

Learning How to Simulate

Applying machine learning methods to
improve molecular dynamics simulations

Adrià Pérez Culubret

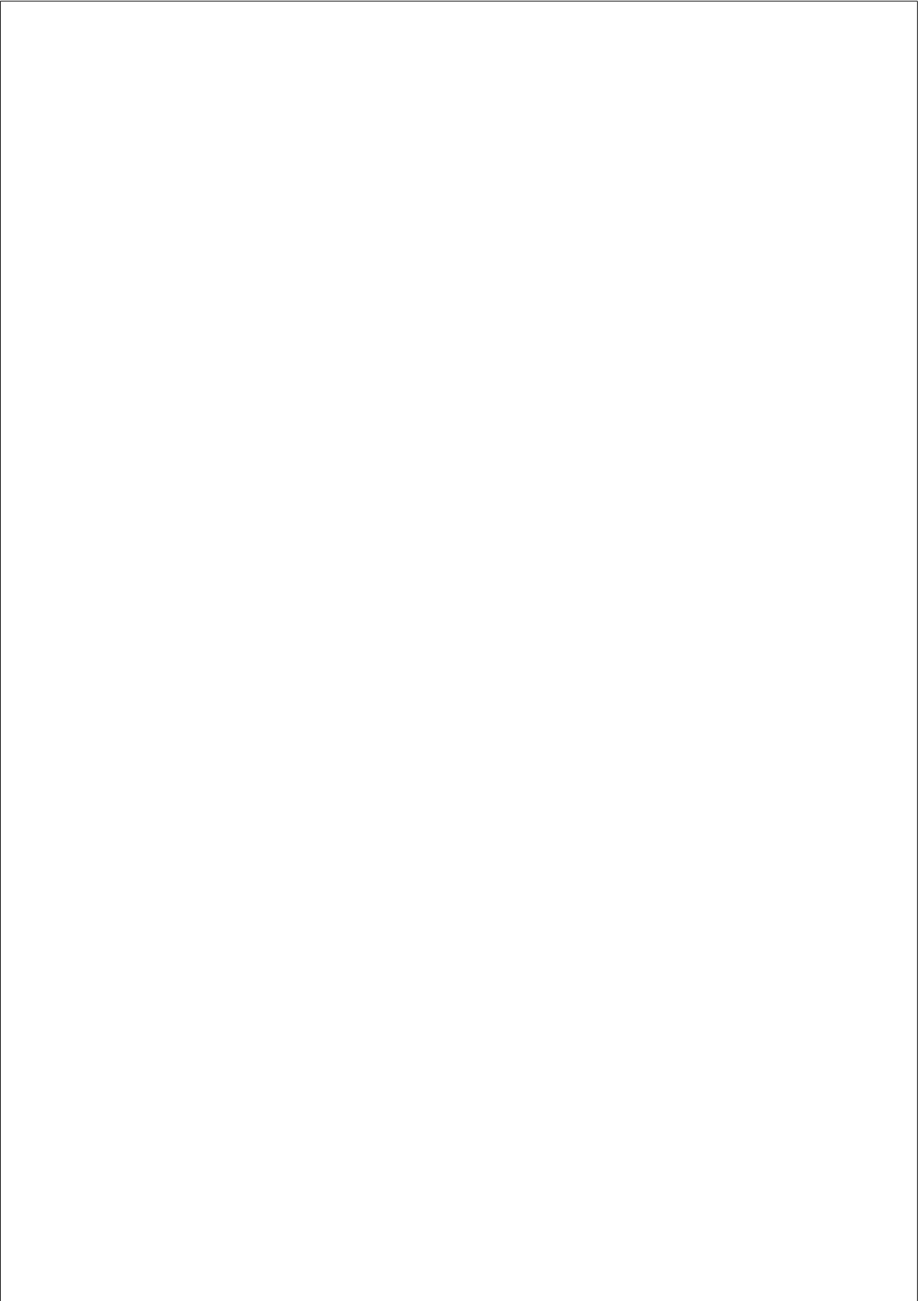
TESI DOCTORAL UPF / year 2021

THESIS SUPERVISOR

Dr. Gianni De Fabritiis

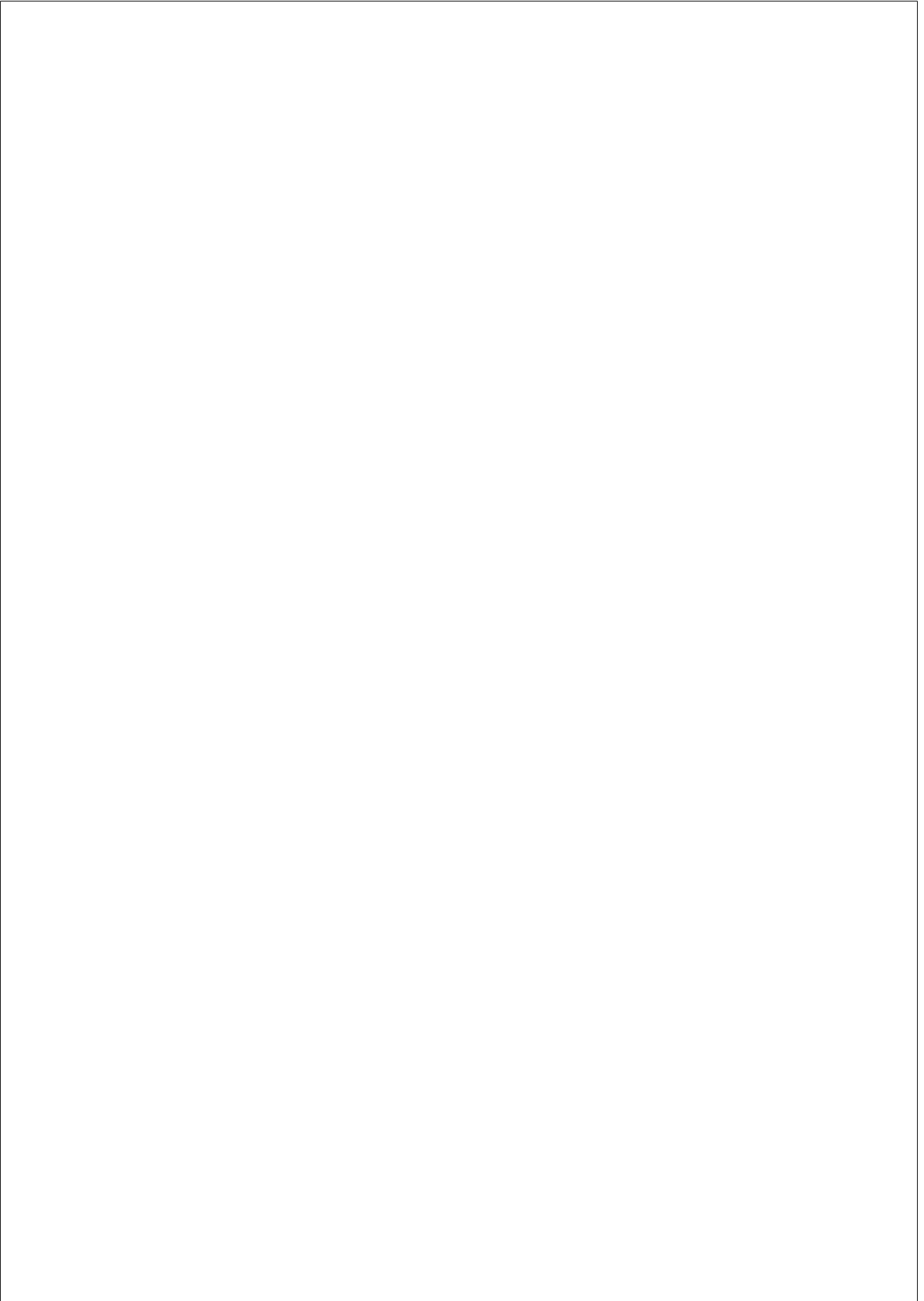
Department of Experimental and Health Sciences





“Time — yet not the time told by the station clock, moving with a jerk five minutes at once, but rather, the time of a tiny timepiece, the hand of which one cannot see move, or the time the grass keeps when it grows, so unobservably one would say it does not grow at all, until some morning the fact is undeniable — time, a line composed of a succession of dimensionless points, time, we say, had gone on, in its furtive, unobservable, competent way, bringing about changes.”

— Thomas Mann, *The Magic Mountain*



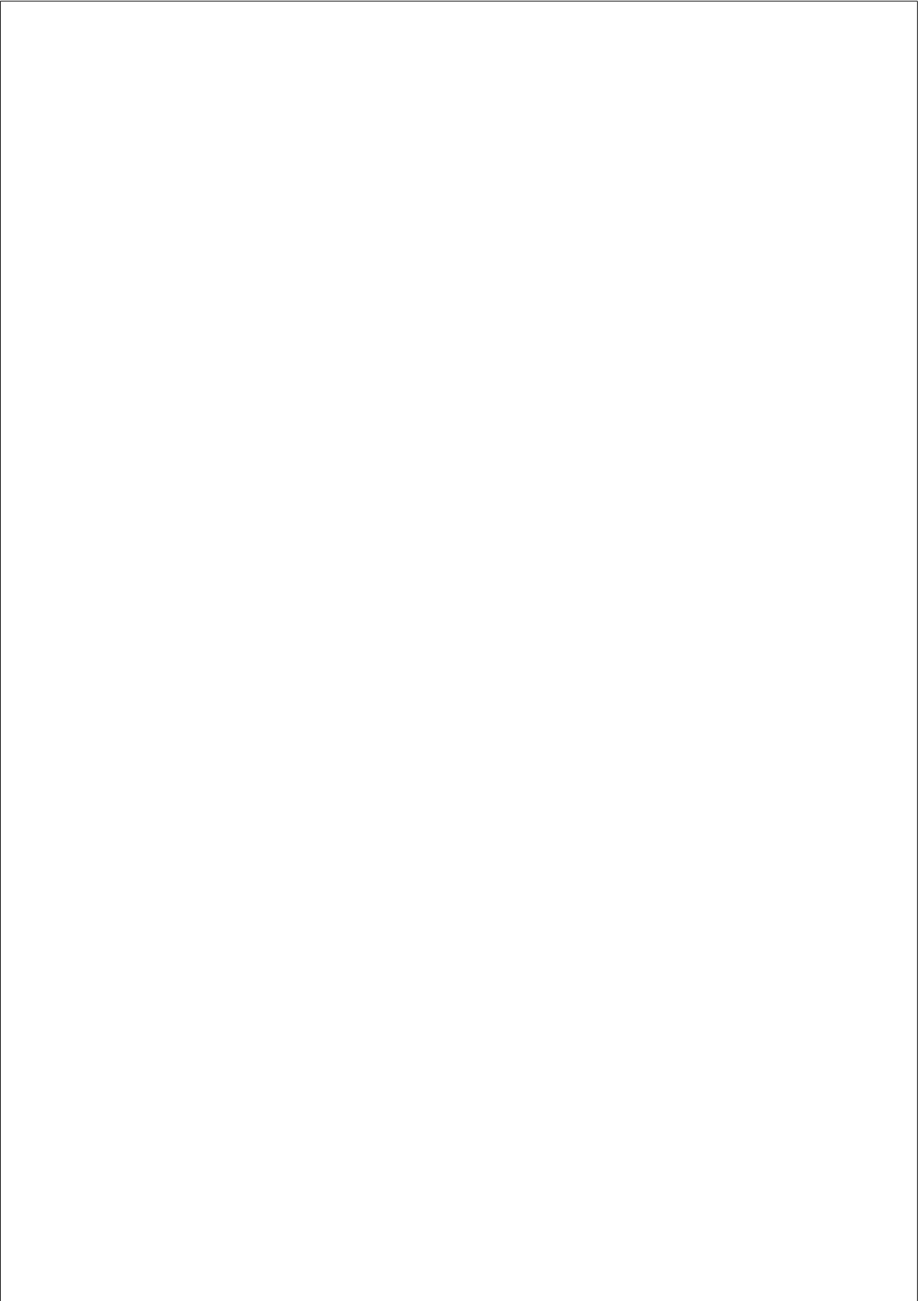
Acknowledgements

First and foremost, I would like to thank my mother, Montserrat. God knows where I would be right now if it wasn't for her strength and resilience. I owe her everything.

I would like to thank Gianni, my supervisor, for trusting in me and my capacities, and granting me the confidence I needed during my PhD. Obviously, I would also like to thank my labmates Pablo, José, Miha, Stefan, Gerard, Maciej, Dominik, Davide and Gabriele, as well as the people from Acellera, João, Alejandro, Alberto, David, Raimondas, Irene and Franck, for their continuous support and for creating such an enjoyable environment.

I would like to thank all the anonymous users at GPUGRID, without whom I would have been able to make my thesis, and for their altruistic donations for science.

I would also like to thank all my family and all my friends that shared this phase with me and helped me go through it. I'd like to list them all, but don't want to risk missing anyone. Last, but not least, thanks to Laia, for helping me wake up at 7:00 am every day to catch the train during my first year, and for putting up with me during my last year.



Abstract

Characterizing protein dynamics is critical to understand the connection between sequence and function. Molecular dynamics simulations are one of the predominant techniques to study protein dynamics due to their capacity to capture dynamical processes of proteins across different timescales with atomic resolution. However, molecular dynamics has limitations that so far limited the ability to become a surrogate model of real protein dynamics, mainly sampling limitations due to the high computational cost and force field accuracy.

In this thesis we tackle these issues with the latest advances in machine learning. In the first part of this thesis, we will develop a novel adaptive sampling algorithm inspired by reinforcement learning methods, and we will later apply it to reconstruct the full binding event between an intrinsically disordered protein and its binding partner. In the second part of this thesis, we develop TorchMD, a deep learning framework for molecular simulations and apply it to learn a coarse-grained potential for protein folding simulations.

Resum

Caracteritzar la dinàmica de les proteïnes és essencial per tal d’entendre la connexió entre seqüència i funció. La simulació de dinàmiques moleculars és una de les tècniques principals per a estudiar la dinàmica de proteïnes per la seva capacitat de capturar processos dinàmics computacionals en diferents escales temporals amb resolució atòmica. Tanmateix, hi ha limitacions que impedeixen que la simulació de dinàmiques moleculars es converteixi en un model substitutiu de les dinàmiques reals de proteïnes, principalment per limitacions de mostreig, causades per l’alt cost computacional de les simulacions, i la inexactitud dels camps de força utilitzats.

En aquesta tesi doctoral tractem aquestes limitacions mitjançant els últims avenços en aprenentatge automàtic. En la primera part de la tesi, desenvoluparem un nou algoritme de mostreig adaptatiu inspirat en mètodes d’aprenentatge reforçat, que aplicarem per a reconstruir la unió entre una proteïna desordenada i la seva parella d’unió. En la segona part de la tesi, desenvoluparem TorchMD, una llibreria d’aprenentatge profund per a simulacions de dinàmica molecular, que aplicarem per a aprendre un potencial “coarse-grained” per a simulacions de plegament de proteïnes.

Preface

The question of life, its existence, functioning, or even its meaning. Those are the reasons that, diffusely, brought me to science, to study biochemistry, bioinformatics, and to finally end up here, writing my thesis.

When I started learning about molecular biology and the complex functioning of cells, I was marveled by all the intrinsic machinery happening there, but also quite astonished at its vast complexity. Metabolism, signaling, replication, all happening in imperceptible scales. What most perplexed me was how this set of interconnected processes and functions, which appeared to me as a massive and chaotic, but yet incredibly coordinated system, was entirely defined by the most basic and essential rules of chemistry and physics.

A couple of years ago I discovered an article, titled "Life's Irreducible Structure", that brought me a bit sense to my perplexity. In there, the Hungarian-British polymath Michael Polanyi argues that life cannot be described solely by the most basic physicochemical laws, and proposes instead a dual-control system, where boundary conditions impose restrictions onto the physicochemical laws to, paraphrasing Polanyi, "harness" them. These boundary conditions cannot be defined in terms of the physicochemical laws they harness, in the same way that one cannot define the content of this text solely by the vocabulary or the individual letters used in it. In other words, these boundary conditions are irreducible to the lower laws and principles that they are harnessing, and constitute themselves a higher level, structuring and organizing whatever is left indeterminate by the lower level. Life itself is irreducible, can't be explained with just atoms interacting with each other. No physical or chemical rule differentiates our genetic code from a random sequence of nucleotides. Understanding life goes through deciphering how biology imposes these boundary conditions and maintains itself.

The thesis you are about to read makes a small step towards this direction, and focuses on the study of protein function by means of molecular dynamics simulations. Paradoxically, to do so, we are going to play with atoms interacting with each other. From the bottom-up, we will try to integrate, with the help of machine learning algorithms, all the small steps

we take, atom by atom, femtosecond by femtosecond, in order to understand the high level principles of molecular biology.

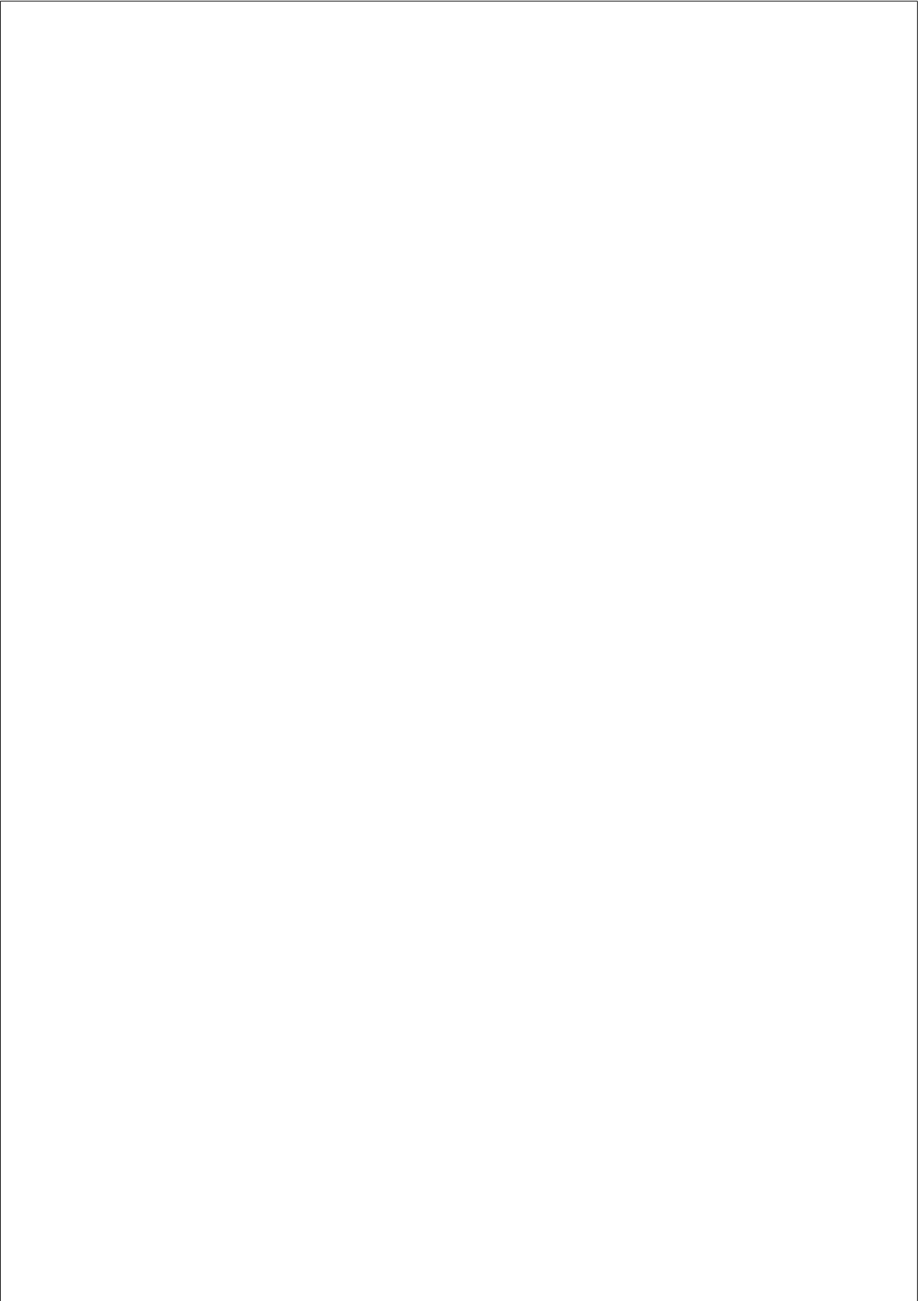
Contents

List of figures	xiii
1 INTRODUCTION	1
1.1 Understanding protein function	1
1.1.1 Functional role of protein dynamics	2
1.1.2 Methods for investigating protein dynamics	5
1.2 Simulating biological timescales: balancing accuracy and efficiency	8
1.2.1 Molecular modeling	9
1.2.2 Molecular Dynamics simulations	12
1.3 Simulations meet machine learning in structural biology	18
1.3.1 Neural Network potentials: learning accurate energy potentials	20
2 OBJECTIVES	23
2.1 Define adaptive sampling algorithms under a strong mathematical framework	23
2.2 Build a deep learning framework for molecular dynamics simulations	24
3 PUBLICATIONS	27
3.1 AdaptiveBandit: A multi-armed bandit framework for adaptive sampling in molecular simulations	27

3.2	Binding-and-folding recognition of an intrinsically disordered protein using adaptive molecular dynamics	38
3.3	TorchMD: A Deep Learning Framework for Molecular Simulations	55
3.4	Simulations meet machine learning in structural biology	56
4	DISCUSSION	77
4.1	AdaptiveBandit	77
4.2	TorchMD	79
5	CONCLUSIONS	81
6	APPENDIX: OTHER PUBLICATIONS	83
6.1	Machine Learning of Coarse-Grained Molecular Dynamics Force Fields	83
6.2	Coarse graining molecular dynamics with graph neural networks	85

List of Figures

1.1	Illustration of a synapse between neurons	2
1.2	Time and size scale of different biomolecular events. . .	4
1.3	Force terms in classical force-fields.	10



Chapter 1

INTRODUCTION

1.1 Understanding protein function

Proteins are considered the machinery of life, as they are responsible of almost all of the processes happening in cells [1]. Let's take a synapse between two nerve cells as an example (Figure 1.1). The machinery behind a synapse happens through the combined effort of a network of proteins interacting with each other, with a diverse set of functions, ranging from structural and transport functions to chemical reaction catalysis and signal transduction. Deficiencies in protein function, related to their misfolding, denaturation and aggregation, are the cause behind various diseases [2, 3]

Proteins are polymeric chains formed by the concatenation of any of the 20 essential amino acids. For any given protein and its unique sequence of amino acids, the poly-peptide chain folds into a specific structure, named the native structure. A critical aspect of proteins is their functional specificity, that is, how they are specialized into particular functions. Enzymes, antibodies, transporters and receptors are all tailored to interact only with molecules within a narrow set of chemical characteristics. Specificity also means that a single mutation or a slight change in a ligand's atomic composition can be translated into an abrupt change in affinity or bioactivity. Diversity and specificity of function are essential

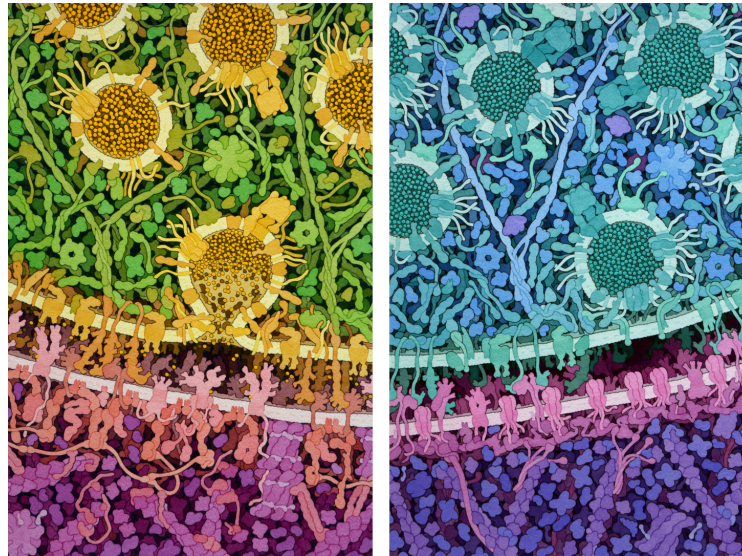


Figure 1.1: **Illustration of a synapse between neurons** Illustration depicting an excitatory synapse (left) and an inhibitory synapse (right) at a cellular scale. Several different types of proteins are shown, such as membrane receptors, transporters, enzyme complexes or protein filaments. The overall picture serves as a graphical example on how a synapse is the result of the combined effect of various cell functions, happening through a dense network of protein machinery. Illustration by David S. Goodsell, the Scripps Research Institute.

characteristics for protein biological function, and all of this unfolds from the unique amino acid arrangement in the poly-peptide chain, which is ultimately encoded in DNA sequences. Understanding how sequence determines protein function, still remains a fundamental problem of biology.

