlinear principal component analysis using autoassociative neural networks. *AIChE Journal* **1991**, *37*, 233–243.

(89) Nguyen, P. H. Complexity of free energy landscapes of peptides revealed by nonlinear principal component analysis. *Proteins: Structure, Function, and Bioinformatics* **2006**, *65*, 898–913.

(90) Scholz, M.; Fraunholz, M.; Selbig, J. In *Principal Manifolds for Data Visualization and Dimension Reduction*; Gorban, A. N., Kégl, B., Wunsch, D. C., Zinovyev, A. Y., Eds.; Springer: Berlin Heidelberg, 2008; pp 44–67.

(91) Comon, P. Independent component analysis, A new concept? *Signal Processing* **1994**, *36*, 287–314.

(92) Borg, I.; Groenen, P. J. *Modern Multidimensional Scaling: Theory and Applications*; Springer Science & Business Media, 2005.

(93) Ceriotti, M.; Tribello, G. A.; Parrinello, M. Simplifying the representation of complex free-energy landscapes using sketch-map. *Proc. Natl. Acad. Sci. USA* **2011**, *108*, 13023–13028.

(94) Roweis, S. T.; Saul, L. K. Nonlinear dimensionality reduction by locally linear embedding. *Science* **2000**, *290*, 2323–2326.

(95) Zhang, Z.; Wang, J. MLLE: Modified locally linear embedding using multiple weights. Advances in Neural Information Processing Systems 19: Proceedings of the 2006 Conference. Cambridge, 2007; pp 1593–1600.

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

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

(98) Silva, V. D.; Tenenbaum, J. B. In *Advances in Neural Information Processing Systems 15*; Thrun, S., Obermayer, K., Eds.; MIT Press: Cambridge, MA, 2002; pp 705–712.

(99) Wang, J. *Geometric Structure of High-Dimensional Data and Dimensionality Reduction*; Springer: Berlin Heidelberg, 2012; pp 221–234.

(100) Weinberger, K. Q.; Saul, L. K. Unsupervised learning of image manifolds by semidefinite programming. *Int. J. Comput. Vision* **2006**, *70*, 77–90.

(101) Belkin, M.; Niyogi, P. Laplacian eigenmaps for dimensionality reduction and data representation. *Neural Computation* **2003**, *15*, 1373–1396.

(102) Donoho, D. L.; Grimes, C. Hessian eigenmaps: Locally linear embedding techniques for high-dimensional data. *P. Natl. Acad. Sci. USA* **2003**, *100*, 5591–5596.

(103) 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. *P. Natl. Acad. Sci. USA* **2005**, *102*, 7426–7431.

(104) 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.

(105) 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. USA* **2017**, *114*, E5494–E5503.

(106) Ferguson, A. L.; Panagiotopoulos, A. Z.; Debenedetti, P. G.; Kevrekidis, I. G. Integrating diffusion maps with umbrella sampling: Application to alanine dipeptide. *J. Chem. Phys.* **2011**, *134*, 135103.

(107) Tribello, G. A.; Ceriotti, M.; Parrinello, M. Using sketch-map coordinates to analyze and bias molecular dynamics simulations. *Proc. Natl. Acad. Sci. USA* **2012**, *109*, 5196–5201.

(108) Abrams, C. F.; Vanden-Eijnden, E. On-the-fly free energy parameterization via temperature accelerated molecular dynamics. *Chem. Phys. Lett.* **2012**, *547*, 114–119.

(109) 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.

(110) Li, C.-G.; Guo, J.; Chen, G.; Nie, X.-F.; Yang, Z. A version of Isomap with explicit mapping. 2006 International Conference on Machine Learning and Cybernetics. 2006; pp 3201–3206.

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

(112) Branduardi, D.; Gervasio, F. L.; Parrinello, M. From A to B in free energy space. *J. Chem. Phys.* **2007**, *126*, 054103.

(113) Preto, J.; Clementi, C. Fast recovery of free energy landscapes via diffusion-map-directed molecular dynamics. *Phys. Chem. Chem. Phys.* **2014**, *16*, 19181–19191.

(114) 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.

(115) Wang, Y.; Ribeiro, J. M. L.; Tiwary, P. Past-future information bottleneck for sampling molecular reaction coordinate simultaneously with thermodynamics and kinetics. *Nat. Commun.* **2019**, *10*, 3573.