1.1.1 Functional role of protein dynamics

Ever since the first crystal structure was resolved in 1958 [4], there has been an exponential increase in our structural knowledge of proteins.

The three-dimensional structure of proteins have been extensively used to explain the molecular mechanisms of numerous protein functions [5]. However, all these static snapshots of proteins can give the false impression of proteins as rigid macromolecules, stabilized at their native fold during their lifetime. On the contrary, as our structural insight continues to grow, it is becoming increasingly clear that proteins behave as dynamic systems, existing as an equilibrium of conformational states, and that this dynamic behaviour is critical for their biological function [6, 7, 8, 9, 10].

One exemplar case of proteins functioning as dynamic systems can be found with intrinsically disordered proteins (IDPs). These particular type of proteins are able to perform their function without having a stable folded state, proving that a stable folded structure is not needed for function [11, 12] and providing further proof of the functional importance of protein dynamics. IDPs account for 30% of the human proteome [13, 14], and have roles mostly in cellular signaling and regulation [11, 15, 16]. Because they do not have a clear global minimum, their structure fluctuates between several conformational states. Many IDPs stabilize into a stable folded structure once they bind to their binding partners [17, 18, 19, 20, 21, 22, 23].

Defining proteins as dynamics systems means that to completely characterize a protein’s structure, we must be able to describe its entire conformational energy landscape, quantifying the relative probabilities of all conformations that coexist in equilibrium (thermodynamics) and the energy barriers separating them (kinetics). To put it simply, we must add a fourth dimension, time, into our structural analysis, and describe how the three-dimensional atomic coordinates change through time [7].

Dynamic processes can be classified depending on their length scale (from local fluctuations to global motions) or on their time scale (from fast to slow timescales) (Figure 1.2) [6, 7]. Generally, local motions occur in faster timescales, while global large-scale motions are encountered in slower timescales. Fast timescales go from bond vibrations at the femtosecond timescale and side-chain rotations at the picosecond timescale to local collective motions of atoms at a nanosecond timescale (such as loop motions or backbone fluctuations). On the other hand, slow timescales

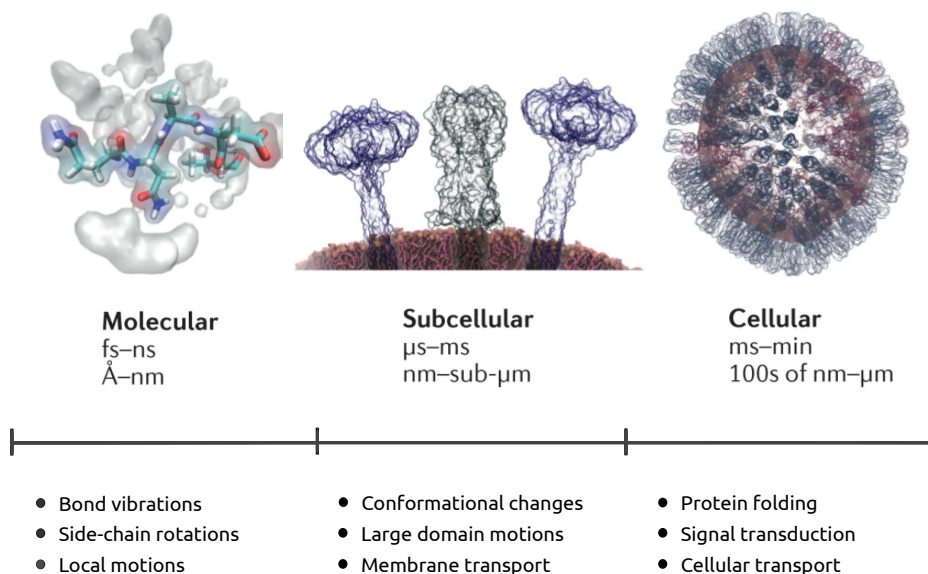


Figure 1.2: **Time and size scale of different biomolecular events.** Different structures and structural models are shown, along with their related timescales and biomolecular events, showing the multiscale nature of proteins and their cellular functions. Adapted from [24].

define events happening at the microsecond timescale or slower, and they go from small conformational changes and allosteric transitions at the nanosecond to microsecond timescales to larger domain motions, up to the complete folding of the polypeptidic chain, at the range of milliseconds to seconds. These slower timescales are also directly related to the relevant biological functions proteins perform, such as enzyme catalysis, membrane transport or signal transduction through protein-protein interactions and, consequently, they are the subject of interest of biological investigation.

Dynamic processes on protein systems occur through a hierarchy of events across different time and length scales. Such hierarchy is linked to the hierarchical structure of proteins [6, 8, 7], as the slower timescales arise from the collective effect of many smaller local fluctuations. In their review, Henzler-Wildman and Kern [7] indicate that the improbability of slower processes results from the collective nature of their local origin, and suggest that the hierarchical dynamics of proteins is defined by the structural restrictions encoded in the amino acid sequence. They even go further, and define the dynamic landscape of a protein as its “personality”, and propose that the conformational substates sampled by a protein are the result of evolutionary selection on states needed for protein function.

1.1.2 Methods for investigating protein dynamics

Ideally, for a complete description of the dynamic landscape of a protein, one would like to determine both its conformational states and the rates of interconversion between them. Due to the structural heterogeneity of proteins and the time and length scales in which their dynamics occur, there is not a single technique that is able to obtain full coverage of the whole spatio-temporal scale, and thus a diverse set of experimental and computational methodologies is required depending on the dynamic events at study.

From the experimental side, the most popular method by far is X-ray crystallography. It was the first methodology that allowed to obtain models of protein structure at an atomic resolution [4]. Since then, more than 158.000 structures have been deposited in the Protein Data Bank (PDB) [25] (which account for nearly 88% of the structures). X-ray crystallography is the main factor behind the exponential growth of structural biology in the past 70 years. Although the majority of structures obtained are single snapshots of the averaged low energy conformations, nowadays there are time-resolved X-ray crystallography methods that enable the study of protein dynamics (X-ray free electron lasers, or XFEL) [26, 27]. Despite its huge popularity, X-ray crystallography still carries the major inconvenience of requiring a homogeneous crystal of the protein, which it is not

always possible to obtain. That is specially true for membrane proteins [28], highly relevant for drug discovery.

Secondly, representing the 7.45% of structures in PDB, we have nuclear magnetic resonance (NMR). The advantages of NMR are the possibility to obtain dynamic timescales and to perform measurements directly in solution, without needing to crystallize the protein. However, the size of systems that can be studied through NMR is limited, as the spectral processing gets complicated for large macromolecules. NMR has been very useful for the study of IDPs [29, 30] due to their inherent flexibility and the impossibility to observe them through X-ray crystallography.

Lastly, we got 3D cryo-electron microscopy (cryoEM). Although it only accounts for 4.5% of the structures in PDB, it has been rising in popularity over the last five years due to technical improvements that have been increasing the achievable resolution [31, 32]. In 2017, the Nobel prize recognized the developments on cryoEM for biomolecule structure resolution, and since then, the number of uploaded cryoEM structures surpasses the NMR ones. As with NMR, crystallization is not required, and only needs small amounts of sample. Additionally, cryoEM enables the possibility to perform single-particle studies [33, 34], that avoid having to average out over all structures present in samples. The main drawback of cryoEM has always been its low resolution compared to other methods, up until now. By the end of 2020, atomic resolution structures ($<1.25\text{\AA}$) were obtained using single-particle cryoEM [35, 36], which will probably result in yet even more novel cryoEM structures.

There are other complementary methods, such as FRET [37] or SAXS [38, 39], that can help in obtaining dynamic information from more challenging systems, but usually at the expense of lower resolution. An extensive review of additional methods to study protein dynamics in solution can be found here [40].

On the computational side, the main method to study protein dynamics is molecular dynamics (MD) simulations. This thesis is centered around MD simulations, and they will be discussed more deeply in the next section. Compared to experimental methods, the main advantage of MD simulations is their unique ability to obtain a complete description

of dynamics with atomic resolution and to explain slow events at the millisecond to microsecond timescale with a femtosecond level of time resolution. But we should not see MD simulations as a complete substitute for experiments. Rather than an alternative method, MD simulations are used as a complement to experiments, mainly because MD simulations depend on experimental results to be performed (although this is likely to change with the arrival of AlphaFold [41]), but also because MD simulations can give a deeper understanding of experimental results, guiding and inspiring new experiments to perform, in a combined effort for understanding protein dynamics.

1.2 Simulating biological timescales: balancing accuracy and efficiency

”” *The underlying physical laws necessary for the mathematical theory of a large part of physics and the whole of chemistry are thus completely known, and the difficulty is only that the exact application of these laws leads to equations much too complicated to be soluble. It therefore becomes desirable that approximate practical methods of applying quantum mechanics should be developed, which can lead to an explanation of the main features of complex atomic systems without too much computation.*

— Paul Dirac [42] (1929)

The complete description of the dynamic landscape of a protein requires to determine **1)** its conformational states, **2)** the relative probabilities of states and **3)** the rates of interconversion between them. The main computational method to do so is molecular simulations, which is a numerical approach to obtain a microscopic description of the atomic motions in a system, allowing us to compute the macroscopic properties of it. There are many ways to perform molecular simulations, and they mainly differ in their accuracy to describe the potential energy of the system, the forces acting between atoms (molecular modeling) and the methods used to sample the conformational landscape. The predictive power of simulations is only as good as the description of the potential energy. Nevertheless, an accurate potential energy description is of no use if we cannot obtain enough samples from it due to computational limitations. Choosing the correct molecular model or sampling method mainly depends on the system to simulate, and the balance between the required accuracy and the computational cost of simulation.

1.2.1 Molecular modeling

The main purpose of molecular models is to produce a fully predictive model of the physical properties of a molecular system. However, no such perfect model exists, and approximations are needed, especially when used for simulations of biomolecules at the microsecond timescale.

Ideally, to predict any physical property of a molecular system, the time-dependent Schrödinger equation can be solved to obtain the wave function of the system and observe how it evolves dynamically through time [43]. In practice, exact solutions for it are impossible to obtain, and several approximations are required to model molecular systems. Depending on the level of detail required to describe a particular system, we will have to choose between different levels of approximation. However, each level of approximation involves a compromise with its accuracy. Consequently, there are several different molecular models available, with varying accuracy, system size capacity and computational cost. Here, I will broadly divide the possible molecular models into three main categories: quantum mechanics (QM) models, classical mechanics models and coarse-grained models.

QM models are the most accurate, since they are able to provide information of the electronic structure of molecules. QM calculations are widely applied on small molecules (around tens of atoms) to compute their electronic energy, with accuracy often close to experimental methods. Density functional theory (DFT) methods [44, 45, 46] are one of the most relevant and common approaches for QM calculations, providing energy estimates with sufficient accuracy at an affordable computational cost, handling systems of around hundreds of atoms. The application of QM methods for atomic models of protein dynamics is very limited, as the system sizes are prohibitive.

QM calculations are always computationally demanding, and their applicability is only limited to small systems. In order to extend our reach to larger molecules, we have to resort to molecular mechanics models, where atomic systems are modeled using classical mechanics, treating atoms as classical particles and electronic degrees of freedom as bonds. Assum-

$$E_{total} = \underbrace{\sum_{bonds} K_r (r - r_{eq})^2 + \sum_{angles} K_\theta (\theta - \theta_{eq})^2 + \sum_{dihedrals} \frac{V_n}{2} [1 + \cos(n\phi - \gamma)]}_{\text{Bonded}} + \underbrace{\sum_{i < j} \left[\frac{A_{ij}}{R_{ij}^{12}} - \frac{B_{ij}}{R_{ij}^6} + \frac{q_i q_j}{\epsilon R_{ij}} \right]}_{\text{Non-bonded}}$$

Figure 1.3: **Force terms in classical force-fields.** Figure shows the basic equations for bonded and non-bonded terms used to compute the potential energy of a system. Figure from [49].

ing the Born-Oppenheimer approximation, the potential energy of atomic systems is calculated as a function of atomic coordinates using classical force fields. These simple classical force-fields are parameterized using QM calculations and experimental data [47], and their accuracy has drastically improved over the years [48]. Classical force-fields provide a significant improvement in computational efficiency, but come at the expense of neglecting quantum phenomena, as electronic structure is completely ignored. Events such as bond formation or cleavage, chemical reaction mechanisms or charge and spin distribution cannot be computed with these types of force-fields.

Classical force-fields are the best option when dealing with biological systems, as it enables the possibility to simulate single proteins at a microsecond to millisecond timescale, with atomic resolution. The exact functional form of these force-fields varies over different implementations, but the basic formulation behind them is the same. Essentially, the potential energy is calculated as the sum of individual terms describing different atomic interactions. These terms are separated between bonded and non-bonded terms. Bonded terms include bonds, angles and dihedrals, where the first two are modeled through harmonic potentials. Non-bonded terms include electrostatic and Van der Waals forces, modeled through Coulomb and Lennard-Jones potentials.

However, even with molecular mechanics approximations, obtaining a detailed dynamic description of large biomolecular systems and related processes of biological relevance remains intractable. For these cases, the complexity of the molecular model can be reduced even further to the so-called coarse-grained models, forgiving the atomistic resolution, and grouping together several atoms in a single particle or “bead” [50]. In general, we can divide coarse-graining approaches between bottom-up and top-down approaches. Bottom-up approaches construct the coarse-grained model based on a “fine-grained” higher-resolution model, such as an atomistic model. Top-down approaches, instead, try to reproduce macroscopic properties of the system, commonly measured by experiments, where the coarse-graining model is itself a “finer-grained” model of such macroscopic observations. Some of the most successful coarse-graining models are MARTINI [51, 52], Rosetta [53], AWSEM [54] and CABS [55]. Some of the most successful applications of coarse-grained models include protein structure prediction [56, 57], protein-protein-protein-lipid interactions in cellular membranes [58, 59] and DNA hybridization [60].

Combined methods using different modeling approaches also exists, with great success. Hybrid QM/MM simulations combine QM calculations for small regions in the system, such as catalytic sites, that require information on the electronic configuration with molecular mechanics onto the rest of the system. Hybrid QM/MM is key where electronic effects are essential for the dynamics at study, such as enzyme catalysis or pH dependant events. The main contributors of this method received a Nobel Prize in 2013 [61]. There is a huge variety of multiscale models available, depending on the size and time scales. An extensive review on multiscale modeling methods, from a molecular to a cellular level, can be found in[24]

1.2.2 Molecular Dynamics simulations

Due to the size and timescales involved, Molecular dynamics (MD) simulations is the main computational method to study protein dynamics. The concept behind MD simulations is fairly simple. A molecular system (in our case, a protein of interest) is represented with atoms as point masses, with an initial randomized velocity, under the influence of a classical force field. The system is then propagated through time, in the order of a few femtoseconds, by integrating Newton’s equations of motion in small discrete time steps, using numerical schemes such as Velocity Verlet [62]. The propagator moves the system forward in time, using the forces obtained by differentiating the potential defined by the force field over the many-body interactions between atoms and obtaining new velocities and positions for all particles. This results in a trajectory of our molecular system, describing its evolution through time and the ensemble of conformations sampled from it.

While MD is the best suited computational method for protein dynamics, it is far from perfect. As with any molecular simulation method, it comes with its limitations in speed and accuracy. First of all, the assumption of a molecular mechanics model comes with the aforementioned limitations of not being able to represent chemical reactions or bond cleavage, making enzyme catalysis impossible to describe or totally neglecting the effect of pH, with side-chain protonation states remaining fixed throughout the simulation. Secondly, and despite the simplicity behind the simulation algorithm, it is computationally expensive to sample relevant biological timescales of proteins. Due to the numerical nature of integration methods, and to maintain numerical stability in the simulation, the integration time step has to be smaller than the fastest dynamic process occurring in the system. Thus, the integration time step cannot grow larger than a few femtoseconds, as bond vibrations happen in the order of tens of femtoseconds. This limitation then causes that to reach relevant timescales, a large number of integration steps have to be performed, in the order of nanoseconds (10^6 fs) or microseconds (10^9 fs). Performing a hundred million integration steps over $10.000 \sim 100.000$ atoms is no

trivial task, even for modern computers, and thus the reason why computational cost is one of the main limiting factors of MD simulations.

Despite this, MD is the only way we can simulate protein dynamics. Much of the work done in the MD field is going towards pushing its boundaries, both in accuracy and sampling capabilities. The first MD simulation of protein dynamics ever performed, back in 1977, consisted only of a 9.2 ps long trajectory of the bovine pancreatic trypsin inhibitor (BPTI) in vacuum [63]. Nowadays, thanks to the many improvements accomplished in the field of MD simulations, microsecond long trajectories are within reach of desktop computers and simulations in the order of milliseconds can be obtained using high performance computing (HPC) resources. The incremental capacities of MD simulations are the result of both software/hardware improvements and novel algorithms for improved sampling and analysis, transforming MD simulations into a high-throughput approach.

Hardware and software improvements

Even though MD simulations are very simple to execute, the computational cost for reaching relevant timescales for protein dynamics is large, due to the aforementioned issue with integration steps. During the first three decades of MD simulations, conventional CPUs were the only available processing units. In order to reach the necessary timescales, MD codes were parallelized and optimized to run on multiprocessor supercomputers in HPC systems [64]. Some effort was even invested in the development of specialized hardware, such as the development of the special-purpose supercomputer Anton [65] (which recently has released its new iteration, Anton-3 [66], able to perform 100 microseconds per day in a 1 million atom system). However, HPC systems were not a consistent solution. Many researchers interested in performing MD simulations do not have the resources to perform such long simulations, nor access to HPC resources, and could only perform nanosecond trajectories in their commodity desktop computers.

Fortunately, this is not the case anymore. Back in 1995, the indus-

try of computational 3D graphics was revolutionized by the release of the first modern graphical processing unit, or GPU (although the term was first coined by Nvidia in 1999). GPUs were a specialized type of processors prepared to deal with extremely parallel computation, such as computational 3D graphics. These type of specialized processors already existed in professional environments, but the release of GPUs were a game-changer for personal computer graphics due to their low cost and performance. The industry of computational 3D graphics kept growing, and GPUs evolved to a more general device, especially with the release of OpenCL and CUDA development environments, which allowed programmers to use GPU computing. This transported GPU computing to many other fields and applications outside of computer graphics. One of the fields that benefited from it was MD simulations, with the development of GPU-based MD simulation codes, such as ACEMD [67], AMBER [68, 69], Desmond [70] or OpenMM [71], that made use of GPU’s intrinsic parallelization to massively increase the speed of simulations. Nowadays, a single GPU produces simulations at the rate of microseconds per day on systems of around ~ 20.000 atoms.

The arrival of GPU computing to MD simulations did not only impact the domestic capabilities of performing MD simulations. GPU-based HPC resources, GPU clusters or distributed computing projects such as Folding@Home or GPUGRID started to appear, allowing researchers to perform simulation experiments that reached aggregate times at the order of milliseconds [72, 73]. Instead of performing one single long trajectory, the most optimal way to use these computational resources in terms of total aggregate time produced is by performing hundreds or thousands of short parallel simulations. Although clearly powerful, parallel MD simulations entail additional issues in terms of data analysis and sampling. Switching a single long trajectory for many short trajectories is an inconvenient if we are interested in sampling slow timescales from our system. Notwithstanding this, hardware and software improvements in MD simulation also came with the development of new analysis and sampling methodologies, making the change from a single trajectory paradigm to high-throughput MD simulations possible.

Markov state model analysis

Performing high-throughput MD simulations is only valuable if we can integrate all the information obtained across all the generated trajectories. Markov state model (MSM) analysis [74, 75] constitutes the cornerstone of high-throughput MD, as it enables the possibility to combine several independent short trajectories into a single kinetic model, building an accurate and interpretable description of the system’s dynamics. MSMs appeared as a necessity to describe complex kinetic events from MD simulations, where single trajectories and basic featurizations (RMSD, radius of gyration, principal component decomposition, etc) failed to do so. The basic concept behind MSMs is that they model the dynamics of the system as a memory-less jump process, where future states are only conditioned on the current state, hence Markovian.

In order to build an MSM, first we need to discretize the sampled conformational space of the system into discrete states, and secondly, use the discretized states to count transitions between them and estimate a transition probability matrix. In other words, to assign each conformation into a state and count all transitions between them. Discretization itself is a multi-step process that tries to simplify the high-dimensional conformational space into a few discrete states so that the transition probability matrix estimation can be performed. Consequently, discretization is critical to correctly model the slow timescales of the system, as it can greatly affect the outcomes of the MSM [74, 76, 77, 78]. The current state-of-the-art process for discretization consists of a combination of dimensionality reduction and clustering of our simulation data. Dimensionality reduction is performed in two steps, where the three-dimensional coordinate space are first projected into simpler features, such as distance between specific atoms or dihedral angles, and later an unsupervised dimensionality reduction method is applied to reduce the dimensionality of the featurized data into less than 10 dimensions. The most used projection method for MSM construction is time-lagged independent component analysis (TICA) [79], an extension of principal component analysis (PCA) that makes use of a time-lagged covariance matrix to better estimate the slowest degrees of

freedom.

Once the simulation data has been discretized, we can count transitions from one state to another and build a count transition matrix, that we can use to estimate the transition probability matrix of the MSM. Since the model estimation is done over the count transition matrix, and transitions are counted independently of the trajectory they come from, the dynamical information from several trajectories can be integrated together. Furthermore, the eigenvectors of the transition matrix correspond to the slow processes of the system, and therefore they can be used to identify the metastable states in it and the timescales at which slow processes occur. Besides that, the transition matrix can also be used to compute other interesting properties of the system, such as free energies, transition pathways and kinetic properties such as on/off rates between two sets of states.

Adaptive Sampling

Performing high-throughput parallel simulations is not straightforward. Because of the shorter lengths of each individual trajectory, it is common that such trajectories are way shorter than the slow timescales of interest, making the sampling of rare events much more difficult in some cases, no matter how many replicas are running in parallel. This is particularly true in cases where we are using MD simulations as a predictive tool, in cases such as protein-ligand binding or protein conformational analysis, where neither the end state or the dynamical path to it is known. To put it in other words, parallel sampling can be a very inefficient way of characterizing equilibrium, sometimes even worse than just a single long MD trajectory.

However, the possibility to perform several independent simulations in parallel, thanks to Markov state modelling, brings us the opportunity to choose different starting points for these simulations. In high-throughput MD, this type of methodology is termed adaptive sampling, and it is widely used to optimize sampling and computational resources. The fundamental idea behind adaptive sampling methods is to perform rounds (or epochs) of parallel simulations and, in between each round, leverage the generated simulation data to make an intelligent selection of the start-

ing conformations for the next round of simulations. By making these decisions, we bias the sampling of our system towards the areas we are interested, avoiding redundant sampling and waste of computational resources. However, despite this sampling bias, no actual force or physical bias is introduced into the system and therefore simulations will still behave under the correct kinetics.

The most relevant part of an adaptive sampling algorithm is the decision making step, as it will define the direction we want to take with our simulations. Usually, this direction is towards an optimal equilibrium characterization of the system at the minimal computational cost possible. There have been several adaptive sampling methods developed which use different decision-making criteria. The first adaptive sampling methods were aimed at reducing the statistical error of the MSM, selecting starting conformations by either the error in mean first passage time [80] or eigenvalues and eigenvectors [81]. One of the most common applications of adaptive sampling methods is to stimulate the exploration of the conformational space [82, 83], particularly important if we need to sample slow and rare events. These type of adaptive sampling algorithms usually rely on discretizing the simulation data and selecting the least sampled regions as new starting points. However, a naive exploration-based criteria might not work in all cases, and additional knowledge is needed to correctly direct sampling. Some algorithms utilize prior knowledge on the system [84, 85, 86] in their selection criteria in order to speed up sampling of certain slow events, such as protein folding [85] or protein-protein interaction [73]. Instead of prior knowledge, geometric features computed from the simulation data itself can also be used in a similar fashion, such as RMSD to the starting structure or pocket volume, in applications such as mutation stability prediction [87] or cryptic pocket detection [88]. It is also worth mentioning the weighted-ensemble algorithms, which distribute sampling along collective variables [89, 90, 91].

In general, adaptive sampling methods have always been designed empirically, and they are not based on any mathematically optimal decision process. There is no theoretical proof that these algorithms are performing optimally in the problem they are trying to solve, the ex-

ploration versus exploitation dilemma often found in optimization algorithms. Some theoretic similarities have been recognized with the multi-armed bandit problem [86, 92] and reinforcement learning [93], but they have not been used to fully describe adaptive sampling under these theoretical frameworks.

1.3 Simulations meet machine learning in structural biology

Note: Parts of this section and the title were taken from my publication 3.4

Without doubt, MD simulations have the potential to become a surrogate model for protein dynamics. The possibility to study protein dynamics and function at biologically relevant timescales with atomic resolution is unparalleled by any other method. In the course of the past two decades, the advances in hardware and software, as well as the development of novel MD sampling and analysis methodologies, have shifted the paradigm of MD simulations towards a high-throughput approach, greatly increasing their capabilities and applications. Today, high-throughput MD simulations are starting to scratch their current boundaries, and new challenges are faced in order to push this boundaries forward.

Despite the technological advances, molecular simulations still constitute a trade-off between accuracy and efficiency. Even with the capacity to generate tens of milliseconds of aggregate simulations, a lot of highly relevant biological timescales are still out of reach. Porting MD simulation engines from CPUs to GPUs meant a huge performance leap, but there is no similar hardware upgrade in sight at the present day. We cannot rely solely on hardware improvements (or in huge HPC facilities) to improve MD sampling capabilities. Furthermore, the accuracy of classical mechanics cannot be improved by computational brute force or by any improvement in simulation efficiency. In order to bring simulations of protein dynamics to new horizons, we must find ways to increase their accuracy, so that they can properly handle phenomena such as bond cleav-

age, pH or enzymatic reactions.

In a recent review we estimated that MD would reach seconds of aggregated sampling using commodity hardware by 2022 [94], generating petabytes of simulation data. For instance, the file size of one second of simulation data of a 60 000-atom system (e.g. a GPCR system) at 0.1 ns per frame is 7.2 Petabytes (reduced to a third using compressed trajectory file formats). This amount of data constitutes a valuable source of information, but the knowledge extracted from it is mainly used to rationalize a particular protein system at hand, not to generalize it to other systems. This thesis envisions a paradigm change for molecular simulations in the near future, where expensive simulations (QM and MD) are not solely used to predict but also to learn general models from where further predictions can be drawn. By doing so, the large computational cost required by simulations becomes justifiable, in particular if the results are more accurate by the use of more expensive simulation methods. The key to build general models and learn from simulations can be found in modern machine learning. By combining the extensive data generated from molecular simulations and the latest machine learning advances, simulations can go beyond a single-use, high-lagged predictive tool.

Synergies between machine learning and simulations are not new. Particularly, in MD simulations, a great example can be found within the Markov state model estimation pipeline, where unsupervised learning methods are used to cluster and reduce the dimensionality of simulation data [95]. However, the reason we can now envision simulations as a data-generation method is primarily because of the recent uprise of deep learning methods [96, 97]. Their capacity to learn complex functions and patterns from massive datasets has revolutionized several fields. The most notable and famous examples can be found in image processing [98] or natural language processing [99]. Deep learning is based on artificial neural networks, a simple mathematical framework organized in layers, each of them performing a matrix multiplication and a non-linear function of the input variables x . The output of a single neuron ϕ in each layer is given by $\phi = f(w^t x + b)$, where w are learnable weights, b is a bias and f is some nonlinear function. Neural networks can have from several to

hundred nested layers, and in such cases is called “deep”. Given enough parameters, a neural network is capable of interpolating any continuous function [100, 101].

There are many different ways to tackle molecular simulations with deep learning [102, 103]. One of the most interesting applications is Boltzmann Generators [104], where MD or Monte Carlo methods are substituted with a neural network that learns to sample conformations from equilibrium, bringing novel ways for sampling rare events. Neural networks have also been used to reproduce the free-energy surface of molecules [105]. Another interesting approach are VAMPnets [106, 107], where a variational score is used to optimize a neural network for Markov state model generation, avoiding the general MSM estimation pipeline, very prone to several rounds of trial-and-error searching for the optimal hyperparameters.

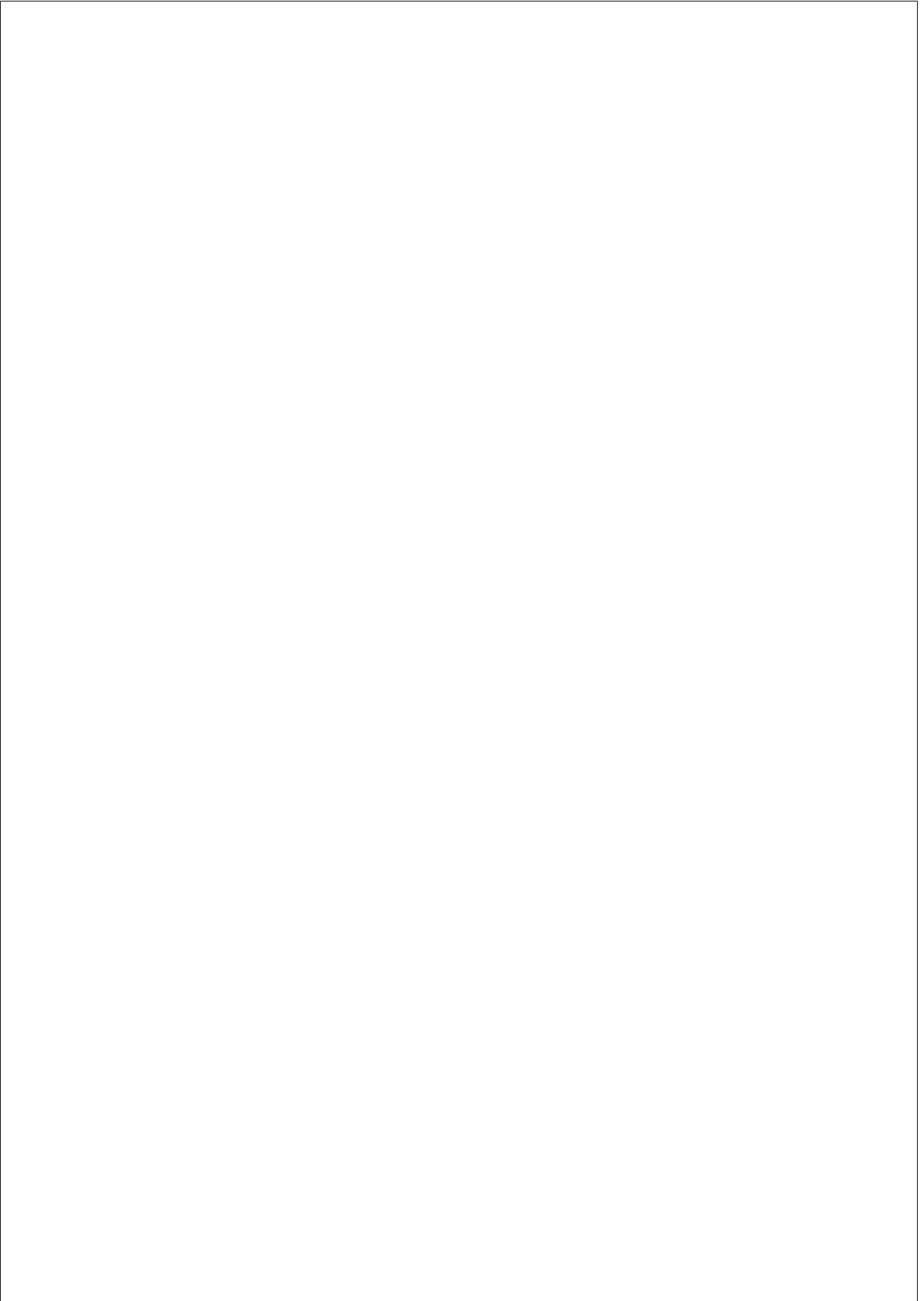
1.3.1 Neural Network potentials: learning accurate energy potentials

For this thesis, we are going to be looking at one of the most promising and powerful machine learning applications for molecular simulations, neural network potentials (NNPs). The core concept behind NNPs is to use a deep neural networks or machine learning models to approximate the potential energy surface of a system and predict forces based on its atomic coordinates. The model is trained using QM data, so that the energy potential is learned with the accuracy of first-principle based methods. In the same way as MD force fields do, forces are true derivatives of the interpolated potential energy surface using the gradients of the neural network. This guarantees that the forces produced by the NNs yield a conservative field [108]. Therefore, NNPs can be used as a force-field and run MD simulations with it. Because inferring from a trained neural network is many orders of magnitude faster than performing ab-initio QM calculations, we could avoid the necessity to use classical mechanistic approximations and greatly improve the accuracy of protein dynamic simulations.

The early work from Behler and Parrinelo [109] has been critical for the development of NNPs, and has sprouted a lot of research on it [110, 108, 111, 112]. The initial effort went in guaranteeing basic physical principles like roto-translational invariance, atom permutation invariance and transferability to the learned potentials. One of the most limiting factors for NNPs is the large amount of training data required to train accurate models. Nonetheless, the cost of generating such datasets is vastly compensated by the benefits of having an accurate and fast NNP-based force-field for protein dynamic simulations. At the moment, NNPs have only been applied to water, small molecules and amino acids, with some reports on a 50ns protein simulation, but there are still many challenges to face in order to routinely perform protein dynamics simulations with NNPs.

In the same way QM data can be used to train machine learning based atomistic force-fields, MD simulation data can be used instead to define coarse-grained potentials. In publication 6.1, a bottom-up force-matching approach is used to train a neural network based coarse-grained potential, and publication 6.2 leverages the SchNet architecture [112, 113] to make the potential transferable. Machine learning can also be applied to automatically learn the coarse-grained mapping [114, 115].

It is clear that NNPs will play a central role in molecular simulations on the coming years. Although there has been a lot of research on developing different potentials and optimized network architectures, there is no available software infrastructure for their application for molecular simulations. The continuous improvement of NNPs requires an infrastructure that supports their usage, validation and continuous development for molecular simulations



Chapter 2

OBJECTIVES

The objective of this thesis has been to look for machine learning solutions to address the sampling limitations of MD simulations, and push its boundaries for its application in the study of protein dynamics. The advancements proposed here try to leverage the synergies between molecular simulations and machine learning, emphasizing the data generation side of MD simulations. First, sampling is tackled directly by defining a novel adaptive sampling algorithm inspired by reinforcement learning, based on the multi-armed bandit theoretical framework. Second, by constructing the framework needed to combine deep learning and molecular dynamics simulation code, in order to advance in the development of NNPs and coarse-grained potentials.

2.1 Define adaptive sampling algorithms under a strong mathematical framework

Adaptive sampling is a key method for high-throughput MD. It has been demonstrated that the opportunity to direct sampling by selecting the starting conformations can accelerate the obtention of slow events by orders of magnitude more. However, most adaptive sampling algorithms are based on empirical rules, without any guarantee of optimality.

Adaptive sampling algorithms face a trade-off between exploration and exploitation. Most of the existing sampling algorithms are fully explorative, which depending on the system, it can be more detrimental than beneficial. There are algorithms that introduce exploitative decisions, but again they work as acquisition functions, and usually require external knowledge of the system.

In the first publication, we introduce high-throughput MD simulations under a reinforcement learning framework, and we describe it in terms of the multi-armed bandit problem, defining the actions and their corresponding rewards intrinsically from the system, without any external knowledge. After that, we solve the bandit problem by means of an upper confidence bound algorithm, which performs optimally in this environment. The main objective is to provide a framework for adaptive sampling algorithms so that they can be adapted for optimal performance in any system.

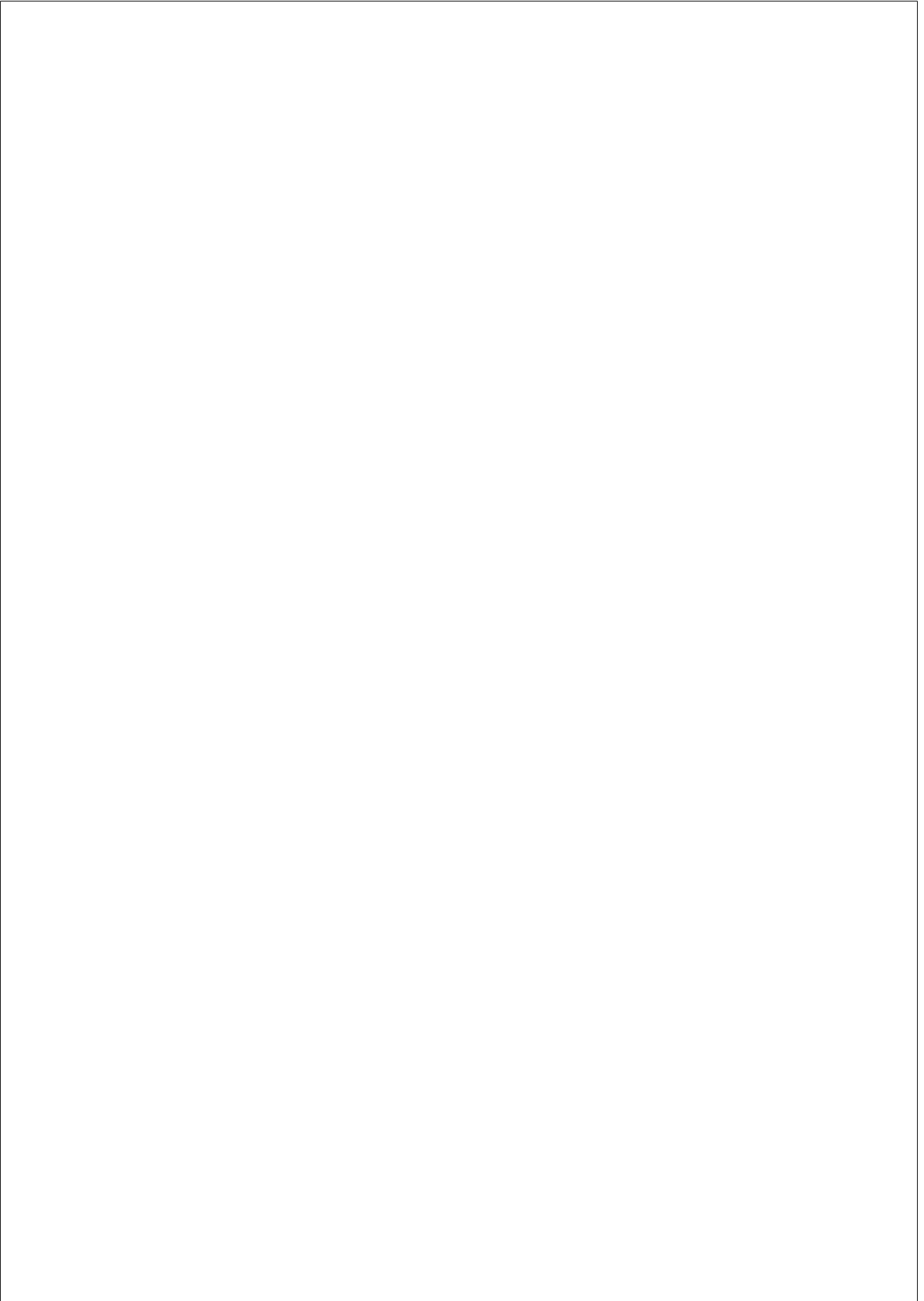
In the second publication, we test AdaptiveBandit in the complex simulation scenario of IDPs, where we reconstruct the coupled folding and binding of the IDP c-Myb with the KIX domain of the CREB protein. This system has proven to be challenging over the course of the thesis, as standard adaptive sampling algorithms were failing to sample the whole process. Experimental values for binding and kinetics are used to validate the results obtained.

2.2 Build a deep learning framework for molecular dynamics simulations

Machine learning potentials are one of the most promising applications of modern machine learning methods for molecular simulations. These potentials are a great example of using simulations as a data-generation tool, from where knowledge is extracted to construct reliable and fast models, able to generalize. Even though there has been a lot of research on these potentials, and several different neural network architectures have been developed, there is still a need for a unified computational

framework to combine these machine learning potentials with MD simulations.

In the third publication, we present TorchMD, a deep learning framework for molecular dynamics simulations. At its core, TorchMD is an MD simulations code entirely written in PyTorch, a popular machine learning Python library. Our objective is to create an MD code that can be easily combined with trained models, acting as external force-fields, and to provide an accessible framework from where to model, train and validate machine learned potentials, as well as performing end-to-end differentiable simulations.



Chapter 3

PUBLICATIONS

3.1 AdaptiveBandit: A multi-armed bandit framework for adaptive sampling in molecular simulations

Pérez A, Herrera-Nieto P, Doerr S, De Fabritiis G. [AdaptiveBandit: A multi-armed bandit framework for adaptive sampling in molecular simulations](#). *Journal of Chemical Theory and Computation*. 2020;16(7):4685-4693

Summary

In this paper, we frame adaptive sampling in terms of a multi-armed bandit problem and propose AdaptiveBandit, an algorithm that uses an action-value function and an upper confidence bound selection algorithm, improving adaptive sampling’s performance and increasing its versatility when faced against different free-energy landscapes. The algorithm is first tested against two different 2D toy models in order to showcase its capabilities. Finally, we compare it against state-of-the-art adaptive sampling methods in the exemplar case of protein folding, where Adaptive-

Bandit is able to sample the folding of villin while the other algorithms fail to do so.

AdaptiveBandit: A Multi-armed Bandit Framework for Adaptive Sampling in Molecular Simulations

Adrià Pérez,^{||} Pablo Herrera-Nieto,^{||} Stefan Doerr, and Gianni De Fabritiis*

Cite This: *J. Chem. Theory Comput.* 2020, 16, 4685–4693

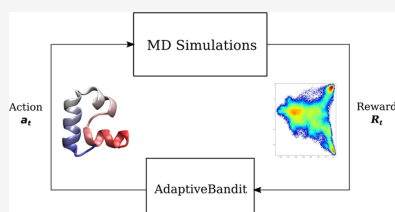
Read Online

ACCESS |

Metrics & More

Article Recommendations

ABSTRACT: Sampling from the equilibrium distribution has always been a major problem in molecular simulations due to the very high dimensionality of the conformational space. Over several decades, many approaches have been used to overcome the problem. In particular, we focus on unbiased simulation methods such as parallel and adaptive sampling. Here, we recast adaptive sampling schemes on the basis of multi-armed bandits and develop a novel adaptive sampling algorithm under this framework, AdaptiveBandit. We test it on multiple simplified potentials and in a protein folding scenario. We find that this framework performs similarly to or better than previous methods in every type of test potential. Furthermore, it provides a novel framework to develop new sampling algorithms with better asymptotic characteristics.



1. INTRODUCTION

In computational biology, macroscopic measurements by computer simulations are obtained by simulating microscopic molecular systems made of the order of a hundred thousand degrees of freedom. Statistical mechanics tells us the analytical form of the equilibrium distribution given the macroscopic constraint of the environment, e.g., constant temperature, pressure, and number of atoms. Therefore, the problem consists in generating samples from such a distribution.

Molecular simulation methods have always been hampered by sampling limitations over the equilibrium distribution due to their computational cost.^{1,2} The two main methods used to obtain samples are molecular dynamics (MD), a numerical scheme where the propagator of the dynamical system is discretized in time and iterated for billions of steps, and Monte Carlo sampling (MC), where the Monte Carlo rule is used to draw samples from the distribution. These sampling methods are also commonly used in other fields to sample for arbitrary probability distributions, and many of the methods developed for molecular simulations have been exploited in such contexts later, for instance, umbrella sampling,³ biased Monte Carlo methods,⁴ or biased molecular dynamics like replica exchange,^{5,6} steered MD,^{7,8} metadynamics,⁹ etc. Progress in molecular simulation sampling has therefore shown its relevance to a broader field of problems. Recently, a new generative method based on normalizing flows¹⁰ has been proposed to sample from the Boltzmann distribution.¹¹

Due to the difficulties in determining the bias a priori, practically equivalent to having a good prior, unbiased methods such as adaptive sampling^{12–15} have been recently developed and used successfully.^{16,17} Similarly, due to the difficulty in

generating good Monte Carlo moves, molecular dynamics is almost always preferred to Monte Carlo methods, largely due to the current efficiency of generating trajectories rooted in the capability of modern hardware. Specialized computer chips like Anton¹⁸ have made it possible to run long simulations of the order of hundreds of microseconds, sampling reversibly fast processes and exploring longer timescales.¹⁹ The advent of GPUs and GPU molecular dynamics software^{20–23} was a notable improvement, greatly increasing the computational efficiency of simulations. This, combined with Markov state models (MSMs),^{24,25} allowed us to reconstruct a complete statistical description of the full dynamical system from many shorter trajectories, obtaining a description that is equivalent to reversible sampling, once at convergence.