(116) Noé, F.; Nüske, F. A variational approach to modeling slow processes in stochastic dynamical systems. *Multiscale Model. Sim.* **2013**, *11*, 635–655.

(117) Mezić, I. Spectral properties of dynamical systems, model reduction and decompositions. *Nonlinear Dyn.* **2005**, *41*, 309–325.

(118) Williams, M. O.; Kevrekidis, I. G.; Rowley, C. W. A data–driven approximation of the Koopman operator: Extending dynamic mode decomposition. *J. Nonlinear Sci.* **2015**, *25*, 1307–1346.

(119) 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.

(120) 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.

(121) Nüske, F.; Keller, B. G.; Pérez-Hernández, G.; Mey, A. S. J. S.; Noé, F. Variational approach to molecular kinetics. *J. Chem. Theory Comput.* **2014**, *10*, 1739–1752.

(122) Noé, F.; Clementi, C. Kinetic distance and kinetic maps from molecular dynamics simulation. *J. Chem. Theory Comput* **2015**, *11*, 5002–5011.

(123) Noé, F.; Banisch, R.; Clementi, C. Commute Maps: Separating Slowly Mixing Molecular Configurations for Kinetic Modeling. *J. Chem. Theory Comput.* **2016**, *12*, 5620–5630.

(124) Pérez-Hernández, G.; Noé, F. Hierarchical time-lagged independent component analysis: Computing slow modes and reaction coordinates for large molecular systems. *J. Chem. Theory Comput.* **2016**, *12*, 6118–6129.

(125) 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.

(126) Klus, S.; Nüske, F.; Koltai, P.; Wu, H.; Kevrekidis, I.; Schütte, C.; Noé, F. Data-driven model reduction and transfer operator approximation. *J. Nonlinear Sci.* **2018**, *28*, 985–1010.

(127) Pande, V. S.; Beauchamp, K.; Bowman, G. R. Everything you wanted to know about Markov State Models but were afraid to ask. *Methods* **2010**, *52*, 99–105.

(128) Noé, F.; Rosta, E. Markov Models of Molecular Kinetics. *J. Chem. Phys.* **2019**, *151*, 190401.

(129) Schütte, C.; Fischer, A.; Huisinga, W.; Deuflhard, P. A direct approach to conformational dynamics based on hybrid Monte Carlo. *J. Comput. Phys.* **1999**, *151*, 146–168.

(130) Deuflhard, P.; Weber, M. Robust Perron cluster analysis in conformation dynamics. *Linear algebra and its applications* **2005**, *398*, 161–184.

(131) Röblitz, S.; Weber, M. Fuzzy spectral clustering by PCCA+: Application to Markov state models and data classification. *Adv. Data Anal. Classif.* **2013**, *7*, 147–179.

(132) Schwantes, C. R.; Pande, V. S. Modeling molecular kinetics with tICA and the kernel trick. *J. Chem. Theory Comput.* **2015**, *11*, 600–608.

(133) Andrew, G.; Arora, R.; Bilmes, J.; Livescu, K. Deep Canonical Correlation Analysis. Proceedings of the 30th International Conference on Machine Learning. Atlanta, Georgia, USA, 2013; pp 1247–1255.

(134) Mardt, A.; Pasquali, L.; Wu, H.; Noé, F. VAMPnets for deep learning of molecular kinetics. *Nat. Commun.* **2018**, *9*, 5.

(135) Chen, W.; Sidky, H.; Ferguson, A. L. Nonlinear Discovery of Slow Molecular Modes using State-Free Reversible VAMPnets. *J. Chem. Phys.* **2019**, *150*, 214114.

(136) Hernández, C. X.; Wayment-Steele, H. K.; Sultan, M. M.; Husic, B. E.; Pande, V. S. Variational encoding of complex dynamics. *Phys. Rev. E* **2018**, *97*, 062412.

(137) Wayment-Steele, H. K.; Pande, V. S. Note: Variational Encoding of Protein Dynamics Benefits from Maximizing Latent Autocorrelation. *J. Chem. Phys.* **2018**, *149*, 216101.

(138) Bonati, L.; Zhang, Y. Y.; Parrinello, M. Neural networks-based variationally enhanced sampling. *Proc. Natl. Acad. Sci. U.S.A.* **2019**, *116*, 17641–17647.