Running not one but hundreds or thousands of simulation trajectories^{26,27} created a new opportunity to decide the starting conditions of these simulations to obtain the best equilibrium characterization at the minimal computational cost, i.e., adaptive sampling. Initially, adaptive sampling algorithms^{12,15} were used to reduce the statistical uncertainty by choosing conformations that contributed the most to the error in the mean first passage time of an MSM,¹² eigenvalues, and eigenvectors,¹³ or by choosing low state populations.^{14,15}

Received: March 2, 2020

Published: June 15, 2020



MSMs were also used to detect those conformations that were kinetically farthest to the starting point to the increase sampling of high-barrier transitions.²⁸ Furthermore, similar algorithms appeared recently, which introduced prior knowledge to the selection criteria,^{29–31} seeking to further speed up the sampling toward equilibrium. One notable example is where contact information is used for protein folding³² or bound state contacts in protein–ligand or protein–protein binding.¹⁷ Other applications have used alternative geometric features, such as RMSD or pocket volume, to improve conformational exploration³³ and to find cryptic pockets.³⁴ It is also worth mentioning the weighted-ensemble algorithms, which distribute sampling across regions in a collective variable.^{35–37} In general, the adaptive sampling policy was always empirical, not based on any mathematical decision process, even though similarities have been recognized with the multi-armed bandit problem^{31,38} and reinforcement learning³⁹ before.

Here, we frame adaptive sampling in terms of a multi-armed bandit problem and propose AdaptiveBandit, an algorithm that uses an action-value function and an upper confidence bound^{40,41} selection algorithm, improving adaptive sampling’s performance and increasing its versatility when faced against different free-energy landscapes. Our main goal is to provide strong fundamentals when facing the exploration–exploitation dilemma by redefining it in terms of reinforcement learning, creating a solid framework from where to easily develop novel algorithms. AdaptiveBandit is available in HTMD (<https://github.com/Acellera/htmd>).⁴²

2. METHODS

2.1. MD Simulations. The configurational space of a molecular system for MD simulations is given by $\chi = \{x = (\mathbf{r}_1, \dots, \mathbf{r}_N) \in \mathbb{R}^{3N}\}$, where N is the number of atoms of the system. Experimental observables O are measured as equilibrium expectations $\langle O \rangle = \int O(x)\mu(x)dx$, where $\mu(x)$ is the equilibrium distribution. The form of this distribution is known; for instance, the Boltzmann distribution in the canonical ensemble at temperature T is

$$\mu(x) = \frac{1}{Z} e^{-U(x)/k_B T} \quad (1)$$

where $U(x)$ is the molecular potential energy, $k_B T$ is the Boltzmann constant multiplied by the temperature, and Z is the normalization factor. MD numerically solves Newton’s equation over the potential $U(x)$ for the variable x , plus a Langevin stochastic term accounting for thermal fluctuations.⁴³ Now consider the state $x(t) \in \chi$ as a specific conformation inside the configurational space χ at time t ; the probability of finding the molecule in configuration $x_{t+\tau}$ at a later time can be expressed by the conditional transition density function $p_{\tau}(x_{t+\tau}|x_t) \sim p_{\tau}(x_{t+\tau}, |x_t)$, which describes the probability of finding state $x_{t+\tau}$ given state x_t at time t after a time increment τ . When performing an MD simulation, the dynamics of the molecular system propagates the state x_t across time. Therefore, MD samples from the transition density p_{τ} given discrete time steps τ to obtain the next state $x_{t+\tau}$. This process is repeated for many steps, generating a trajectory of conformations.

The main goal of performing MD simulations is to obtain a good representation of the system’s equilibrium distribution $\mu(x)$, i.e., the probability to find conformation x under equilibrium conditions, to measure the average of observable

$\langle O \rangle$. If an MD trajectory τ is long enough, sampling from p_{τ} is equivalent to sampling from $\mu(x)$ (eq 1)

$$\lim_{\tau \rightarrow \infty} p_{\tau}(x_{t+\tau}|x_t) = \mu(x) \quad (2)$$

Generating long enough trajectories is computationally expensive, and often practically impossible when trying to sample slow events. However, long trajectories can be substituted by short parallel trajectories. While in principle one could model directly the conditional probability in eq 2, in practice this is not possible due to the very high dimensional space. Fortunately, it can be shown that the dynamics can be separated into a slow and fast set of variables,²⁴ and because the contributions of fast variables decay exponentially in τ , a reliable MSM can be constructed in terms of the slow variables to compute the thermodynamic averages. Usually, time-independent component analysis (tICA)⁴⁴ and clustering methods are used to study this set of variables during sampling, which is necessary to build the MSM. Once we obtain the MSM, computed by estimating the transition probabilities from discrete conformational states, one can derive the thermodynamic and kinetic properties, just assuming local, not global, equilibrium (i.e., τ is much shorter than what is necessary to satisfy eq 2).

2.2. Multi-armed Bandit Problem. The multi-armed bandit problem is a simplified reinforcement learning setting where one faces the exploration versus exploitation dilemma. The problem is defined as a tuple $\langle \mathcal{A}, \mathcal{R}, \gamma \rangle$, where \mathcal{A} is a set of k actions $\mathcal{A} = \{a_1, a_2, \dots, a_k\}$ and \mathcal{R} is an unknown probability distribution $\mathcal{R}^a = \mathbb{P}[r|a]$ of rewards given the chosen action. We choose $\gamma = 0$ for totally discounted rewards. At each time step t , the agent applies a policy $\pi_t = \mathbb{P}[a]$ to select an action $a_t \in \mathcal{A}$, based on previous actions taken and the respectively obtained rewards. Subsequently, the environment returns a reward $r_t \sim \mathcal{R}^{a_t}$. Given that we set $\gamma = 0$, we define the value of an action $Q_{\pi}(a)$ as its instantaneous mean reward

$$Q_{\pi}(a) = \mathbb{E}_{\pi}[r|a] \quad (3)$$

The goal is to find the optimal policy π^* that maximizes the cumulative reward $\sum_{t=1}^T r_t$. Policies must take into account the exploration versus exploitation dilemma and combine both explorative actions, to sample their associated unknown reward function to update their value estimates, and greedy actions, to increase the total cumulative reward by choosing the action with the highest value estimate. The main advantage of describing adaptive sampling in terms of a multi-armed bandit is that we can benefit from the extensive literature on bandits to find solutions and replace heuristic policies with more mathematically sound ones.

2.3. AdaptiveBandit. Standard adaptive sampling algorithms work by performing several rounds or epochs of short parallel simulations. In each round, the algorithm is faced with the decision to select any of the sampled conformations from where to respawn a new round of simulations. The objective of these decisions is to avoid any redundant sampling and optimize our simulations to obtain the desired goal (which can be anything from a full equilibrium characterization of a molecular system to sampling a specific conformation or dynamic event) at the minimum computational cost.

Here, we recast adaptive sampling in bandit terms, defining its tuple $\langle \mathcal{A}, \mathcal{R}, \gamma \rangle$. We define the action space \mathcal{A} in terms of

all possible conformations that are resolvable, i.e., they have been visited at least once

$$\mathcal{A} = \mathcal{H}_m = \{x_k \in \mathbb{R}^{3N}, k = 1, \dots, K_m\} \quad (4)$$

where K_m is the number of sampled configurations at epoch m .

There are different possible choices for the a priori unknown reward function \mathcal{R} that the policy will try to maximize, and it will mostly depend on your objective with the simulation experiment.

Because most of our MD experiments are usually aimed at sampling metastable states of interest, e.g., folded states of proteins or bound states between proteins and ligands, we have defined the reward \mathcal{R} to be proportional to minus the free energy so that the optimal policy always picks conformations from the low-free-energy, stable states. Therefore, we define the reward \mathcal{R}_a of action a as the mean of the minus free energies of each configuration x visited in the trajectory started with action a , i.e.,

$$\mathcal{R}_a = \langle -k_B T \log(\mu(x)) \rangle_{(a, x_1, \dots, x_t)} \quad (5)$$

where $\mu(x)$ is the equilibrium distribution over a partition of the configuration space containing the conformation x , e.g., MSM microstates. The average is computed over the succeeding frames in the trajectory starting from a .

The action space would be too large to compute meaningful value estimations for each conformation, and there is no way to know the exact equilibrium distribution. To address this issue, we take advantage of MSM analysis to redefine the tuple $\langle \mathcal{A}, \mathcal{R}, \gamma \rangle$ in a more practical form. We define a reduced and tractable action space using the MSM's discretized conformational space and use the stationary distribution of each state to obtain an estimate of their free energy to compute the rewards. We count each trajectory frame as an action taken, and use the succeeding frames to assign the reward. Because rewards strongly depend on how accurate the MSM estimation is, we use the latest MSM to recompute all past rewards from all trajectories at each epoch, differently from the common Q-learning approaches.⁴³ Not only does it ensure the best free energy estimation possible but it also addresses the increasing action space problem due to new conformations being sampled. For every epoch, the discretized conformational space is redefined, all frames are reassigned, and rewards are recomputed on the newly defined states. Resampling conformations for the next epoch are picked randomly from the selected state.

2.4. Solving the Multi-armed Bandit Problem. With the bandit tuple defined, we now need to deal with the exploration–exploitation trade-off and optimally solve it. To do so, AdaptiveBandit relies on the UCB1 algorithm⁴¹ to optimize the action-picking policy, which defines the upper confidence bound for action values based on the number of times the agent has picked that action and the total number of actions taken. Therefore, actions are selected based on

$$a_t = \operatorname{argmax}_{a \in \mathcal{A}} \left[Q_t(a) + c \sqrt{\frac{\ln t}{N_t(a)}} \right] \quad (6)$$

where t denotes the total number of actions taken, $Q_t(a)$ is the estimated action-value for action a , $N_t(a)$ is the number of times action a has been selected (prior to time t), and c is a parameter controlling the degree of exploration. UCB1 follows the principle of “optimism in the face of uncertainty”,

prioritizing actions with uncertain value estimations, even if those values are not the greatest. To select an action, UCB1 takes into account not only the estimated value of that action but also the amount of uncertainty on such value. By doing so, the algorithm not only promotes action exploration but also prioritizes the exploration of the most promising ones. In the long term, as our knowledge of action-values increases, the exploration term will decrease, and more greedy actions will be selected. UCB1 has a theoretical bound of $O(\sqrt{kt \log(L_t)})$ on its total regret L_t .⁴¹

2.5. AdaptiveBandit with Knowledge-Based Initialization. AdaptiveBandit also has the option to initialize action-value estimates with external knowledge from the system, providing an initial value estimation to new actions, aiding to prioritize the most valuable actions. While in previous methods^{17,31} this is done by forcing the algorithm to sample from conformations based on a fixed empirical ranking, here we use the bandit formalism to initialize Q in eq 6 with an empirical action-value function. This notably allows for the MSM to correct the initial prior suggestion for Q given enough sampling. This is not true in previous schemes, where a partially wrong prior can affect sampling to the point of nonconvergence to the intended results due to its degeneracy, i.e., even just some wrong contact information could kinetically bias the simulations far from the folding funnel. We demonstrate this aspect in the Results section. The initial prior $Q_{\text{prior}}(a)$ is defined as the maximum score in a state, and it is recalculated at each epoch, after reclustering. $Q_{\text{prior}}(a)$ is used to initialize $Q(a)$, converted from a scoring function to free energy by converting the score into a probability, given a maximum Q . Therefore, AdaptiveBandit will already start with non-zero $Q(a)$ and will act greedily on the scoring function preferred actions. States are assigned an initial pseudocount $N_0(a)$, representing the statistical certainty of $Q_{\text{prior}}(a)$. When selecting the next action a_t , $Q_{\text{prior}}(a)$ is averaged with the other rewards obtained with action a , and the initial pseudocount $N_0(a)$ is added to the action count $N_t(a)$.

2.6. Other Adaptive Sampling Algorithms. To evaluate AdaptiveBandit's performance, we have tested it against several different adaptive sampling strategies, mainly the standard low-counts adaptive sampling, FAST,³¹ and Exploration–Exploitation.

The low-counts adaptive sampling is a simple and intuitive strategy that is optimal in pure exploration scenarios.⁴² The method works by selecting conformations from the least populated clusters at each adaptive epoch. The other two methods, FAST and Exploration–Exploitation, are goal-oriented, where external knowledge on the system is used to guide sampling.

FAST is also inspired by the multi-armed bandit problem, but the implementation differs as it uses an acquisition function to rank discrete conformational states rather than a reward function by definition, and actions (and their outcomes) do not influence their value estimates. The acquisition function contains an exploitation term, defined by the goal-scoring function that assigns a fixed value to each state, and an exploration term, based on state counts. The FAST implementation we used works as

$$\rho_i = (1 - \alpha)\phi_i + \alpha\psi_i \quad (7)$$

where ρ_i is the score for state i , ϕ is the exploitative value obtained from the goal function for state i , ψ is the exploration

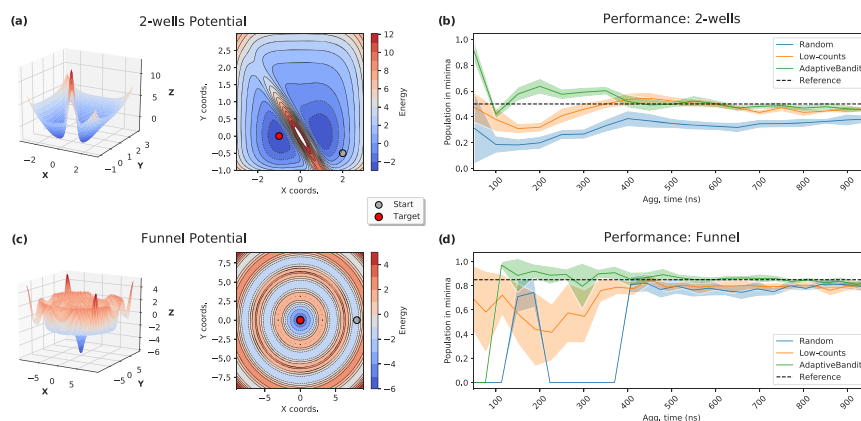


Figure 1. Performance comparison between random, low-counts, and AdaptiveBandit in the experiments with 2D potentials. (a, c) 3D view and top view of the 2-wells and funnel potentials. Global minima are located at $(-1, 0)$ and $(0, 0)$ coordinates, respectively. The gray dot indicates starting points for the simulations and the red dot indicates the target global minima where the population is measured at every epoch. (b, d) Performance comparison of the total aggregate simulation time needed for random, low-counts, and AdaptiveBandit sampling methods in the 2-wells and funnel potential, respectively, to achieve correct population estimates at their global minimum. The population is set to 0 when the targeted minima are not detected in half of the bootstrapped MSMs for that specific aggregate time.

value defined by state i counts (as in low-counts adaptive sampling), and α is a parameter regulating the weight of both terms. Both ϕ and ψ terms are scaled to values that range from 0 to 1. The states are defined as the microstates obtained by the constructed Markov model at each epoch.

Finally, we have Exploration–Exploitation, a strategy inspired by the popular method for multi-armed bandits ϵ -greedy, implemented in HTMD’s AdaptiveGoalEG.⁴² Simulations are restarted ϵ times from the top goal ranking states, and $1 - \epsilon$ times from the least sampled states (i.e., the low-counts strategy).

2.7. Langevin Dynamics on 2D Potentials. We designed a set of experiments in a simple simulation set up, performing Langevin dynamics on a single point mass of 1000 amu and a diffusion coefficient of $10 \text{ \AA}^2/\text{ns}$ at 300 K on two different potentials: a 2-wells potential (Figure 1a) inspired from ref 46, given by

$$U(x, y) = -3e^{-(x-1)^2-y^2} - 3e^{-(x+1)^2-y^2} + 15e^{-0.32(x^2+y^2+20(x+y)^2)} + 0.0512(x^4 + y^4) + 0.4e^{-2-4y} \quad (8)$$

and a funnel potential (Figure 1c) given by

$$U(x, y) = 2 \cos(2\sqrt{x^2 + y^2}) - 8e^{-(x^2+y^2)} + 0.2 \frac{((x/8)^2 + (y/8)^2)^3}{((x/8)^2 + (y/8)^2)^3} \quad (9)$$

A reference baseline for each 2D potential was calculated using an MSM built with 10 and 500 μs of aggregate simulation time for the 2-wells and funnel potential, respectively, spawning trajectories from conformations covering the whole surface. The equilibrium probability was determined to be 50 and 85%, respectively, on each of the global minima.

A total of $1 \mu\text{s}$ was simulated for each combination of method and potential, spawning 25 trajectories of 0.1 ns at each epoch for a total of 400 epochs. The performance at each epoch was measured as the mean of the equilibrium probabilities for the macrostate containing the targeted minimum for 10 independent MSMs built with 80% of the bootstrapped data. All of the MSM calculations were performed using HTMD.⁴²

For the goal methods, we simulated a total of $2 \mu\text{s}$ for each method, spawning 10 trajectories per epoch with trajectories of 0.05 ns. Values of $\alpha = 0.1$ for FAST and $\epsilon = 0.1$ for Exploration–Exploitation were selected. In AdaptiveBandit, the exploration rate was set to $c = 0.01$ and the initial pseudocounts to $N_0(a) = 50$. The c value was selected so that the scale of both $Q(a)$ values and the UCB part $\sqrt{\ln(t)/N_t(a)}$ is the same, based on estimates for Q_t related to the maximum free energy we would expect, and t_t related to the number of total samples we can afford for the experiment.

2.8. MD Simulation Setup. The simulation system for the chicken villin headpiece (PDB:2F4K) was built with HTMD.⁴² We solvated villin in a 64 \AA cubic box with a NaCl concentration of 0.05 M. The starting unfolded conformations for the runs were selected from a villin unfolding trajectory at a high temperature (500 K).

In this context, we tested AdaptiveBandit with $c = 0.01$ and $N_0(a) = 100$, against two different FAST setups, $\alpha = 0.5$ and 0.1. A goal-scoring function was used to guide the algorithms, based on the number of native $C\alpha$ contacts formed. For each setup, we ran parallel simulations of 10 ns, with 5–10 simulations per epoch, until we reached a total aggregate time of $4 \mu\text{s}$. All simulations were run with ACEMD,²² using the CHARMM22* force field⁷⁷ on a local GPU cluster. A short HTMD code listing is provided as an example to run AdaptiveBandit for villin simulations (Listing 1).

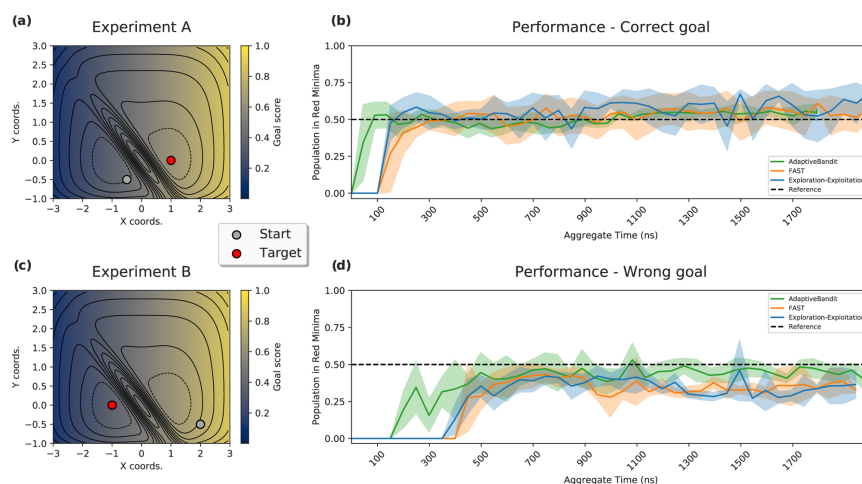


Figure 2. Performance comparison between goal-oriented algorithms FAST, Exploration-Exploitation, and AdaptiveBandit in the experiments with 2D potentials. (a, c) Top view of the 2-wells potential. Goal distribution across the potential is shown. Gray dots indicate the starting conformations for the runs. Red dots indicate the minima where the population is measured. (b, d) Performance comparison of total aggregate simulation time needed for FAST, Exploration-Exploitation, and AdaptiveBandit methods to correctly estimate the populations at their target minimum. The population is set to 0 when the targeted minima are not detected in half of the bootstrapped MSMs for that specific aggregate time.

```

from hmd.ui import *
from sklearn.cluster import MiniBatchKMeans
from jobqueue.localqueue import LocalGPUQueue
from goals import goalFunction

refmol = Molecule('villin_2f4k.pdb')
md = AdaptiveBandit()
md.app = LocalGPUQueue()
md.generatorspath = './generators'

md.clustmethod = MiniBatchKMeans
md.projection = MetricSelfDistance('protein_and_name_CA')
md.goalFunction = delayed(goalFunction)(refmol)
md.ticadim = 3
md.nmin = 5
md.nmax = 10
md.nframes = 1000000

md.exploration = 0.01 ## "c" value
md.goal_init = 100 ## prior initialization value

md.run()
    
```

3. RESULTS

3.1. Performance Testing on 2D Potentials. The initial objective is to compare the performance of a set of adaptive sampling algorithms in a simple environment defined by 2D potentials. For this purpose, we performed Langevin dynamics on two different potentials: the 2-wells potential, composed of two minima separated by a high energetic barrier (Figure 1a), and a funnel potential, composed of concentric isoenergetic regions with the global minimum located at its center (Figure 1c). The funnel potential is a useful benchmark to test the exploration-exploitation balance, as a purely exploratory strategy would tend to guide toward the outer circular wells, while the minimum is in the center. The objective of these experiments is to predict the equilibrium population of the targeted minima. The equilibrium populations are computed

with MSM analysis to assess how different sampling strategies affect the MSM estimation.

First, AdaptiveBandit was compared with two other common sampling policies, based on simple heuristics: random selection and the low-counts policy. The results for the 2-wells potential (Figure 1b) show a similar performance for the low-counts policy and AdaptiveBandit. Both converge at the baseline population (50%), while random sampling underestimates it. Because the potential just contains two large minima, comprising almost the entire conformational space, a fully explorative heuristic algorithm like the low-counts is optimal, as there is no need to prioritize anything besides exploring the two minima. AdaptiveBandit is able to reach the same optimal performance.

For the funnel potential (Figure 1d), the relative size of the minima is much smaller compared to the conformational space; hence, its detection by random sampling is more inefficient than for the other two algorithms. The low-counts method is able to reach the minima faster, as it is to cover the space quickly. Both these algorithms obtain a slight underestimation of the equilibrium population. On the other hand, AdaptiveBandit achieves a more accurate estimation and reaches convergence with 4 times less aggregate time than the other algorithms, highly reducing the computational resources needed to obtain accurate estimations of the equilibrium distribution.

This first test here showcases how introducing an exploitation term to quantify an action-value, besides the exploration term, either increases or equals the performance of fully exploratory algorithms on obtaining correct equilibrium estimations in the tested systems. Value estimations of each action help in prioritizing sampling on the most relevant areas

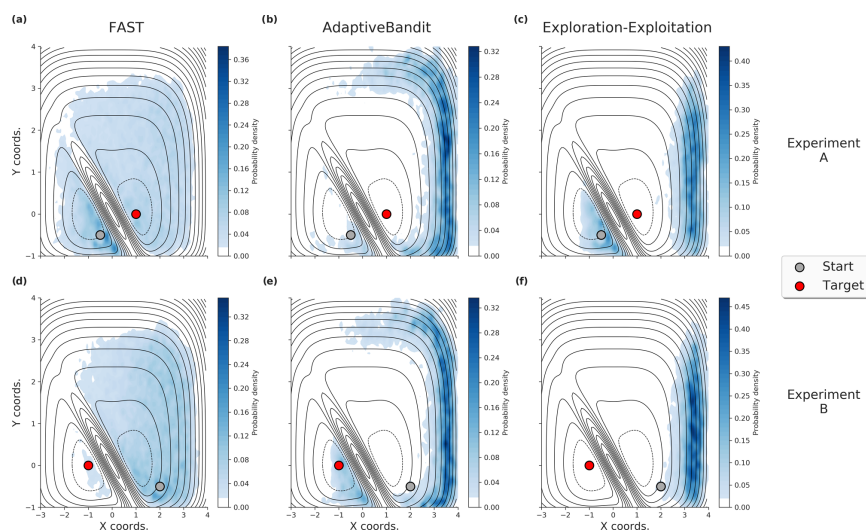


Figure 3. Simulation respawning distribution by algorithm across the 2-wells potential. Each plot depicts the probability distribution of selected conformations throughout the runs, obtained by the kernel density estimate.⁴⁸ The starting points for each run are represented with a gray dot and the target minimum with a red one. The goal distribution (not shown) is the same as in Figure 2. Subplots (a–c) represent the spawning probability distribution across the potential surface on experiment A, target minima at coordinates (1, 0), for FAST, AdaptiveBandit, and Exploration–Exploitation algorithms, and (d–f) for experiment B, target minima at coordinates (0, –0.5).

of the conformational space, rather than just exploring everything and sampling irrelevant conformations. Although in the 2-wells potential this does not make a big difference, it does in the funnel potential, where AdaptiveBandit focuses sampling on the minima by identifying its relevance with action-value estimates and does not waste resources on exploring irrelevant conformations.

3.2. Using System External Knowledge. Next, we want to test how AdaptiveBandit performs in the 2-wells potential against two existing methods that incorporate an exploitative term by employing external knowledge on the system. The pair of tested algorithms, also known as goal-oriented methods, are FAST³¹ and Exploration–Exploitation. To make sure AdaptiveBandit is at the same level of system knowledge as the other methods, the information provided by the goal function was used in AdaptiveBandit through knowledge-based initialization (as explained in Methods).

The goal function employed in the experiments with the 2-wells potential increases the score linearly with the *x*-axis (Figure 2a,c), thus creating a gradient of reward pushing to the right boundary of the potential. Two tests were performed in different scenarios. In the first test, the target minimum has a greater score than the starting coordinate (Experiment A, Figure 2a). In the second one, the target minimum has a lower goal than the starting conformations and, therefore, requires opposition to the goal’s influence to obtain accurate estimations on the target minimum (Experiment B, Figure 2c).

For experiment A, all methods reached the reference population, with AdaptiveBandit needing slightly less simulation time to reach the correct population estimation in the

target (Figure 2b). Differences in the algorithms can be visualized by a distribution plot of the spawning conformations in Figure 3. During the initial epochs, both FAST and Exploration–Exploitation follow the goal, spawning new simulations pushing against the energy barrier. AdaptiveBandit, on the other hand, quickly discovers the target minima and starts exploring other areas and directs sampling not only in the high score region but also in its surroundings. Even though the performance of all three algorithms is similar, differences in the spawning patterns between the three algorithms can be appreciated throughout the experiment. FAST presents a more explorative behavior and respawns simulations from all along the conformational space (Figure 3a). On the other hand, Exploration–Exploitation presents a highly exploitative behavior, strongly focusing on the highest goal-scoring region once it is discovered (Figure 3c). In between, AdaptiveBandit presents an overall greedy behavior, but with higher levels of exploration than the Exploration–Exploitation method, which translates into a small boost in its performance. It is interesting to point out the few resources invested by AdaptiveBandit in the origin minima, which demonstrates that the algorithm quickly identifies it as a noninteresting area (Figure 3b).

For experiment B, AdaptiveBandit reaches the target minimum faster, and equilibrium populations are estimated more accurately (Figure 2d). Both Exploration–Exploitation and FAST require more simulation time to reach the target minima and fail to converge on the correct equilibrium populations. In this scenario, Exploration–Exploitation is greatly focused in the high scoring region (Figure 3f), resulting

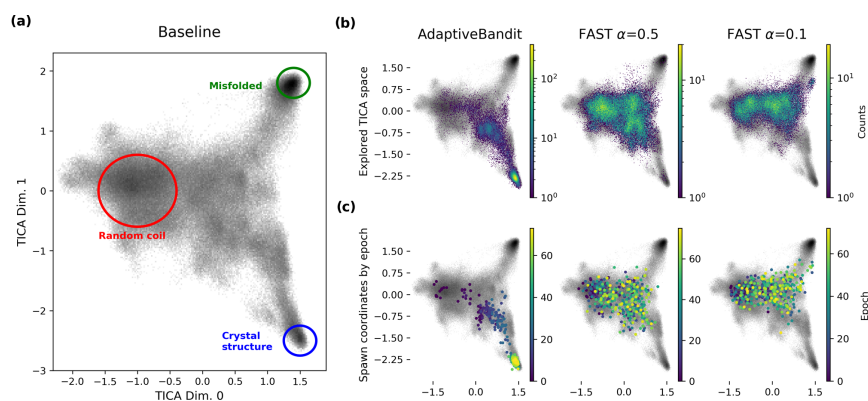


Figure 4. Villin folding simulations. (a) Conformational space for the folding of villin on the baseline data set. The tICA space includes large regions of random coil (the initial conformations are located within the red circle), misfolded conformations (green circle), and crystal-like structures (blue circle). (b) Exploration of the conformational space by sampling algorithms. Each plot includes the baseline exploration depicted on gray and the explored space with a colored heatmap. (c) Spawning coordinates for new epochs. Scattered points indicate starting conformations for new epochs, colored from first (purple) to last (yellow).

in a marginal exploration of the target minimum, while FAST and specially AdaptiveBandit do perform a more significant search on it (Figure 3d,e). Comparison between AdaptiveBandit and FAST spawning patterns (Figure 3d,e) reveals the differences in the exploration profile, where again FAST thoroughly spawns conformations from every explored point in the surface, while AdaptiveBandit, following the goal, explores the boundaries of the conformational space. Even if differences in performance are not substantially large, the experiment shows us the inability of FAST and Exploration–Exploitation to update the initial action-value estimates, translating into a lack of adaptation to the system being sampled. In opposition, AdaptiveBandit is able to correct the prior action-value estimates and readjust the sampling policy to a more optimal one, as it uses exploitation intrinsically based on MSM estimations from the available simulation data and external knowledge is introduced as prior information, rather than as the function to optimize. The ability to update the system knowledge at each epoch is crucial in experiments where the goal-scoring function used has high levels of degeneracy or is directly wrong. Asymptotically, AdaptiveBandit should always be better as it is logarithmically bound on the number of trials to the total regret⁴¹ (the difference between the maximum possible reward and the current reward), whereas Exploration–Exploitation and FAST are linearly bound.

3.3. Testing on Protein Folding Simulations. Besides testing in simple 2D potentials, we explored AdaptiveBandit’s performance in a more realistic and challenging scenario. AdaptiveBandit was tested on protein folding simulations, using villin as a benchmark. The chicken villin headpiece consists of a chain of 35 residues that folds into a three α -helical bundle, sharing a common hydrophobic core.⁴⁹ It is known to have a fast-folding rate of $(0.7 \mu\text{s})^{-1}$.⁴⁹ Our target for this test is to reach the folded state with the minimum amount of aggregate time and compare how AdaptiveBandit and FAST distribute sampling across the conformational space of villin. Because we are testing the algorithm’s effect rather than the

technical capabilities of reaching villin’s folding state with MD, we set up very short simulation times to increase the number of epochs and ensure we are evaluating the algorithm’s performance. The goal function used for the algorithms maximizes the number of native $C\alpha$ contacts formed to guide sampling on to the folded state. Thirty microseconds of villin folding simulations was used to build some reference tICA dimensions to evaluate the sampled conformational space from each method. The first two tICA dimensions reveal three main states (Figure 4a): the unfolded state (random coil), the folded structure, and a misfolded state.

Figure 4b shows the distinctive behavior of AdaptiveBandit and FAST while sampling the folding path. AdaptiveBandit clearly reaches the crystal structure. FAST struggles to do so due to the very short trajectories used, which produces a sampling bias, as indicated in ref 50. The results showcase how AdaptiveBandit is able to select the most relevant conformations to reach the folded state, prioritizing the most promising actions from the subset of undersampled actions. On the contrary, FAST, even in its most greedy setting ($\alpha = 0.1$), is not able to correctly prioritize the most relevant states and keeps exploring over random coil states, even in the latest epochs (Figure 4c). The greedy setting also presents a slight misdirection toward the misfolded state, which suggests that the used goal-scoring function has degeneracy, and it does not differentiate enough between native-like structures and misfolded structures that are very far dynamically. As commented in the previous experiment using external knowledge on the 2-wells potential, methods like FAST or Exploration–Exploitation that rely only on external information can be severely hampered when the provided information does not represent the true energetic gradient. AdaptiveBandit prevents this by updating the prior information with rewards coming from interacting with the system and observing its response to our actions.

4. CONCLUSIONS

AdaptiveBandit formally introduces adaptive sampling into reinforcement learning by describing it in terms of multi-armed bandits and builds upon it to deliver a novel algorithm with increased performance and flexibility across different energy landscapes. AdaptiveBandit is able to perform equally or better than previous adaptive sampling algorithms in a diverse set of systems, and it has demonstrated the ability to learn from simulation results. AdaptiveBandit works both with and without external knowledge of the system, and it can update prior beliefs in the system based on the results obtained during the experiment.

Goal-oriented adaptive sampling methods, as in ref 31, also get inspiration from exploration–exploitation strategies, like ϵ -greedy. The context, however, is quite different as there is no definition of a multi-armed bandit framework and a reward per action; rather it is more akin to directly defining an acquisition function. Furthermore, the greediness is toward predetermined states given from external knowledge on the system. AdaptiveBandit, as used here, uses exploitation intrinsically without requiring external information. It is, however, a possibility to do so and use the experimental data to provide a prior for the sampling.

We have exemplified here cases where AdaptiveBandit works better due to its adaptability and flexibility, but this does not mean that it could underperform in other scenarios. Our implementation of AdaptiveBandit relies on good MSM estimates and, therefore, the action-value estimates carry on with errors caused not only by discretization and dimensionality reduction but also by the sampling bias, especially on estimations of equilibrium populations.⁵⁰ Additionally, AdaptiveBandit's performance also depends on the c hyperparameter to regulate exploration and it is not very intuitive, as it must be tuned according to the scale of both terms in eq 6.

The version of AdaptiveBandit presented here defines a reward proportional to the free energy of each state and utilizes the UCB1 algorithm to optimize the action-picking policy. However, this is not the only possible way to apply AdaptiveBandit, and the algorithm can be changed to better adapt to the experiment and systems. We hope that our work inspires the development of new adaptive sampling algorithms built under theoretical fundamentals instead of using simple heuristic policies.

AUTHOR INFORMATION

Corresponding Author

Gianni De Fabritiis — Computational Science Laboratory, Universitat Pompeu Fabra, 08003 Barcelona, Spain; *Acellera Labs, 08005 Barcelona, Spain; Institutió Catalana de Recerca i Estudis Avançats, 08010 Barcelona, Spain*; orcid.org/0000-0003-3913-4877; Email: gianni.defabritiis@upf.edu

Authors

Adrià Pérez — Computational Science Laboratory, Universitat Pompeu Fabra, 08003 Barcelona, Spain; orcid.org/0000-0003-2637-1179

Pablo Herrera-Nieto — Computational Science Laboratory, Universitat Pompeu Fabra, 08003 Barcelona, Spain

Stefan Doerr — Computational Science Laboratory, Universitat Pompeu Fabra, 08003 Barcelona, Spain; *Acellera Labs, 08005 Barcelona, Spain*; orcid.org/0000-0002-8678-8657

Complete contact information is available at:

<https://pubs.acs.org/10.1021/acs.jctc.0c00205>

Author Contributions

^{||}A.P. and P.H.-N. contributed equally to this work.

Notes

The authors declare no competing financial interest.

ACKNOWLEDGMENTS

G.D.F. acknowledges support from MINECO (Unidad de Excelencia María de Maeztu, funded by the AEI (CEX2018-000792-M) and BIO2017-82628-P), FEDER, and Secretaria d'Universitats i Recerca de la Generalitat de Catalunya. This project received funding from the European Union's Horizon 2020 Research and Innovation Program under Grant Agreement No. 823712 (CompBioMed2).

REFERENCES

- (1) Martínez-Rosell, G.; Giorgino, T.; Harvey, M. J.; de Fabritiis, G. Drug discovery and molecular dynamics: methods, applications and perspective beyond the second timescale. *Curr. Top. Med. Chem.* **2017**, *17*, 2617–2625.
- (2) Pérez, A.; Martínez-Rosell, G.; De Fabritiis, G. Simulations meet machine learning in structural biology. *Curr. Opin. Struct. Biol.* **2018**, *49*, 139–144.
- (3) Torrie, G. M.; Valleau, J. P. Nonphysical sampling distributions in Monte Carlo free-energy estimation: Umbrella sampling. *J. Comput. Phys.* **1977**, *23*, 187–199.
- (4) Frenkel, D.; Smit, B., Eds.; *Understanding Molecular Simulation: From Algorithms to Applications*, 1st ed.; Academic Press, Inc.: Orlando, FL, 1996.
- (5) Sugita, Y.; Okamoto, Y. Replica-exchange molecular dynamics method for protein folding. *Chem. Phys. Lett.* **1999**, *314*, 141–151.
- (6) Fukunishi, H.; Watanabe, O.; Takada, S. On the Hamiltonian replica exchange method for efficient sampling of biomolecular systems: Application to protein structure prediction. *J. Chem. Phys.* **2002**, *116*, 9058–9067.
- (7) Izrailev, S.; Stepaniants, S.; Isralewitz, B.; Kosztin, D.; Lu, H.; Molnar, F.; Wriggers, W.; Schulten, K. *Computational Molecular Dynamics: Challenges, Methods, Ideas*; Springer, 1999; pp 39–65.
- (8) Isralewitz, B.; Gao, M.; Schulten, K. Steered molecular dynamics and mechanical functions of proteins. *Curr. Opin. Struct. Biol.* **2001**, *11*, 224–230.
- (9) Laio, A.; Parrinello, M. Escaping free-energy minima. *Proc. Natl. Acad. Sci. U.S.A.* **2002**, *99*, 12562–12566.
- (10) Rezende, D. J.; Mohamed, S. Variational inference with normalizing flows. arXiv:1505.05770. arXiv.org e-Print archive. <https://arxiv.org/abs/1505.05770> (accessed May 21, 2015).
- (11) Noé, F.; Olsson, S.; Köhler, J.; Wu, H. Boltzmann generators: Sampling equilibrium states of many-body systems with deep learning. *Science* **2019**, *365*, No. eaaw1147.
- (12) Singhal, N.; Pande, V. S. Error analysis and efficient sampling in Markovian state models for molecular dynamics. *J. Chem. Phys.* **2005**, *123*, No. 204909.
- (13) Hinrichs, N. S.; Pande, V. S. Calculation of the distribution of eigenvalues and eigenvectors in Markovian state models for molecular dynamics. *J. Chem. Phys.* **2007**, *126*, No. 244101.
- (14) Pronk, S.; Larsson, P.; Pouya, I.; Bowman, G. R.; Haque, I. S.; Beauchamp, K.; Hess, B.; Pande, V. S.; Kasson, P. M.; Lindahl, E. In *Copernicus: A New Paradigm for Parallel Adaptive Molecular Dynamics*, Proceedings of 2011 International Conference for High Performance Computing, Networking, Storage and Analysis, 2011; p 60.
- (15) Doerr, S.; De Fabritiis, G. On-the-fly learning and sampling of ligand binding by high-throughput molecular simulations. *J. Chem. Theory Comput.* **2014**, *10*, 2064–2069.
- (16) Noé, F.; Schütte, C.; Vanden-Eijnden, E.; Reich, L.; Weikl, T. R. Constructing the equilibrium ensemble of folding pathways from

short off-equilibrium simulations. *Proc. Natl. Acad. Sci. U.S.A.* **2009**, *106*, 19011–19016.

(17) Plattner, N.; Doerr, S.; De Fabritiis, G.; Noé, F. Complete protein-protein association kinetics in atomic detail revealed by molecular dynamics simulations and Markov modelling. *Nat. Chem.* **2017**, *9*, 1005.

(18) Shaw, D. E.; Deneroff, M. M.; Dror, R. O.; Kuskin, J. S.; Larson, R. H.; Salmon, J. K.; Young, C.; Batson, B.; Bowers, K. J.; Chao, J. C.; et al. Anton, a Special-Purpose Machine for Molecular Dynamics Simulation. *Commun. ACM* **2008**, *51*, 91–97.

(19) Lindorff-Larsen, K.; Piana, S.; Dror, R. O.; Shaw, D. E. How fast-folding proteins fold. *Science* **2011**, *334*, 517–520.

(20) Friedrichs, M. S.; Eastman, P.; Vaidyanathan, V.; Houston, M.; Legrand, S.; Beberg, A. L.; Ensign, D. L.; Bruns, C. M.; Pande, V. S. Accelerating molecular dynamic simulation on graphics processing units. *J. Comput. Chem.* **2009**, *30*, 864–872.

(21) Harvey, M. J.; De Fabritiis, G. An implementation of the smooth particle mesh Ewald method on GPU hardware. *J. Chem. Theory Comput.* **2009**, *5*, 2371–2377.

(22) Harvey, M. J.; Giupponi, G.; Fabritiis, G. D. ACEMD: accelerating biomolecular dynamics in the microsecond time scale. *J. Chem. Theory Comput.* **2009**, *5*, 1632–1639.

(23) 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*, No. e1005659.

(24) 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*, No. 174105.

(25) Bowman, G. R.; Pande, V. S.; Noé, F. *An Introduction to Markov State Models and Their Application to Long Timescale Molecular Simulation*; Springer Science & Business Media, 2013; Vol. 797.

(26) Buch, I.; Giorgino, T.; De Fabritiis, G. Complete reconstruction of an enzyme-inhibitor binding process by molecular dynamics simulations. *Proc. Natl. Acad. Sci. U.S.A.* **2011**, *108*, 10184–10189.

(27) Martínez-Rosell, G.; Harvey, M. J.; De Fabritiis, G. Molecular-simulation-driven fragment screening for the discovery of new CXCL12 inhibitors. *J. Chem. Inf. Model.* **2018**, *58*, 683–691.

(28) Zhou, T.; Caflisch, A. Free energy guided sampling. *J. Chem. Theory Comput.* **2012**, *8*, 2134–2140.

(29) Sabbadin, D.; Moro, S. Supervised molecular dynamics (SuMD) as a helpful tool to depict GPCR-ligand recognition pathway in a nanosecond time scale. *J. Chem. Inf. Model.* **2014**, *54*, 372–376.

(30) Perez, A.; MacCallum, J. L.; Dill, K. A. Accelerating molecular simulations of proteins using Bayesian inference on weak information. *Proc. Natl. Acad. Sci. U.S.A.* **2015**, *112*, 11846–11851.

(31) Zimmerman, M. I.; Bowman, G. R. FAST conformational searches by balancing exploration/exploitation trade-offs. *J. Chem. Theory Comput.* **2015**, *11*, 5747–5757.

(32) Ovchinnikov, S.; Park, H.; Varghese, N.; Huang, P.-S.; Pavlopoulos, G. A.; Kim, D. E.; Kamisetty, H.; Kyrpides, N. C.; Baker, D. Protein structure determination using metagenome sequence data. *Science* **2017**, *355*, 294–298.

(33) Zimmerman, M. I.; Hart, K. M.; Sibbald, C. A.; Frederick, T. E.; Jimah, J. R.; Knoverek, C. R.; Tolia, N. H.; Bowman, G. R. Prediction of new stabilizing mutations based on mechanistic insights from Markov state models. *ACS Cent. Sci.* **2017**, *3*, 1311–1321.

(34) Cruz, M. A.; Frederick, T. E.; Singh, S.; Vithani, N.; Zimmerman, M. I.; Porter, J. R.; Moeder, K. E.; Amarasinghe, G. K.; Bowman, G. R. Discovery of a cryptic allosteric site in Ebola's undruggable VP35 protein using simulations and experiments. *bioRxiv*, 2020.

(35) Huber, G. A.; Kim, S. Weighted-ensemble Brownian dynamics simulations for protein association reactions. *Biophys. J.* **1996**, *70*, 97–110.

(36) Dickson, A.; Brooks, C. L., III WExplore: hierarchical exploration of high-dimensional spaces using the weighted ensemble algorithm. *J. Phys. Chem. B* **2014**, *118*, 3532–3542.

(37) Donyapour, N.; Roussey, N. M.; Dickson, A. REVO: Resampling of ensembles by variation optimization. *J. Chem. Phys.* **2019**, *150*, No. 244112.

(38) Zimmerman, M. I.; Porter, J. R.; Sun, X.; Silva, R. R.; Bowman, G. R. Choice of adaptive sampling strategy impacts state discovery, transition probabilities, and the apparent mechanism of conformational changes. *J. Chem. Theory Comput.* **2018**, *14*, 5459–5475.

(39) 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.

(40) Lai, T. L.; Robbins, H. Asymptotically efficient adaptive allocation rules. *Adv. Appl. Math.* **1985**, *6*, 4–22.

(41) Auer, P. Using confidence bounds for exploitation-exploration trade-offs. *J. Mach. Learn. Res.* **2002**, *3*, 397–422.

(42) Doerr, S.; Harvey, M.; Noé, F.; De Fabritiis, G. HTMD: high-throughput molecular dynamics for molecular discovery. *J. Chem. Theory Comput.* **2016**, *12*, 1845–1852.

(43) Loncharich, R. J.; Brooks, B. R.; Pastor, R. W. Langevin dynamics of peptides: The frictional dependence of isomerization rates of N-acetylalanine-N'-methylamide. *Biopolymers* **1992**, *32*, 523–535.

(44) 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*, No. 015102.

(45) Sutton, R. S.; Barto, A. G. *Reinforcement Learning: An Introduction*; MIT Press, 2018.

(46) Pan, A. C.; Roux, B. Building Markov state models along pathways to determine free energies and rates of transitions. *J. Chem. Phys.* **2008**, *129*, No. 064107.

(47) Piana, S.; Lindorff-Larsen, K.; Shaw, D. E. How robust are protein folding simulations with respect to force field parameterization. *Biophys. J.* **2011**, *100*, L47–L49.

(48) Waskom, M. *seaborn.kdeplot*. <https://seaborn.pydata.org/generated/seaborn.kdeplot.html/> (accessed Jan 11, 2020).

(49) Kubelka, J.; Chiu, T. K.; Davies, D. R.; Eaton, W. A.; Hofrichter, J. Sub-microsecond protein folding. *J. Mol. Biol.* **2006**, *359*, 546–553.

(50) Wan, H.; Voelz, V. A. Adaptive Markov state model estimation using short reseeded trajectories. *J. Chem. Phys.* **2020**, *152*, No. 024103.