(139) Valsson, O.; Parrinello, M. Variational approach to enhanced sampling and free energy calculations. *Phys. Rev. Lett.* **2014**, *113*, 090601.

(140) Quer, J.; Donati, L.; Keller, B. G.; Weber, M. An automatic adaptive importance sampling algorithm for molecular dynamics in reaction coordinates. *SIAM J. Sci. Comput.* **2018**, *40*, A653–A670.

(141) Donati, L.; Keller, B. G. Girsanov reweighting for metadynamics simulations. *J. Chem. Phys.* **2018**, *149*, 072335.

(142) Donati, L.; Hartmann, C.; Keller, B. G. Girsanov reweighting for path ensembles and Markov state models. *J. Chem. Phys.* **2017**, *146*, 244112.

(143) Husic, B. E.; Noé, F. Deflation reveals dynamical structure in nondominant reaction coordinates. *J. Chem. Phys.* **2019**, *151*, 054103.

(144) Nadler, B.; Lafon, S.; Coifman, R.; Kevrekidis, I. G. Diffusion Maps - a Probabilistic Interpretation for Spectral Embedding and Clustering Algorithms. Principal Manifolds

for Data Visualization and Dimension Reduction. Berlin, Heidelberg, 2008; pp 238–260.

(145) Rohrdanz, M. A.; Zheng, W.; Maggioni, M.; Clementi, C. Determination of reaction coordinates via locally scaled diffusion map. *J. Chem. Phys.* **2011**, *134*, 124116.

(146) Berry, T.; Harlim, J. Variable bandwidth diffusion kernels. *Appl. Comput. Harmon. Anal.* **2016**, *40*, 68–96.

(147) Banisch, R.; Trstanova, Z.; Bittracher, A.; Klus, S.; Koltai, P. Diffusion maps tailored to arbitrary non-degenerate Itô processes. *Appl. Comput. Harmon. Anal.* **2018**, *48*, 242–265.

(148) Collet, P.; Martinez, S.; San Martin, J. *Quasi-Stationary Distributions: Markov Chains, Diffusions and Dynamical Systems*; Springer Science & Business Media, 2012.

(149) Lai, R.; Lu, J. Point Cloud Discretization of Fokker–Planck Operators for Committor Functions. *Multiscale Model. Simul.* **2018**, *16*, 710–726.

(150) Prinz, J.-H.; Held, M.; Smith, J. C.; Noé, F. Efficient computation, sensitivity, and error analysis of committor probabilities for complex dynamical processes. *Multiscale Model. Simul.* **2011**, *9*, 545–567.

(151) Thiede, E. H.; Giannakis, D.; Dinner, A. R.; Weare, J. Galerkin approximation of dynamical quantities using trajectory data. *J. Chem. Phys.* **2019**, *150*, 244111.

(152) Khoo, Y.; Lu, J.; Ying, L. Solving for high dimensional committor functions using artificial neural networks. *Research in the Mathematical Sciences* **2019**, *6*, 1.

(153) Molgedey, L.; Schuster, H. G. Separation of a mixture of independent signals using time delayed correlations. *Phys. Rev. Lett.* **1994**, *72*, 3634–3637.

(154) Takano, H.; Miyashita, S. Relaxation modes in random spin systems. *J. Phys. Soc. Jpn.* **1995**, *64*, 3688–3698.

(155) Hirao, H.; Koseki, S.; Takano, H. Molecular dynamics study of relaxation modes of a single polymer chain. *J. Phys. Soc. Jpn* **1997**, *66*, 3399–3405.

(156) 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.

(157) 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.

(158) Giannakis, D.; Slawinska, J.; Zhao, Z. Spatiotemporal Feature Extraction with Data-Driven Koopman Operators. Proceedings of the 1st International Workshop on Feature Extraction: Modern Questions and Challenges at NIPS 2015. Montreal, Canada, 2015; pp 103–115.

(159) Takens, F. *Detecting strange attractors in turbulence*; Lecture Notes in Mathematics; Springer, 1981; Vol. 898; pp 366–381.

(160) Aeyels, D. Generic observability of differentiable systems. *SIAM J. Control Optim.* **1981**, *19*, 595–603.

(161) Zwanzig, R. *Nonequilibrium Statistical Mechanics*; Oxford University Press, 2001.

(162) Łuczka, J. Non-Markovian stochastic processes: Colored noise. *Chaos* **2005**, *15*, 026107.