3.2 Binding-and-folding recognition of an intrinsically disordered protein using adaptive molecular dynamics

Herrera-Nieto P, Pérez A, De Fabritiis G. Binding-and-folding recognition of an intrinsically disordered protein using adaptive molecular dynamics. *Manuscript pending for submission*

Summary

In this paper we make use of AdaptiveBandit to simulate the coupled binding and folding of the disordered protein cMyb with the KIX domain. The newly developed sampling algorithm succeeds in recovering the entire event, where other sampling algorithms failed. We construct a Markov state model in order to understand how the binding and folding occurs, validating the model using experimental values. The model helps us to identify a mixed binding and folding process that combines conformational selection and induced fit mechanisms. Results bring new insights to disordered proteins and their mechanisms of function with atomic resolution.

Binding-and-folding recognition of an intrinsically disordered protein using adaptive molecular dynamics

Pablo Herrera-Nieto^{a,1}, Adrià Pérez^{a,1}, and Gianni De Fabritiis^{a,b,2}

^aComputational Science Laboratory, Barcelona biomedical research park (PRBB), Universitat Pompeu Fabra, Dr Aiguader 88, Barcelona 08003, Spain; ^bInstitució Catalana de Recerca i Estudis Avançats (ICREA), Passeig Lluís Companys 23, 08010 Barcelona, Spain

This manuscript was compiled on November 15, 2021

Intrinsically disordered proteins participate in many biological processes by folding upon binding with other proteins. However, coupled folding and binding processes are not well understood from an atomistic point of view. One of the main questions is whether folding occurs prior or after binding. Here we use a novel unbiased high-throughput adaptive sampling approach to reconstruct the binding and folding between the disordered transactivation domain of c-Myb and the KIX domain of the CREB-binding protein. The reconstructed long term dynamical process highlights the binding of a short stretch of amino acids on c-Myb as a folded α -helix. Leucine residues, specially Leu298 to Leu302, establish initial native contacts that prime the binding and folding of the rest of the peptide, with a mixture of conformational selection on the N-terminal region with an induced fit of the C-terminal.

conformational selection | coupled folding binding | intrinsically disordered proteins | molecular dynamics simulations | protein-protein interactions

Intrinsically disordered proteins (IDPs) participate in many biological functions despite lacking a stable tertiary structure (1). Initial clues for the function of IDPs were revealed by structural studies (2, 3), showing that proteins that were disordered in isolation became folded upon interacting with their partners, opening to question how folding couples with binding.

Recently, molecular dynamics (MD) simulations have been successfully applied to reconstruct biological dynamic events in problems such as protein-ligand (4) and protein-protein (5, 6) binding, as well as protein folding (7, 8). MD has also been applied in the field of IDPs (9–12). In particular, the Mdm2 protein and the disordered 12-residue N-terminal region of p53 were studied using implicit solvent simulations(9), parallel full-atom simulations totalling 831 μ s (11), biased free-energy-based sampling (10), and both biased and unbiased simulations in order to estimate kinetics on the second timescale (12). For another system, KIX-pKID, a single event of binding (13) has been sampled at all-atom resolution.

The KIX–c-Myb binding-and-folding mechanism has been extensively studied experimentally as an exemplar case of protein-IDP interaction (14–20). The KIX domain of the CREB-binding protein is a short 87-aa region composed of three α -helices (designated as α -1, α -2 and α -3, from N-terminal to C-terminal) forming a compact bundle (3). KIX represents a paradigm of binding promiscuity: it binds to many IDPs, including the proto-oncogene c-Myb (3) (Figure 1.a), with multiple binding conformations (14). However, the system composed by KIX–c-Myb remained outside of the scope of all-atom molecular simulations due to the extension

of the IDP (it doubles the lengths of p53) and the existence of multiple binding modes between them (14). In particular, it is unclear whether the interaction takes place by conformational selection, i.e. c-Myb needs to be folded before binding to its partner, or by induced-fit, where binding not only happens independently of c-Myb’s secondary structure but also triggers its folding, as shown for other IDPs (KIX-pKID) (13, 21). Understanding these aspects has implications on the drugability of disordered proteins. Another important factor is c-Myb’s high helicity in isolation and the consequences it might exert on the final complex structure, which features an extended α -helical c-Myb bound to KIX. Some reports support the induced-fit approach based on kinetics and mutagenesis studies (15, 16, 20), while others advocate for a mixed mechanism (14); yet not a detailed model for the binding process is available.

In this paper, we take advantage of a novel algorithm which frames the MD sampling problem from a reinforcement learning perspective (see Methods) to reconstruct multiple binding modes between c-Myb and KIX. This new sampling algorithm was key for us to reconstruct the binding process, as previous attempts over the years using other state-of-the-art adaptive sampling methods (22, 23) were not successful, always failing to recover the NMR bound structure. Results provide insights in the binding mechanism between these two proteins,

Significance Statement

Many intrinsically disordered proteins fold upon interacting with their protein partners. Molecular dynamic simulations have been extensively applied in the challenging task of recreating experimentally determined structures. Here, we take advantage of a novel adaptive sampling algorithm to reconstruct the binding and folding between the disordered transactivation domain of c-Myb and the KIX domain of the CREB-binding protein. We report the full reconstruction of binding of c-Myb to KIX and the various pathways underlying the process, and we determine whether it happens through a conformational selection or an induced-fit process. We believe that the use of molecular simulations and adaptive sampling methods can significantly impact the understanding of conformational dynamics of disordered proteins.

Author contributions: P.H.N. and A.P. generated and analyzed the data; P.H.N., A.P. and G.D.F. wrote the paper; G.D.F. designed research.

¹These authors contributed equally.
The authors declare no conflict of interest.

²To whom correspondence should be addressed. E-mail: gianni.defabritiis@upf.edu

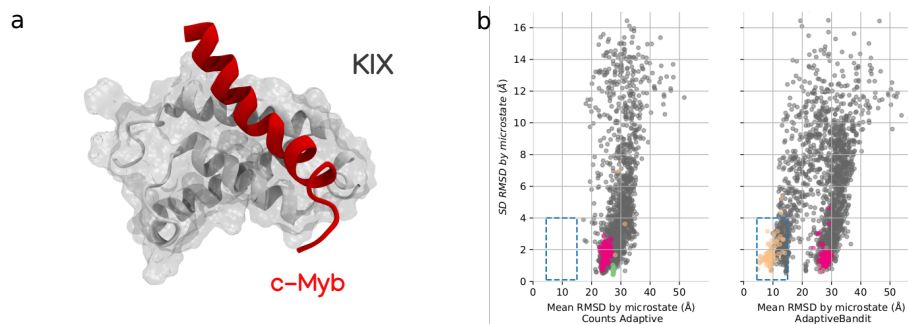


Fig. 1. Exploration performance. **a)** KIX—cMyb NMR structure. KIX domain is shown as a white surface and ribbon and c-Myb bound to KIX as a red helix (PDB code 1SB0). **Exploration performance by b)** Counts Adaptive ($\sim 480 \mu\text{s}$), and AdaptiveBandit ($\sim 225 \mu\text{s}$) is shown by plotting the mean RMSD (on the x axis) and standard deviation (on the y-axis) for each of the MSM’s microstates, color mapped accordingly to their microstate assignment. Dashed square indicates the *bound zone*, placed in the region corresponding to low mean RMSD and standard deviation.

56 supporting a mixed model that combines both conformational
57 selection and induced-fit.

58 Results

59 **Adaptive sampling the KIX—c-Myb binding-and-folding pro-**
60 **cess.** Simulations to reconstruct the KIX—c-Myb binding
61 mode were performed following an adaptive sampling strategy.
62 In adaptive sampling, successive rounds of simulations are per-
63 formed in an iterative step-wise manner, where an acquisition
64 function over the currently sampled conformation is defined.
65 Initially, we compare two acquisition functions: a count-based
66 one and another inspired by reinforcement learning.

67 The new AdaptiveBandit method (24) is framed into a
68 simplified reinforcement learning problem, the multi-armed
69 bandit problem (see *Materials & Methods*). We use the upper
70 confidence bound (UCB) algorithm (25) to optimize an action
71 picking policy in order to maximize future rewards, optimally
72 balancing the exploration of new higher rewarding actions
73 with the exploitation of the most known rewarding ones. The
74 reward function, which associates the action to the reward
75 given by the system, defines what we want to optimize. In
76 this work, we choose the reward to be minus the free energy
77 of each configuration visited in the trajectory spawn from a
78 given action (see Eq.2 in *Materials & Methods*), where the free
79 energy of a conformation is given by the corresponding MSM
80 microstate computed with the data available at the current
81 sampling epoch.

82 Standard low counts adaptive sampling (22) (called Counts
83 Adaptive) can be shown to be optimal in pure exploration
84 conditions (23). Counts are computed over clusters of con-
85 formations; this method is, however, noisy as clusters can be
86 poorly populated. Therefore, in the implementation available
87 in (23), counts are computed over a smaller subset by grouping
88 clusters (microstates) into macrostates, constructing a Markov
89 State Model (MSM) (26) with the available data at each round.
90 The acquisition function is given by proportionally choosing
91 macrostates as $1/c$, where c represents macrostate counts, and
92 by randomly selecting conformations within them.

93 A comparison between Counts Adaptive and AdaptiveBan-

dit is provided in Figure 1.b. The batch based on Counts
Adaptive (48 epochs) failed to connect microstates similar to
the NMR structure in over $\sim 480 \mu\text{s}$, reaching at best
an RMSD around 7 Å, indicating excessive exploration. For
us, it was impossible to build an MSM with the bound state
with previous methods, and novel approaches were needed to
reconstruct the binding-and-folding process between KIX
and c-Myb successfully. AdaptiveBandit provides converged
estimates of kinetics and thermodynamics after just $150 \mu\text{s}$
of sampling (Supplementary Figure 5).

94
95
96
97
98
99
100
101
102
103
Identification of the bound state. The full data set of the Adap-
104 tiveBandit run accounted for a total simulation time of ~ 450
105 μs , split across 40 epochs, and was the one used to study
106 the molecular features of KIX—c-Myb binding-and-folding.
107 An MSM was built based on all-pair $C_\alpha + C_\beta$ distances be-
108 tween KIX and c-Myb, self distances between C_α of c-Myb,
109 secondary structure of c-Myb and RMSD to the NMR bound
110 conformation (PDB ID: 1SB0). The MSM defines three kine-
111 tically similar sets of conformations, referred as macrostates
112 (Figure 2.a and Supplementary Figure 1.b): a highly popu-
113 lated state with an heterogeneous mixture of conformations
114 (*unbound*), a well defined c-Myb bound state (*bound*) and,
115 finally, a secondary bound state (*secondary*). Representative
116 structures of all states can be found in Figure 2.b. The *bound*
117 macrostate contains structures with a minimum RMSD of 3.00
118 Å with respect to the NMR structure.

119
120
121
122
123
124
125
126
127
128
129
130
131
132
The *bound* state identifies the primary cMyb bound pose
in the hydrophobic groove between α -1 and α -3) of KIX. On
average, it shares 36% of the fraction of native inter-molecular
contacts (Q_{int}) with the original NMR structure, as shown in
Figure 2.c. These contacts mainly involve the interaction of
c-Myb residues Leu298 and Leu302 with residues across the
primary binding interface: Leu302 contacts Leu603, Leu653,
and specially Leu607 of KIX, which is buried down in the
pocket, whereas Leu298 establishes additional native contacts
with Ala610, Ile657, and Tyr658. Q_{int} reaches up to 80% in
those microstates exhibiting the tightest bound conformations,
and, in addition to the leucine binding, they feature most of
the contacts between the C-terminal half of c-Myb and KIX,

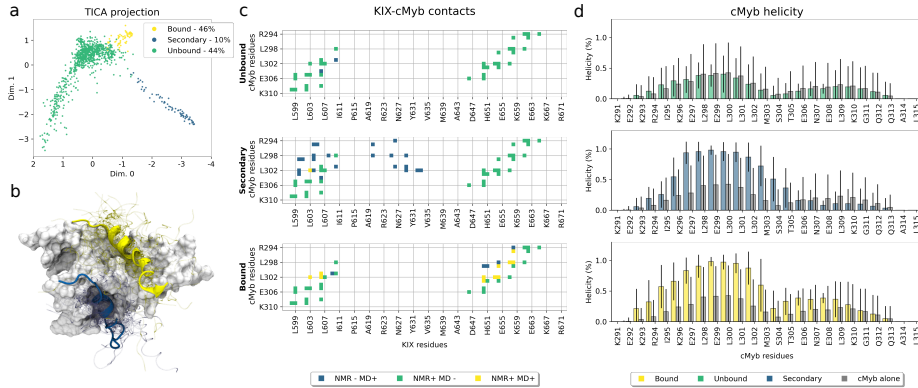


Fig. 2. KIX & c-Myb binding model. a) States distribution across the TICA space: microstates are represented as dots and are colored following their macrostate assignment. b) Representative structures: PDB structure 1SB0 is depicted with KIX as gray surface, c-Myb bound to the primary interface as a yellow ribbon, and c-Myb bound to the secondary interface as a blue ribbon. c-Myb backbones for 30 representatives MD structures of *bound* and *secondary* states are displayed with blurry yellow and blue clouds, respectively. c) Macrostate contact fingerprint: profile of contacts established between c-Myb and KIX in each macrostate in at least 50% of the structures. Blue color represents contacts present in the state but not in the original NMR structure; green indicates original NMR contacts not found in the MSM state; and yellow squares represent contact matches, found in both NMR and MD structures. d) Macrostate cMyb helicity: helicity fraction per residue of c-Myb in each macrostate. Helicity for the cMyb peptide alone is depicted in grey in each plot for comparison.

133 which are not that prevalent across the *bound* macrostate
 134 (Supplementary Figure 2). The main contacts missing account
 135 for the electrostatic interactions established between Arg294
 136 and the region on α -3. There are some conformations where
 137 these interactions occur, but their are prevalence in those
 138 microstates is less than 50%.

139 Secondary structure profile for MD derived states matches
 140 the experimental description of c-Myb (14, 19), as shown in
 141 Supplementary Figure 3: the 25 residues are separated in two
 142 halves by residues Met303 and Ser304. The N-terminal half
 143 shows a high helical tendency, around 20-30% for residues
 144 in positions 297 to 302 with c-Myb in isolation, being maximal
 145 in bound states. Experimentally, this N-terminal half
 146 in isolation reaches even higher helicity levels ($\sim 70\%$) when
 147 using an extended construct of c-Myb (14). On the other
 148 hand, the C-terminal section exhibits low helical propensity
 149 when in isolation, and increases when bound to KIX,. The
 150 full helix conformation only appears in those microstates with
 151 the tightest bound conformations.

152 **Secondary binding mode.** The existence of alternative binding
 153 poses between c-Myb and KIX has also been reported
 154 (14). The MSM shows the presence of a secondary binding
 155 mode (referred as *secondary*), occupying a novel interface,
 156 located between α -1 and α -2 (Figure 1.b and Supplementary
 157 Figure 9). The interaction of the *secondary* state resembles
 158 the *bound* binding mode: the N-terminal half is folded in the
 159 typical α -helix, while the C-terminal section remains mostly
 160 unstructured. The presence of a native contact in this secondary
 161 binding mode is due to the penetration of Leu302,
 162 locating close to Leu603's backbone in KIX, rather than by
 163 side-chain proximity. Leu298 and Leu302 of c-Myb are deeply
 164 buried in an hydrophobic pocket composed by residues Val604,
 165 Val608, Leu620 (found in the G2 helix, which connects α -1

and α -2) and Val629. Kinetically, there is a 10-fold difference
 166 in the mean first passage time for binding between both
 167 sites — $(9.96 \pm 3.57) \cdot 10^3$ ns for binding to *bound* site and
 168 $(1.05 \pm 0.46) \cdot 10^5$ ns for the *secondary* site — that may account
 169 for the preferential binding of c-Myb to the primary interface.
 170

Model validation. To validate the model, we compared the kinetic
 171 parameters derived from it with available information
 172 (17). Experimental values from Shammias et al. were calculated
 173 at temperatures ranging from 278 to 298 K, while
 174 simulations were executed at physiological temperature (310
 175 K). k_{on} values display a temperature independent tendency,
 176 whereas for temperature dependent variables k_{off} and k_d values
 177 had to be extrapolated to 310K (Supplementary Figure 4).
 178 Hence, reference values for k_{off} and free energy (obtained from
 179 k_d) resulted in $866 s^{-1}$ and $-6.81 kcal mol^{-1}$ respectively.
 180

181 Due to the size of the peptide compared to the solvation
 182 box it is hard for the MSM to automatically define the correct
 183 bulk state. We therefore defined a bulk state that contains
 184 conformations where the distance between KIX and cMyb is
 185 maximized. Bulk state was defined by taking those microstates
 186 where the minimum distance between KIX and cMyb is higher
 187 than a threshold. Consequently, some kinetic and thermodynamic
 188 estimates have a dependency on such distance threshold
 189 (Supplementary Figure 6a) as this affects the definition of
 190 the bulk state. However, the computed k_{off} and free energy
 191 estimates are practically stable after a minimal separation distance
 192 of just 4 Å. The MSM estimation of k_{on} is not affected by
 193 the bulk state distance threshold. The obtained estimate is
 194 $(3.61 \pm 1.56) \cdot 10^7 M^{-1} s^{-1}$, in agreement with the experimental
 195 value $(2.2 \pm 0.2) \cdot 10^7 M^{-1} s^{-1}$ (17). k_{off} estimates
 196 range from $6.73 \cdot 10^3 s^{-1}$ to $21.73 \cdot 10^3 s^{-1}$, overestimating
 197 the extrapolated experimental value by an order of magnitude.
 198 Free energy estimates range from $-6.27 kcal mol^{-1}$ to

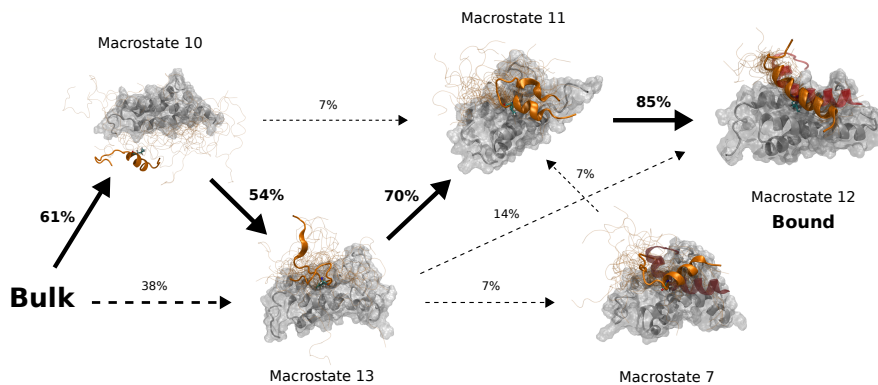


Fig. 3. Complete c-Myb binding process to KIX domain. Main pathways leading from *Bulk* (macrostate 14) to the *Bound* state (macrostate 12). Fluxes are shown as percentages near the arrows. Only those fluxes higher than 5% are shown. The arrow thickness is proportional to the flux percentage. Straight arrows indicate the maximum flux path, while dashed arrows show other fluxes. Each macrostate structure shows KIX as the white surface and ribbons and c-Myb as the orange ribbon and tubes. For each macrostate, 25 conformations are shown as thin tubes with one structure highlighted as a ribbon structure that includes the side chain of Leu302, colored by atom element. Additionally, for macrostate 7 and 12, the reference NMR c-Myb structure is shown as a transparent red ribbon for comparison.

199 $-7.09 \text{ kcal mol}^{-1}$, containing the extrapolated experimental
200 value inside the interval.

201 We further verified the reproducibility of the kinetic and
202 thermodynamic measurements to ensure model convergence by
203 building multiple MSMs using incrementally more trajectories.
204 Convergence is reached at $150 \mu\text{s}$ on all the aforementioned
205 estimates (Supplementary Figure 5). The discrepancy between
206 the experimental reference and computed k_{off} values translate
207 into a faster dissociation in our model. We also verified if this
208 was due to normal discretization errors in the MSM projection
209 or to the fact that our simulations did not obtain a complete
210 bound conformation between KIX and cMyb. In order to
211 test this hypothesis, additional long trajectories (8 replicas
212 of $2 \mu\text{s}$ each) were run starting from bound NMR and MD
213 derived conformations. We constructed an MSM using both
214 simulation datasets. However, the free energy estimations are
215 only marginally improved (Supplementary Figure 6b). Thus,
216 we concluded that the additional bound simulations do not
217 add additional information and we restrict the analysis with
218 just the AdaptiveBandit set of simulations as this is the most
219 general case where no NMR information is available.

220 **Binding follows both induced-fit and conformational selec-**
221 **tion.** In order to gain additional structural insight of the bind-
222 ing process, we constructed an MSM with a higher number of
223 macrostates, using the same lag time. We used transition path
224 theory (27, 28) to calculate fluxes leading from the purely bulk
225 state to the bound conformations. Out of the 15 macrostates
226 of the new MSM, only a reduced set of 6 is sufficient to ex-
227 plain binding to the primary interface. The other macrostates
228 describe either the secondary binding mode or other unstable
229 interactions between KIX and cMyb. The network generated
230 by the flux interchanges between macrostates (Figure 3) can
231 be separated into three events: the establishment of the initial
232 contacts, binding and folding of the N-term section of cMyb

233 and finally, reaching the bound conformation by binding and
234 folding of the C-term section of cMyb. The first step of the
235 binding process features the first native contacts found across
236 the KIX—c-Myb binding pathway, which involves residues
237 Leu302 of c-Myb. The role of Leu302 as the main driving
238 force for the interaction has already been described (3), and
239 is due in part to the kink in the helix created by neighbours
240 residues Met303 and Ser304, which exposes Leu302 allowing
241 for a deep penetration inside the binding pocket. Besides, on
242 the KIX residues contacted at this stage is Leu603, which is
243 one of the most exposed residues in the hydrophobic pocket
244 later occupied by Leu302.

245 To determine if cMyb folding precedes or follows binding
246 at this step, we looked at the flux passing through different
247 cMyb conformations in macrostate 13 (Supplementary Figure
248 8). We see that almost half of the flux goes through N-
249 term helical conformations, suggesting that the presence of
250 helix in residues 297 to 302 facilitates these first binding step,
251 following a conformational selection mechanism. There is
252 also a considerable flux going through unfolded conformations,
253 meaning there is also Leu302 binding through induced-fit.

254 The second step of the binding process goes from macrostate
255 13 to macrostate 11, where several contacts are formed across
256 the N-term of cMyb. The last step, which goes from macrostate
257 11 to 12 (the bound state), involves forming the last contacts
258 on the C-term and completely folding cMyb. Here, we also looked
259 at the flux passing through macrostate 11 (Supplementary
260 Figure 8c) to discern between conformational selection and
261 induced-fit on the C-term folding and binding mechanism.
262 Here, all the flux goes through conformations where the C-
263 term is unfolded, meaning that only when the C-term native
264 contacts start to happen we see a complete cMyb folding to
265 an alpha helix, following a clear induced-fit mechanism.