(163) Camilloni, C.; Pietrucci, F. Advanced simulation techniques for the thermodynamic and kinetic characterization of biological systems. *Adv. Phys.:X.* **2018**, *3*, 1477531.

(164) Hänggi, P.; Talkner, P.; Borkovec, M. Reaction-rate theory: Fifty years after Kramers. *Rev. Mod. Phys.* **1990**, *62*, 251–341.

(165) Perez, D.; Uberuaga, B. P.; Shim, Y.; Amar, J. G.; Voter, A. F. Accelerated molecular dynamics methods: introduction and recent developments. *Annu. Rep. Comput. Chem.* **2009**, *5*, 79–98.

(166) Straub, J. E.; Borkovec, M.; Berne, B. J. Calculation of dynamic friction on intramolecular degrees of freedom. *J. Phys. Chem.* **1987**, *91*, 4995–4998.

(167) Hummer, G.; Kevrekidis, I. G. Coarse molecular dynamics of a peptide fragment: Free energy, kinetics, and long-time dynamics computations. *J. Chem. Phys.* **2003**, *118*, 10762–10773.

(168) Lange, O. F.; Grubmüller, H. Collective Langevin dynamics of conformational motions in proteins. *J. Chem. Phys.* **2006**, *124*, 214903.

(169) 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.

(170) Horenko, I.; Hartmann, C.; Schütte, C.; Noé, F. Data-based parameter estimation of generalized multidimensional Langevin processes. *Phys. Rev. E* **2007**, *76*, 016706.

(171) Micheletti, C.; Bussi, G.; Laio, A. Optimal Langevin modeling of out-of-equilibrium molecular dynamics simulations. *J. Chem. Phys.* **2008**, *129*, 074105.

(172) Darve, E.; Solomon, J.; Kia, A. Computing generalized Langevin equations and generalized Fokker–Planck equations. *P. Natl. Acad. Sci. USA* **2009**, *106*, 10884–10889.

(173) Legoll, F.; Lelièvre, T. Effective dynamics using conditional expectations. *Nonlinearity* **2010**, *23*, 2131.

(174) Schaudinnus, N.; Bastian, B.; Hegger, R.; Stock, G. Multidimensional Langevin modeling of nonoverdamped dynamics. *Phys. Rev. Lett.* **2015**, *115*, 050602.

(175) Meloni, R.; Camilloni, C.; Tiana, G. Properties of low-dimensional collective variables in the molecular dynamics of biopolymers. *Phys. Rev. E* **2016**, *94*, 052406.

(176) Lesnicki, D.; Vuilleumier, R.; Carof, A.; Rotenberg, B. Molecular hydrodynamics from memory kernels. *Phys. Rev. Lett.* **2016**, *116*, 147804.

(177) Daldrop, J. O.; Kappler, J.; Brünig, F. N.; Netz, R. R. Butane dihedral angle dynamics in water is dominated by internal friction. *P. Natl. Acad. Sci. USA* **2018**, *115*, 5169–5174.

(178) Zhang, Q.; Brujić, J.; Vanden-Eijnden, E. Reconstructing free energy profiles from nonequilibrium relaxation trajectories. *J. Stat. Phys.* **2011**, *144*, 344–366.

(179) Pérez-Villa, A.; Pietrucci, F. Free energy, friction, and mass profiles from short molecular dynamics trajectories. 2018; arXiv:1810.00713 [cond-mat.stat-mech], https://arxiv.org/abs/1810.00713 (accessed June 17, 2020).

(180) Samanta, A.; Chen, M.; Yu, T.-Q.; Tuckerman, M.; E, W. Sampling saddle points on a free energy surface. *J. Chem. Phys.* **2014**, *140*, 164109.

(181) 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. *P. Natl. Acad. Sci. USA* **2015**, *112*, 15030–15035.

(182) Pipolo, S.; Salanne, M.; Ferlat, G.; Klotz, S.; Saitta, A. M.; Pietrucci, F. Navigating at will on the water phase diagram. *Phys. Rev. Lett.* **2017**, *119*, 245701.

(183) AlQuraishi, M. End-to-End Differentiable Learning of Protein Structure. *Cell Systems* **2019**, *8*, 292–301.e3.

(184) Shaw, D. E.; Maragakis, P.; Lindorff-Larsen, K.; Piana, S.; Dror, R. O.; Eastwood, M. P.; Bank, J. A.; Jumper, J. M.; Salmon, J. K.; Shan, Y.; Wriggers, W.