266 The overall mechanism works as an induced-fit binding
267 and folding, but we can see a mixed mechanism during the

268 first binding steps, where both conformational selection and
 269 induced-fit seem to take a part in facilitating the first contacts
 270 through cMyb’s Leu302. In summary, initial steps can be
 271 greatly benefited from pre-folded helical structures of c-Myb
 272 (Figure 3), although *Binding before folding* is also observed.
 273 Binding of helical conformations dominates the initial steps
 274 of the interaction, but for the interaction of the C-terminal
 275 tail, folding follows binding. No limiting steps in the binding
 276 process are observed; hence no possible transition states can
 277 be defined, as pointed out by experimental reports (17).

278 **Conclusion.** The analysis presented here provides a detailed
 279 molecular description of binding of c-Myb to the primary
 280 interface of KIX, summarized as a two-step process, where
 281 initially the N-terminal region of c-Myb binds with a preferred
 282 helical conformation, allowing the formation of native contacts
 283 and, in the last step, folding and binding of the C-terminal.
 284 Study of the fluxes derived from the MSM show the relevance
 285 of residue Leu302, not only in the final bound structure, but
 286 also as the responsible of establishing the first contacts and
 287 serving as an anchoring point between c-Myb and KIX.

288 The model describes an overall induced-fit binding mecha-
 289 nism, as the complete folding of cMyb is only observed when
 290 native contacts have been formed. Conformational selection
 291 would only affect the first binding stage on residues 298 to 302
 292 and not the whole length of the peptide, whereas the latter
 293 stages of binding follow an induced-fit mechanism.

294 Overall, our results provide a detailed mechanistic model
 295 for the binding of c-Myb to the primary interface of KIX,
 296 as well as showing the interaction with a secondary binding
 297 site, by using unbiased full-atom MD simulations and MSM
 298 analysis. The novel MD sampling approach used in this work,
 299 AdaptiveBandit, had a crucial role in resolving this type of
 300 folding and binding process. The method is implemented and
 301 available in the HTMD python package (23). However, more
 302 algorithms can be derived within the same adaptive bandit
 303 framework. While here we choose the reward to be minus the
 304 free energy, other choices could optimize different costs, for
 305 example, improving the precision of the off-rate or optimizing
 306 sampling in the context of structure prediction.

307 Materials & Methods

308 **Molecular dynamics simulations.** In order to generate initial confor-
 309 mations for c-Myb (residues 291 to 315), we ran multiple parallel
 310 simulations. The peptide was solvated in a cubic water box of 64 Å
 311 side with a NaCl concentration of 0.05 M. First, the peptide was
 312 simulated at 500 K for 120 ns to unfold the initial structure.

313 Then, 200 systems were built by placing one random unstruc-
 314 tured c-Myb conformation in conjunction with KIX in opposite
 315 corners of a 64 Å side cubic water box with a NaCl concentration
 316 of 0.05 M, resulting in final protein concentration of ~3.2 mM.

317 All systems were built using HTMD (23) and simulated with
 318 ACEMD (29), the CHARMM22* force field (30) and TIP3P water
 319 model (31). A Langevin integrator was used with a damping con-
 320 stant of 0.1 ps⁻¹. The integration time step was set to 4 fs, with
 321 heavy hydrogen atoms (scaled up to four times the hydrogen mass)
 322 and holonomic constraints on all hydrogen-heavy atom bond terms.
 323 Electrostatics were computed using PME with a cutoff distance of 9
 324 Å and grid spacing of 1 Å. After energy minimization, equilibration
 325 for all systems was done in an NPT ensemble at 303 K, 1 atm, with
 326 heavy atoms constrained at 1 kcal mol⁻¹ Å². Energy minimization
 327 was run for 500 steps and equilibrated for 2 ns.

328 Production runs of 250 ns were performed at 310 K using the
 329 distributed computing project GPUGrid (32), following an adaptive
 330 sampling strategy. The final data set included 1,809 trajectories of

250 ns, resulting in an aggregated simulation time of ~450 μs. Addi-
 331 tionally, a set of long MD runs were performed starting from bound
 332 structures. Four models of the NMR determined structure and
 333 four random bound conformations were selected and equilibrated as
 334 previously described. A total of 8 long trajectories of ~2 μs each
 335 were generated.
 336

Markov state model analysis. The projected space used for building
 337 the MSM included four different featurizations: all pair C_α + C_β
 338 atoms distances between KIX and c-Myb to account for the inter-
 339 action between the two proteins, self-distances between every C_α of
 340 c-Myb and its secondary structure, to monitor its conformation, and
 341 finally, RMSD to the bound structure. TICA was used at a lag time
 342 τ = 20 ns (implied timescales are shown in Supplementary Figure
 343 1.a) for both the distance features and the secondary structure
 344 features, taking the 4 most relevant components from the distance
 345 features (both inter-distances and cMyb self-distances) and the 3
 346 most relevant components from the secondary structure features.
 347

The 8-dimensional projected data was discretized into 2,000 clus-
 348 ters using the mini batch k-means algorithm (33). The microstates
 349 defined in the MSM were coarse-grained into larger meta-stable
 350 macrostates by using PCCA++ (34). For the estimation of kinetic
 351 values, the original MSM was modified by creating an additional
 352 macrostate, considered as the *bulk* state for all subsequent calcula-
 353 tions to obtain the kinetics of binding. The bulk state was created by
 354 taking those microstates where the minimum distance between KIX
 355 and cMyb was higher than a threshold. Error in kinetic measures
 356 was estimated by creating 50 independent MSMs using a random
 357 set containing 80% of the simulation data.
 358

To obtain the kinetic pathway of binding and folding, we in-
 359 creased the number of macrostates in the MSM using PCCA++
 360 again. Fluxes between macros were estimated using transition path
 361 theory (27, 28). For the intra-macrostate flux analysis, we computed
 362 the mean helicity of cMyb for each microstate in it, and clustered
 363 them into 4 main states which describe the peptide’s grade of helix
 364 formation. All analysis were performed with HTMD (23).
 365

AdaptiveBandit sampling. The multi-armed bandit problem is de-
 366 fined by (A, R, γ), where an action a_t ∈ A and R^a is a (stochastic)
 367 reward function. We choose γ = 0 for totally discounted rewards.
 368 The optimal policy π_a ∼ P[a] selects actions a_t in order to maxi-
 369 mize the cumulative future rewards. The construction of an optimal
 370 selection strategy requires handling the exploration-exploitation
 371 problem. AdaptiveBandit relies on the UCB1 algorithm (25), defin-
 372 ing an upper confidence bound for each action-value estimate based
 373 on the number of times an action has been picked and the total
 374 amount of actions taken
 375

$$376 \quad a_t = \operatorname{argmax}_{a \in A} \left[Q_t(a) + c \sqrt{\frac{\ln t}{N_t(a)}} \right], \quad [1]$$

where t denotes the total number of actions taken, Q_t(a) = E_π[r|a]
 377 is the action-value estimation, N_t(a) is the number of times action
 378 a has been selected (prior to time t) and c is a constant controlling
 379 the degree of exploration. As for the reward definition, there are
 380 different choices depending on the objective, e.g. here, the interest
 381 is sampling the bound metastable state, hence, we rewarded actions
 382 based on the stability of conformations using MSM estimations of
 383 the free energy for each state
 384

$$385 \quad \mathcal{R}_a = -\langle k_B T \log(\mu(x)) \rangle_{(a, x_1, \dots, x_T)}, \quad [2]$$

where μ(x) is the equilibrium distribution estimated by the MSM
 386 with the current available data and the average is performed over
 387 the frames in the trajectory starting from a. AdaptiveBandit uses
 388 the MSM discretized conformational space to define the action set
 389 and at each round acquires a random conformation from the selected
 390 states to respawn new simulations. A more formal description of the
 391 bandit framework and AdaptiveBandit in the context of adaptive
 392 sampling as well as analysis in simpler, analytical potentials are
 393 available at (24). The AdaptiveBandit sampling algorithm is made
 394 available in the HTMD (23) Python package.
 395

Adaptive Sampling parameters. For both the AdaptiveBandit and
 396 the count Adaptive runs, the construction of MSMs at each epoch
 397 was done using the residue-residue contacts between KIX and c-Myb
 398

measured as the minimum contacts between residues at a threshold of 5 Å, and the backbone dihedral angles of c-Myb. Time independent component analysis (TICA) (35) was used for dimensionality reduction using a lag time of $\tau = 20$ frames and keeping the 3 first dimensions, which were later clustered with a k-centers algorithm. AdaptiveBandit was performed during 40 epochs with a c value of 0.01.

ACKNOWLEDGMENTS. The authors thank Kresten Lindorff-Larsen and Frank Noé for their critical reading of the manuscript. The authors also thank volunteers at GPUGRID.net for contributing with computational resources and Acclera for funding. G.D.F. acknowledges support from MINECO (Unidad de Excelencia María de Maeztu CEX2018-000782-M and BIO2017-82628-P) and FEDER. This project received funding from the European Union’s Horizon 2020 Research and Innovation Programme under Grant Agreement No. 823712 (CompBioMed2 Project).

1. HJ Dyson, PE Wright, Intrinsically unstructured proteins and their functions. *Nat. reviews Mol. cell biology* **6**, 197 (2005).
2. PH Kussie, et al., Structure of the MDM2 oncoprotein bound to the p53 tumor suppressor transactivation domain. *Science* **274**, 948–953 (1996).
3. T Zor, RN De Guzman, HJ Dyson, PE Wright, Solution structure of the KIX domain of CBP bound to the transactivation domain of c-Myb. *J. molecular biology* **337**, 521–534 (2004).
4. I Buch, T Giorgino, G De Fabritiis, Complete reconstruction of an enzyme-inhibitor binding process by molecular dynamics simulations. *Proc. Natl. Acad. Sci.* **108**, 10184–10189 (2011).
5. N Plattner, S Doerr, G De Fabritiis, F Noé, Complete protein-protein association kinetics in atomic detail revealed by molecular dynamics simulations and markov modelling. *Nat. chemistry* **9**, 1005 (2017).
6. A Borgia, et al., Extreme disorder in an ultrahigh-affinity protein complex. *Nature* **555**, 61 (2018).
7. K Lindorff-Larsen, S Piana, RO Dror, DE Shaw, How fast-folding proteins fold. *Science* **334**, 517–520 (2011).
8. S Piana, K Lindorff-Larsen, DE Shaw, Atomistic description of the folding of a dimeric protein. *The J. Phys. Chem. B* **117**, 12935–12942 (2013).
9. MC Zwieter, et al., Efficient atomistic simulation of pathways and calculation of rate constants for a protein-peptide binding process: application to the MDM2 protein and an intrinsically disordered p53 peptide. *The journal physical chemistry letters* **7**, 3440–3445 (2016).
10. JA Morrone, A Perez, J MacCallum, KA Dill, Computed binding of peptides to proteins with melt-accelerated molecular dynamics. *J. chemical theory computation* **13**, 870–876 (2017).
11. G Zhou, GA Pantelopulos, S Mukherjee, VA Voeltz, Bridging microscopic and macroscopic mechanisms of p53-MDM2 binding with kinetic network models. *Biophys. Journal* **113**, 785–793 (2017).
12. F Paul, et al., Protein-peptide association kinetics beyond the seconds timescale from atomistic simulations. *Nat. communications* **8**, 1095 (2017).
13. SH Chong, H Im, S Ham, Explicit characterization of the free energy landscape of pkid-kix coupled folding and binding. *ACS Cent. Sci.* **5**, 1342–1351 (2019).
14. M Arai, K Sugase, HJ Dyson, PE Wright, Conformational propensities of intrinsically disordered proteins influence the mechanism of binding and folding. *Proc. Natl. Acad. Sci.* **112**, 9614–9619 (2015).
15. R Giri, A Morrone, A Toto, M Brunori, S Gianni, Structure of the transition state for the binding of c-Myb and KIX highlights an unexpected order for a disordered system. *Proc. Natl. Acad. Sci.* **110**, 14942–14947 (2013).
16. S Gianni, A Morrone, R Giri, M Brunori, A folding-after-binding mechanism describes the recognition between the transactivation domain of c-Myb and the KIX domain of the CREB-binding protein. *Biochem. biophysical research communications* **428**, 205–209 (2012).
17. SL Shammam, AJ Travis, J Clarke, Remarkably fast coupled folding and binding of the intrinsically disordered transactivation domain of cMyb to CBP KIX. *The journal physical chemistry B* **117**, 13346–13356 (2013).
18. A Toto, et al., Molecular recognition by templated folding of an intrinsically disordered protein. *Sci. reports* **6**, 21994 (2016).
19. A Poosapati, E Gregory, WM Borchers, LB Chemes, GW Daughdrill, Uncoupling the folding and binding of an intrinsically disordered protein. *J. molecular biology* **430**, 2389–2402 (2018).
20. SL Shammam, AJ Travis, J Clarke, Allostery within a transcription coactivator is predominantly mediated through dissociation rate constants. *Proc. Natl. Acad. Sci.* **111**, 12055–12060 (2014).
21. K Sugase, HJ Dyson, PE Wright, Mechanism of coupled folding and binding of an intrinsically disordered protein. *Nature* **447**, 1021 (2007).
22. S Doerr, G De Fabritiis, On-the-fly learning and sampling of ligand binding by high-throughput molecular simulations. *J. chemical theory computation* **10**, 2064–2069 (2014).
23. S Doerr, M Harvey, F Noé, G De Fabritiis, HTMD: high-throughput molecular dynamics for molecular discovery. *J. chemical theory computation* **12**, 1845–1852 (2016).
24. A Pérez, P Herrera-Nieto, S Doerr, GD Fabritiis, Adaptivebandit: A multi-armed bandit framework for adaptive sampling in molecular simulations (2020).
25. P Auer, Using confidence bounds for exploitation-exploration trade-offs. *J. Mach. Learn. Res.* **3**, 397–422 (2002).
26. JH Prinz, et al., Markov models of molecular kinetics: Generation and validation. *The J. chemical physics* **134**, 174105 (2011).
27. E Weinan, E Vanden-Eijnden, Towards a theory of transition paths. *J. statistical physics* **123**, 503 (2006).
28. F Noé, C Schütte, E Vanden-Eijnden, L Reich, TR Weikl, Constructing the equilibrium en-

- semble of folding pathways from short off-equilibrium simulations. *Proc. Natl. Acad. Sci.* **106**, 19011–19016 (2009).
29. MJ Harvey, G Giupponi, GD Fabritiis, ACEMD: accelerating biomolecular dynamics in the microsecond time scale. *J. chemical theory computation* **5**, 1632–1639 (2009).
30. S Piana, K Lindorff-Larsen, DE Shaw, How robust are protein folding simulations with respect to force field parameterization? *Biophys. Journal* **100**, L47–L49 (2011).
31. WL Jorgensen, J Chandrasekhar, JD Madura, RW Impey, ML Klein, Comparison of simple potential functions for simulating liquid water. *The J. chemical physics* **79**, 926–935 (1983).
32. I Buch, MJ Harvey, T Giorgino, DP Anderson, G De Fabritiis, High-throughput all-atom molecular dynamics simulations using distributed computing. *J. chemical information modeling* **50**, 397–403 (2010).
33. F Pedregosa, et al., Scikit-learn: Machine learning in Python. *J. machine learning research* **12**, 2825–2830 (2011).
34. S Rößitz, M Weber, Fuzzy spectral clustering by pcca+: application to markov state models and data classification. *Adv. Data Analysis Classif.* **7**, 147–179 (2013).
35. G Pérez-Herrández, F Paul, T Giorgino, G De Fabritiis, F Noé, Identification of slow molecular order parameters for Markov model construction. *The J. chemical physics* **139**, 07B604_1 (2013).

Supplementary Information for

Reconstruction of the binding pathway of an intrinsically disordered protein using Molecular Dynamics simulations

Pablo Herrera-Nieto, Adrià Pérez and Gianni De Fabritiis

Gianni De Fabritiis.
E-mail: gianni.defabritiis@upf.edu

This PDF file includes:

Figs. S1 to S9

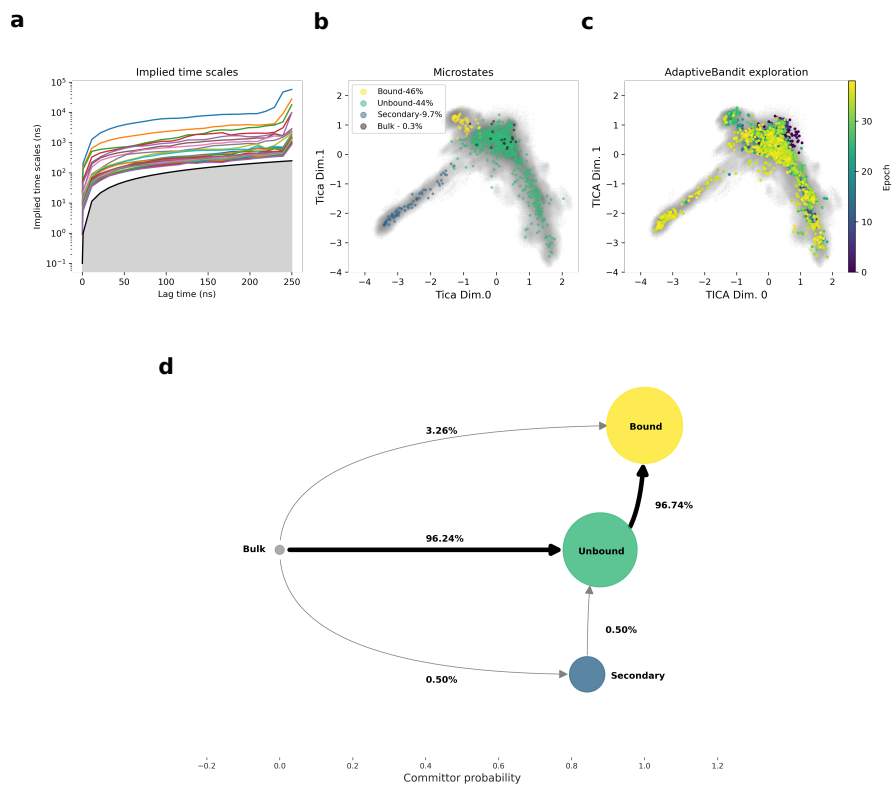


Fig. S1. Markov state model summary a) Implied time scales of the MD data. b) Microstate distribution across the first two TICA dimensions. Each microstate is colored by their corresponding macrostate. Legend shows the population of each macrostate. c) AdaptiveBandit exploration of the TICA space. Each colored point indicates a starting point selected by AdaptiveBandit to respawn a new trajectory. Color indicates the epoch number. In grey, the area covered by the projected simulation data without clustering, both in b) and c). d) Flux pathway from *bulk* to *bound*. Nodes are placed according to the committor probability. The y axis is manually set for better visualization of the graph. Node size is proportional to the equilibrium distribution. Node color corresponds to macrostate assignment as in b). Flux percentage is shown near each arrow. Main pathway is indicated with black, thicker arrows.

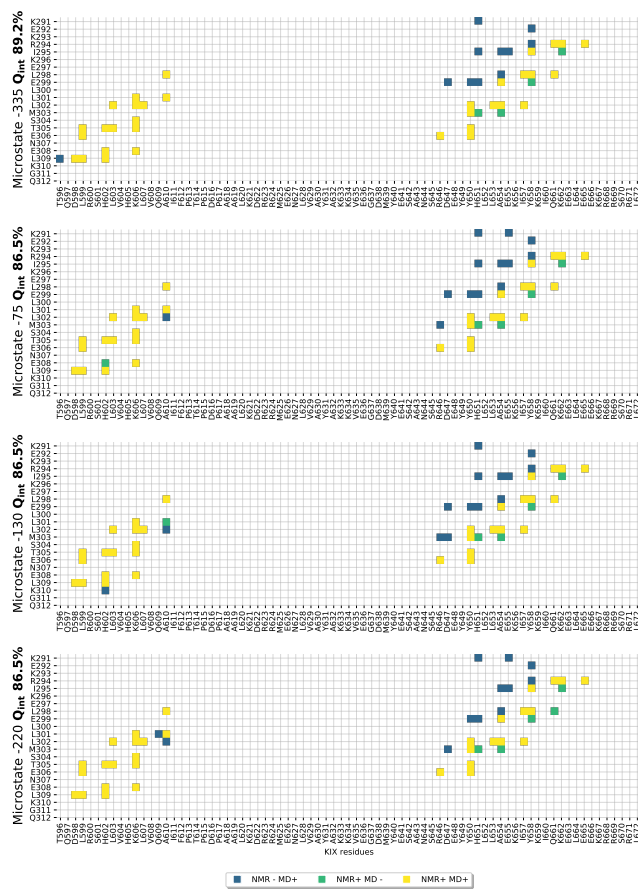


Fig. S2. Maximum Q_{int} microstates contact fingerprint. Profile of contacts established between c-Myb and KIX in microstates with maximum fraction of native binding contacts Q_{int} . Blue color represents contacts present in the state but not in the original NMR conformation, green indicates native contacts not found in the MSM state and yellow squares represent a match on that contact, found in both the NMR model and MD microstate. A contact is considered present in a microstate when it appears in at least 50% of the conformations in that state.

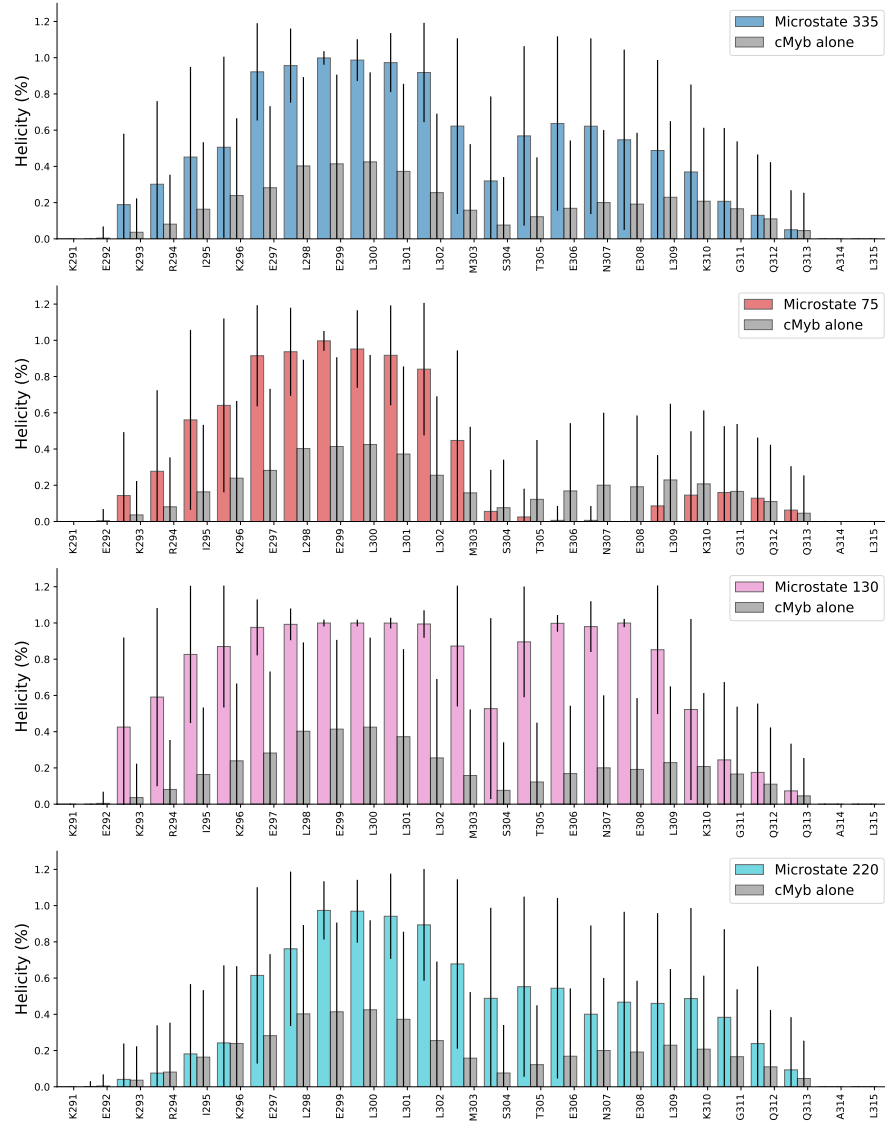


Fig. S3. c-Myb helicity. Comparison of the by-residue helicity fraction of c-Myb between the four microstates with maximum Q_{int} . The helicity profile for the peptide in

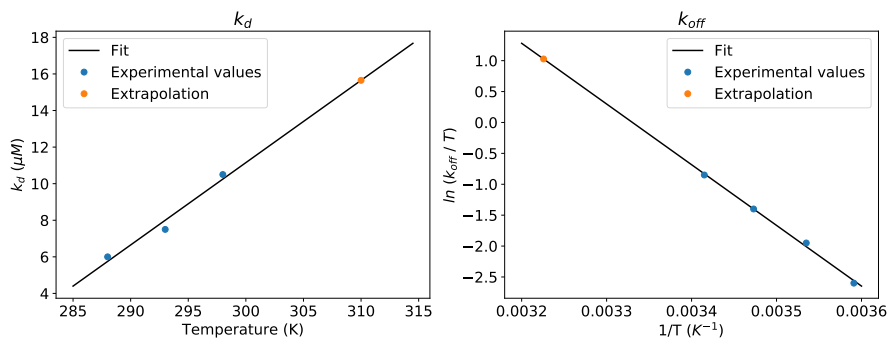


Fig. S4. Extrapolations of a) k_d and b) k_{off} values from experimental data.