Atomic-Level Characterization of the Structural Dynamics of Proteins. *Science* **2010**, *330*, 341–346.

(185) Krivov, S. V.; Karplus, M. Diffusive reaction dynamics on invariant free energy profiles. *P. Natl. Acad. Sci. USA* **2008**, *105*, 13841–13846.

(186) 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.

(187) Brandt, S.; Sittel, F.; Ernst, M.; Stock, G. Machine Learning of Biomolecular Reaction Coordinates. *J. Phys. Chem. Lett.* **2018**, *9*, 2144–2150.

(188) Chen, S.; Wiewiora, R. P.; Meng, F.; Babault, N.; Ma, A.; Yu, W.; Qian, K.; Hu, H.; Zou, H.; Wang, J.; Fan, S.; Blum, G.; Pittella-Silva, F.; Beauchamp, K. A.; Tempel, W.; Jiang, H.; Chen, K.; Skene, R.; Zheng, Y. G.; Brown, P. J.; Jin, J.; Luo, C.; Chodera, J. D.; Luo, M. The Dynamic Conformational Landscapes of the Protein Methyltransferase SETD8. *eLife* **2019**, *8*, e45403.

(189) Trapl, D.; Horvacanin, I.; Mareska, V.; Ozcelik, F.; Unal, G.; Spiwok, V. Anncolvar: Approximation of Complex Collective Variables by Artificial Neural Networks for Analysis and Biasing of Molecular Simulations. *Front. Mol. Biosci.* **2019**, *6*, 25.

(190) Henzler-Wildman, K.; Kern, D. Dynamic personalities of proteins. *Nature* **2007**, *450*, 964–972.

(191) Schramm, V. L. Enzymatic Transition States, Transition-State Analogs, Dynamics, Thermodynamics, and Lifetimes. *Annu. Rev. Biochem.* **2011**, *80*, 703–732.

(192) Lee, G. M.; Craik, C. S. Trapping Moving Targets with Small Molecules. *Science* **2009**, *324*, 213–215.

(193) 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.

(194) Noé, F.; Clementi, C. Collective variables for the study of long-time kinetics from molecular trajectories: Theory and methods. *Curr. Opin. Struc. Biol.* **2017**, *43*, 141–147.

(195) Doerr, S.; Ariz-Extreme, I.; Harvey, M. J.; Fabritiis, G. D. Dimensionality reduction methods for molecular simulations. 2017; arXiv:1710.10629 [stat.ML], https://arxiv.org/abs/1710.10629 (accessed June 17, 2020).

(196) Kohlhoff, K.; 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*, 15–21.

(197) 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.

(198) Bowman, G. R. *An Introduction to Markov State Models and Their Application to Long Timescale Molecular Simulation*; Advances in Experimental Medicine and Biology; Springer Netherlands, 2014; pp 7–22.

(199) Wodak, S. J.; Paci, E.; Dokholyan, N. V.; Berezovsky, I. N.; Horovitz, A.; Li, J.; Hilser, V. J.; Bahar, I.; Karanicolas, J.; Stock, G.; Hamm, P.; Stote, R. H.; Eberhardt, J.; Chebaro, Y.; Dejaegere, A.; Cecchini, M.; Changeux, J.-P.; Bolhuis, P. G.; Vreede, J.; Faccioli, P.; Orioli, S.; Ravasio, R.; Yan, L.; Brito, C.; Wyart, M.; Gkeka, P.; Rivalta, I.; Palermo, G.; McCammon, J. A.; Panecka-Hofman, J.; Wade, R. C.; Pizio, A. D.; Niv, M. Y.; Nussinov, R.; Tsai, C.-J.; Jang, H.; Padhorny, D.; Koza-

kov, D.; McLeish, T. Allostery in Its Many Disguises: From Theory to Applications. *Structure* **2019**, *27*, 566–578.

(200) Fiorin, G.; Klein, M. L.; Hénin, J. Using collective variables to drive molecular dynamics simulations. *Mol. Phys.* **2013**, *111*, 3345–3362.

(201) Ung, P. M.-U.; Rahman, R.; Schlessinger, A. Redefining the Protein Kinase Conformational Space with Machine Learning. *Cell Chem. Biol.* **2018**, *25*, 916–924.

(202) Degiacomi, M. T. Coupling Molecular Dynamics and Deep Learning to Mine Protein Conformational Space. *Structure* **2019**, *27*, 1034–1040.

(203) Óscar, D.; Dalton, J. A.; Giraldo, J. Artificial Intelligence: A Novel Approach for Drug Discovery. *Trends Pharmacol. Sci.* **2019**, *40*, 550–551.

(204) Bonomi, M.; Branduardi, D.; Bussi, G.; Camilloni, C.; Provasi, D.; Raiteri, P.; Donadio, D.; Marinelli, F.; Pietrucci, F.; Broglia, R. A.; Parrinello, M. PLUMED: A portable plugin for free-energy calculations with molecular dynamics. *Comput. Phys. Commun.* **2009**, *180*, 1961–1972.

(205) Case, D. A.; Cheatham III, T. E.; Darden, T.; Gohlke, H.; Luo, R.; Merz Jr., K. M.; Onufriev, A.; Simmerling, C.; Wang, B.; Woods, R. J. The Amber biomolecular simulation programs. *J. Comput. Chem.* **2005**, *26*, 1668–1688.

(206) Berendsen, H.; van der Spoel, D.; van Drunen, R. GROMACS: A message-passing parallel molecular dynamics implementation. *Computer Physics Communications* **1995**, *91*, 43–56.

(207) Phillips, J. C.; Braun, R.; Wang, W.; Gumbart, J.; Tajkhorshid, E.; Villa, E.; Chipot, C.; Skeel, R. D.; KalÃ©, L.; Schulten, K. Scalable molecular dynamics with NAMD. *J. Comput. Chem.* **2005**, *26*, 1781–1802.

(208) Eastman, P.; Swails, J.; Chodera, J. D.; McGibbon, R. T.; Zhao, Y.; Beauchamp, K. A.; Wang, L. P.; Simmonett, A. C.; Harrigan, M. P.; Stern, C. D.; Wiewiora, R. P.; Brooks, B. R.; Pande, V. S. OpenMM 7: Rapid development of high performance algorithms for molecular dynamics. *PLoS Comput. Biol.* **2017**, *13*, e1005659.

(209) Chollet, F., et al. Keras. 2015; https://keras.io (accessed June 17, 2020).

(210) Abadi, M.; Agarwal, A.; Barham, P.; Brevdo, E.; Chen, Z.; Citro, C.; Corrado, G. S.; Davis, A.; Dean, J.; Devin, M.; Ghemawat, S.; Goodfellow, I.; Harp, A.; Irving, G.; Isard, M.; Jia, Y.; Jozefowicz, R.; Kaiser, L.; Kudlur, M.; Levenberg, J.; Mané, D.; Monga, R.; Moore, S.; Murray, D.; Olah, C.; Schuster, M.; Shlens, J.; Steiner, B.; Sutskever, I.; Talwar, K.; Tucker, P.; Vanhoucke, V.; Vasudevan, V.; Viégas, F.; Vinyals, O.; Warden, P.; Wattenberg, M.; Wicke, M.; Yu, Y.; Zheng, X. TensorFlow: Large-Scale Machine Learning on Heterogeneous Systems. 2015; `http://tensorflow.org/(accessedJune17,2020)`, Software available from tensorflow.org.

(211) Paszke, A.; Gross, S.; Chintala, S.; Chanan, G.; Yang, E.; DeVito, Z.; Lin, Z.; Desmaison, A.; Antiga, L.; Lerer, A. Automatic Differentiation in PyTorch. NIPS 2017 Workshop on Autodiff. 2017.

(212) Howard, J., et al. fastai. 2018; https://github.com/fastai/fastai (accessed June 17, 2020).

(213) 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.

(214) 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.

(215) Swenson, D. W. H.; Prinz, J.-H.; Noe, F.; Chodera, J. D.; Bolhuis, P. G. OpenPathSampling: A Python Framework for Path Sampling Simulations. 2. Building and Customizing Path Ensembles and Sample Schemes. *J. Chem. Theory Comput.* **2019**, *15*, 837–856.

(216) Noé, F.; Olsson, S.; Köhler, J.; Wu, H. Boltzmann generators: Sampling equilibrium states of many-body systems with deep learning. *Science* **2019**, *365*.

(217) Carry, J.-C.; Clerc, F.; Minoux, H.; Schio, L.; Mauger, J.; Nair, A.; Parmantier, E.; Le Moigne, R.; Delorme, C.; Nicolas, J.-P.; Krick, A.; Abecassis, P.-Y.; Crocq-Stuerga, V.; Pouzieux, S.; Delarbre, L.; Maignan, S.; Bertrand, T.; Bjergarde, K.; Ma, N.; Lachaud, S.; Guizani, H.; Lebel, R.; Doerflinger, G.; Monget, S.; Perron, S.; Gasse, F.; Angouillant-Boniface, O.; Filoche-Romme, B.; Murer, M.; Gontier, S.; Prevost, C.; Monteiro, M.-L.; Combeau, C. SAR156497, an Exquisitely Selective Inhibitor of Aurora Kinases. *J. Med. Chem.* **2015**, *58*, 362–375.

(218) DrugBank. https://www.drugbank.ca (accessed June 17, 2020).

(219) Clinical Trials. https://clinicaltrials.gov (accessed June 17, 2020).

(220) Gkeka, P.; Evangelidis, T.; Pavlaki, M.; Lazani, V.; Christoforidis, S.; Agianian, B.; Cournia, Z. Investigating the Structure and Dynamics of the PIK3CA Wild-Type and H1047R Oncogenic Mutant. *PLOS Comput. Biol.* **2014**, *10*, 1–12.

(221) Gkeka, P.; Papafotika, A.; Christoforidis, S.; Cournia, Z. Exploring a Non-ATP Pocket for Potential Allosteric Modulation of PI3Kα. *J. Phys. Chem. B* **2015**, *119*, 1002–1016.

(222) Hub, J. S.; de Groot, B. L. Detection of Functional Modes in Protein Dynamics. *PLOS Comput. Biol.* **2009**, *5*, 1–13.

(223) O'Connell, J. P.; Porter, J. R.; Lawson, A.; Kroeplien, B.; Rapecki, S. E.; Norman, T. J.; Warrellow, G. J.; Arakaki, T. L.; Burgin, A. B.; Pitt, W. R.; Calmiano, M. D.; Schubert, D. A.; Lightwood, D. J.; Wootton, R. J. Novel TNFα structure for use in therapy. 2015; PCT/E P2015/074491.

(224) Barati Farimani, A.; N. Feinberg, E.; Pande, V. Binding Pathway of Opiates to μ-Opioid Receptors Revealed by Machine Learning. *Biophys. J.* **2018**, *114*, 62a–63a.

(225) N. Feinberg, E.; Barati Farimani, A.; Uprety, R.; Hunkele, A.; Pasternak, G.; Majumdar, S.; Pande, V. Machine Learning Harnesses Molecular Dynamics to Discover New μ-Opioid Chemotypes. 2018; arXiv:1803.04479 [q-bio.BM], https://arxiv.org/abs/1803.04479 (accessed June 17, 2020).

(226) Schmidt, M.; Lipson, H. Distilling free-form natural laws from experimental data. *Science* **2009**, *324*, 81–85.

(227) Cheng, B.; Engel, E. A.; Behler, J.; Dellago, C.; Ceriotti, M. Ab initio thermodynamics of liquid and solid water. *P. Natl. Acad. Sci. USA* **2019**, *116*, 1110–1115.

(228) Brieuc, F.; Schran, C.; Uhl, F.; Forbert, H.; Marx, D. Converged quantum simulations of reactive solutes in superfluid helium: The Bochum perspective. *J. Chem. Phys.* **2020**, *152*, 210901.

(229) Kapil, V.; Wilkins, D. M.; Lan, J.; Ceriotti, M. Inexpensive modeling of quantum dynamics using path integral generalized Langevin equation thermostats. *J. Chem. Phys.* **2020**, *152*, 124104.

(230) Häse, F.; Kreisbeck, C.; Aspuru-Guzik, A. Machine learning for quantum dynamics: deep learning of excitation energy transfer properties. *Chem. Sci.* **2017**, *8*, 8419–8426.

(231) Jasinski, A.; Montaner, J.; Forrey, R. C.; Yang, B. H.; Stancil, P. C.; Balakrishnan, N.; Dai, J.; Vargas-Hernández, R. A.; Krems, R. V. Machine-learning-corrected quantum dynamics calculations. 2020; arXiv:2001.06592 [physics.chem-ph], https://arxiv.org/abs/2001.06592 (accessed June 17, 2020).

Graphical TOC Entry