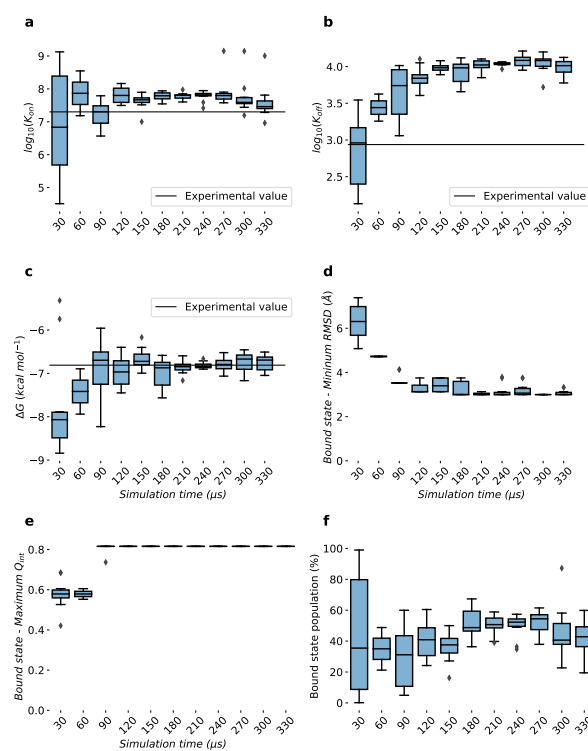


Fig. S5. Statistics convergence across the MD run of a) k_{on} , b) k_{off} , c) free energy and d) microstate minimum RMSD, e) bound state maximum Q_{int} and f) bound state population, computed by the MSM. Each data point was calculated by building 10 different MSMs, bootstrapping 80% of the trajectories each time.

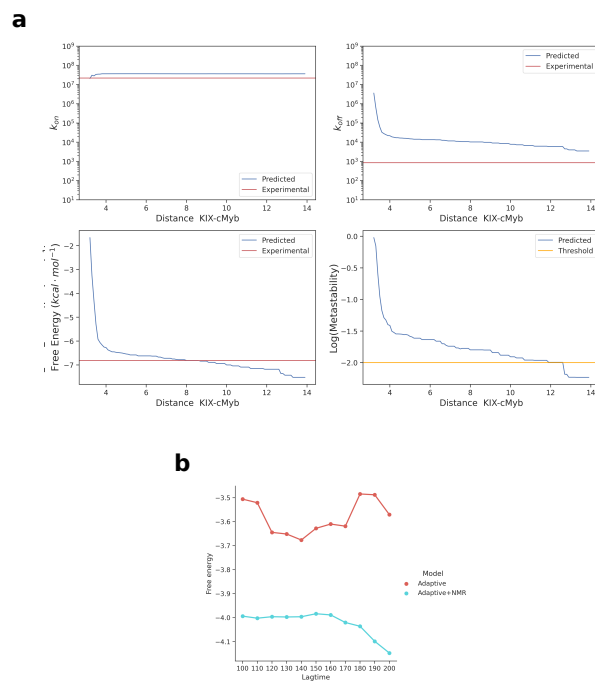


Fig. S6. a) Bulk state variability of K_{on} , K_{off} , free energy and metastability term, depending on the maximum distance threshold between KIX and cMyb used. The metastability term is defined as the bulk state self-transition probability (named for plot simplicity). Blue line shows the variable estimate, red line shows the reference experimental value and yellow line shows the defined threshold for deciding whether the bulk state is stable enough or not. **b)** Free energy estimates at different MSM lag times for the AdaptiveBandit simulations alone and together with the long trajectories starting from the NMR structures. The models were performed without defining a bulk state.

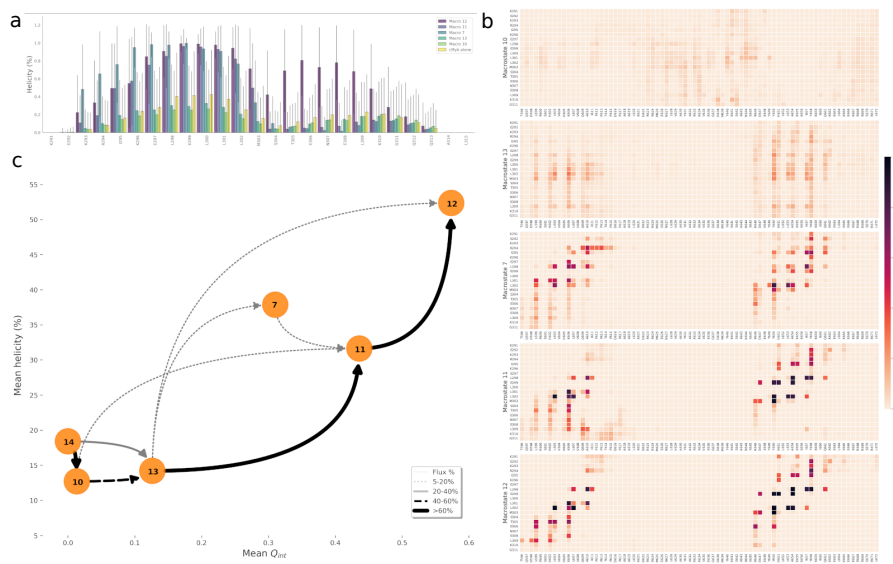


Fig. S7. Complete binding process of c-Myb to KIX. Structural analysis of the states involved in the main binding flux pathway on the 15 macrostate MSM. **a)** Mean helicity per residue and **b)** mean contacts profile is shown for the macrostates present in the binding process. **c)** Main pathways leading from Macrostate 14 (*Bulk*) to Macrostate 12 (*Bound*). Nodes are placed according to the fraction of native contacts $Q_{i,i}$, with respect to the NMR model on the x axis, and mean helicity on the y axis. Arrows represent the connection between macrostates, and their color, thickness and trace the percentage of the total flux traversing them.

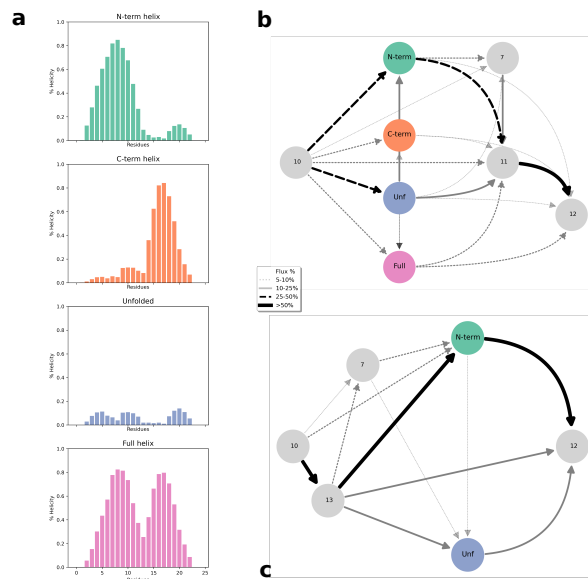


Fig. S8. Detailed intra-macrostate flux analysis. **a**) Cluster centers corresponding to the four main cMyb folded states: Full helix, Unfolded, N-terminal helix, C-terminal helix. These centers were computed using the mean helicity of all microstates. The centers are displayed with bar plots showing the helicity per residue. **b,c**) Flux pathways across clustered macrostates 13 (b) and 11 (c). The selected macrostates were clustered using the four centers defined in a), and the flux was recomputed using these newly defined clusters. Node colours and names indicate which cluster center from a) they correspond. Node positioning was manually set for visualization purposes. Arrows represent the connection between macrostates/clusters, and their color, thickness, and trace the percentage of the total flux traversing them.

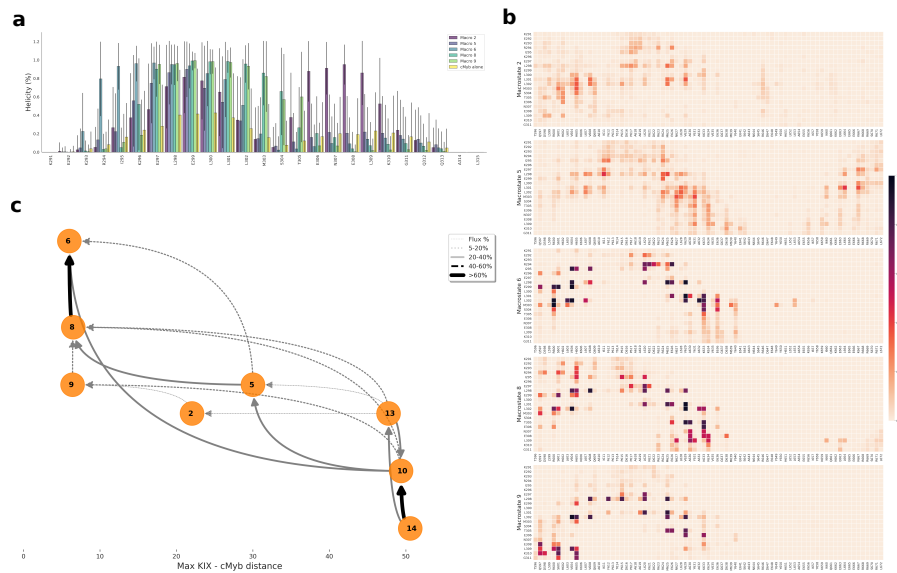


Fig. S9. Secondary binding path of KIX and c-Myb. Study of the states involved in the secondary binding pathway, using the 15 macrostate MSM. For selected macrostates the **a)** mean helicity and **b)** KIX—c-Myb contacts profile is shown. Contact and helicity data for macrostates 10 and 13 are shown in FigS7. **c)** Main pathways leading from Macrostate 14 (*Bulk*) to Macrostate 6 (*Secondary*). Nodes are placed according to the maximum distance between KIX and cMyb of each state on the *x* axis. The *y* axis is manually set for better visualization of the graph. Arrows represent the connection between macrostates, and their color, thickness, and trace the percentage of the total flux traversing them.

3.3 TorchMD: A Deep Learning Framework for Molecular Simulations

Doerr S, Majewski M, Pérez A, Krämer A, Clementi C, Noé F, Giorgino T, De Fabritiis G. [TorchMD: A Deep Learning Framework for Molecular Simulations](#). *Journal of Chemical Theory and Computation* 2021;17(4):2355-2363

Summary

In this paper, we present TorchMD, a framework made to combine molecular simulations with machine learned potentials. TorchMD is a molecular dynamics code made entirely with PyTorch, which allows for an easy integration with other machine learning potentials based in the same library. The common bonded and non-bonded terms of MD are extended with an additional term for neural-network based potentials. Additionally, due to the intrinsic nature of PyTorch, TorchMD is able to perform end-to-end differentiable simulations.

3.4 Simulations meet machine learning in structural biology

Pérez A, Martínez-Rosell G, De Fabritiis G. [Simulations meet machine learning in structural biology](#). *Current Opinion in Structural Biology* 2018;49:139-144.

Summary

In this opinion article we precede the upcoming scenario of molecular simulations being used as a data generation tool, where machine learning methods will be used to construct predictive models based on expensive simulation data. We discuss the data generation capacity of MD simulations, as well as its limitations in sampling and accuracy, and we envision the combination of simulations with novel machine learning algorithms to fix the of low-throughput, high-latency predictions of MD simulations.

TorchMD: A Deep Learning Framework for Molecular Simulations

Stefan Doerr, Maciej Majewski, Adrià Pérez, Andreas Krämer, Cecilia Clementi, Frank Noe, Toni Giorgino, and Gianni De Fabritiis*

 Cite This: *J. Chem. Theory Comput.* 2021, 17, 2355–2363

 Read Online


ACCESS |

 Metrics & More

 Article Recommendations

 Supporting Information

ABSTRACT: Molecular dynamics simulations provide a mechanistic description of molecules by relying on empirical potentials. The quality and transferability of such potentials can be improved leveraging data-driven models derived with machine learning approaches. Here, we present TorchMD, a framework for molecular simulations with mixed classical and machine learning potentials. All force computations including bond, angle, dihedral, Lennard-Jones, and Coulomb interactions are expressed as PyTorch arrays and operations. Moreover, TorchMD enables learning and simulating neural network potentials. We validate it using standard Amber all-atom simulations, learning an ab initio potential, performing an end-to-end training, and finally learning and simulating a coarse-grained model for protein folding. We believe that TorchMD provides a useful tool set to support molecular simulations of machine learning potentials. Code and data are freely available at github.com/torchmd.



1. INTRODUCTION

Classical molecular dynamics (MD) is a compute-intensive technique that enables quantitative studies of molecular processes. Of the possible modeling approaches, classical all-atom MD represents all of the atoms of a chosen system explicitly (including solvent) and accounts for interatomic forces through classical bonded and nonbonded potentials. It has seen remarkable developments due to its fidelity, and it has been applied with success to problems such as conformational changes, folding, binding, permeation, and many others.¹ It has, however, faced two significant challenges: first, the calculation of the tables of interatomic potentials known as force fields² has traditionally been highly time-consuming and requires significant fine-tuning; second, it is compute-intensive, and despite heroic efforts and progress in accelerating MD codes,³ it still struggles to reach the temporal scales of several important physiological processes.

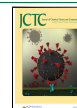
Machine learning (ML) potentials have become especially attractive with the advent of deep neural network (DNN) architectures, which enable the example-driven definition of arbitrarily complex functions and their derivatives. As such, DNNs offer a very promising avenue to embed fast-yet-accurate potential energy functions in MD simulations, after training on large-scale databases obtained from more expensive approaches. One particularly interesting feature of neural network potentials is that they can learn many-body interactions. The SchNet architecture,^{4,5} for instance, learns a set of features using continuous filter convolutions on a graph neural network and predicts the forces and energy of the system. SchNet was originally used in quantum chemistry to predict energies of small molecules from their atomistic representations. A key feature of using SchNet is that the model is inherently transferable across molecular systems. More recently, this has been extended to learn a potential of mean force which involves averaging of a potential over some

coarse-grained degrees of freedom,^{6–12} which however pose challenges in their parametrization.^{13,14} Indeed, molecular modeling on a more granular scale has been tackled by so-called coarse-graining (CG) approaches before,^{15–20} but it is particularly interesting in combination with DNNs.

Here, we introduce TorchMD, a molecular dynamics code built from scratch to leverage the primitives of the ML library PyTorch.²¹ TorchMD enables the rapid prototyping and integration of machine-learned potentials by extending the bonded and nonbonded force terms commonly used in MD with DNN-based ones of arbitrary complexity. The two key points of TorchMD are that, being written in PyTorch, it is very easy to integrate other ML PyTorch models, like ab initio neural network potentials (NNPs)^{5,22} and machine learning coarse-grained potentials.^{8,9} Second, TorchMD provides the capability to perform end-to-end differentiable simulations,^{14,23,24} being differentiable on all of its parameters. Similarly, Jax²⁵ was used to perform end-to-end differentiable molecular simulations on Lennard-Jones systems²⁶ and for biomolecular systems as well.²⁷ Other efforts have tackled the integration of MD codes with DNN libraries, although in different contexts. For all-atom models, Wang et al.²³ demonstrated the use of graph networks to recover empirical atom types. Ab initio QM-based training of potentials is being tackled by several groups, including Gao et al.,²² Yao et al.,²⁸ and Schütt et al.²⁹ but not using a differentiable PyTorch environment.

Received: December 29, 2020

Published: March 17, 2021



This paper provides an account of the capabilities of TorchMD (Section 2), highlighting the functional forms supported and an effective fitting strategy for data-driven DNN potentials. All of the TorchMD code, including a tutorial on coarse-graining the chignolin protein and the corresponding training data, is open-source and available at github.com/torchmd.

2. METHODS

2.1. TorchMD Simulations. TorchMD is, at first glance, a standard molecular dynamics code. It offers NVT ensemble simulations including a Langevin thermostat. Starting atomic velocities are derived from a Maxwell–Boltzmann distribution. Integration is done using the velocity Verlet algorithm. Long-range electrostatics are approximated using the reaction field method.³⁰ TorchMD also supports simulations of periodic systems. Minimization is done using the L-BFGS algorithm. Because it is written in Python using PyTorch arrays, it is also very simple to modify, and simulations can be run on any devices supported by PyTorch (CPU, GPU, TPU). However, unlike specialized MD codes³¹ it is not designed for speed. TorchMD uses chemical units consistent with classical MD codes such as ACEMD,³¹ namely kcal/mol for energies, K for temperatures, g/mol for masses, and Å for distances.

2.2. Analytical Potentials. TorchMD supports reading AMBER force-field parameters through parmed.³² In addition to that, to allow for faster prototyping and development, it implements its own easy to read YAML-based force-field format. An example YAML force-field file for the simulation of a water box is given in Figure 1. Currently, TorchMD’s missing features include hydrogen bond constraints and neighbor lists.

TorchMD implements the functional form of the AMBER potential.³³ It offers all basic AMBER terms: harmonic bonds, angles, torsions, and nonbonded van der Waals and electrostatic energies. The above potentials are implemented as follows. The bonded potential terms are calculated as

$$V_{\text{bonded}} = k_0(r - r_{\text{eq}})^2$$

where k_0 is the force constant, r is the distance between the bonded atoms, and r_{eq} is the equilibrium distance between them.

The angle terms are calculated as

$$V_{\text{angle}} = k_\theta(\theta - \theta_{\text{eq}})^2$$

where θ is the angle between the three bonded atoms, k_θ is the angular force constant, and θ_{eq} is the equilibrium angle.

The torsion terms are calculated as

$$V_{\text{torsion}} = \sum_{n=1}^{n_{\text{max}}} k_n(1 + \cos(n\phi - \gamma))$$

where ϕ is the dihedral angle between the four atoms, γ is the phase offset, and k_n is the amplitude of the harmonic component of periodicity n .

The nonbonded van der Waals (VdW) terms are calculated as

$$V_{\text{VdW}} = \frac{A}{r^{12}} - \frac{B}{r^6}$$

where $A = 4\epsilon\sigma^{12}$ and $B = 4\epsilon\sigma^6$ with ϵ being the well depth of the interaction of two atoms, and σ is the distance at which the energy is zero. The VdW potential also supports a cutoff by

```

1  atomtypes: [OT, HT]
2
3  bonds:
4  (OT, HT):
5    k0: 450.000
6    req: 0.9572
7  (HT, HT):
8    k0: 0.000
9    req: 1.5139
10
11 angles:
12 (HT, OT, HT):
13   k0: 55.000
14   theta0: 104.52
15
16 lj:
17 OT:
18   sigma: 3.1507
19   epsilon: -0.1521
20 HT:
21   sigma: 0.4000
22   epsilon: -0.0460
23
24 electrostatics:
25 OT:
26   charge: -0.834
27 HT:
28   charge: 0.417
29
30 masses:
31 OT: 15.9994
32 HT: 1.008
    
```

Figure 1. An example YAML force field for water molecules.

using a switching distance. Its energy is then obtained by multiplying the V_{VdW} term with the scaling factor

$$S = 1 - 6x^3 + 15x^4 - 10x^5$$

with $x = (r - r_s)/(r_c - r_s)$

where r_s is the switching distance, and r_c is the cutoff distance.

Electrostatics without cutoff are implemented using the following potential

$$V_{\text{electrostatic}} = k_e \frac{q_i q_j}{r}$$

where $k_e = \frac{1}{4\pi\epsilon_0}$ is Coulomb’s constant, q_i and q_j are the charges of the two atoms, and r is the distance between them. Electrostatics with cutoff are modified to use the reaction field method³⁰ as follows

$$V_{\text{electrostatic}} = k_e q_i q_j \left(\frac{1}{r} + k_{\text{rf}} r^2 - c_{\text{rf}} \right)$$

$$k_{\text{rf}} = \left(\frac{1}{r_c^3} \right) \left(\frac{\epsilon_{\text{sol}} - 1}{2\epsilon_{\text{sol}} + 1} \right)$$

$$c_{\text{rf}} = \left(\frac{1}{r_c} \right) \left(\frac{3\epsilon_{\text{sol}}}{2\epsilon_{\text{sol}} + 1} \right)$$

where r_c corresponds to the cutoff distance, and ϵ_{sol} corresponds to the solvent dielectric constant.

In addition to the above, TorchMD also trivially allows the use of any other external potential V_{ext} written in PyTorch which takes atomic coordinates as input and output energy and forces.

Thus, the total potential is calculated as

$$V_{total} = \sum_{bonds}^{n_{bonds}} V_{bonded} + \sum_{angles}^{n_{angles}} V_{angle} + \sum_{torsions}^{n_{torsions}} V_{torsion} + \sum_i^{n_{atms}} \sum_{j<i}^{n_{atms}} (V_{VDW} + V_{electrostatic}) + \sum_{ext}^{n_{ext}} V_{ext} \quad (1)$$

Since PyTorch offers automatic differentiation, there is no need to calculate analytical gradients from the forces. Forces can be obtained with a single autograd PyTorch call on the total energy of the system. Analytical gradients have been nevertheless implemented for all analytical AMBER potential terms for performance reasons.

2.3. Training Machine Learning Potentials. TorchMD provides a fully usable code for training neural network potentials in PyTorch called TorchMD-Net (github.com/torchmd/torchmd-net). Currently we are using a SchNet-based⁴ model. However, it would be straightforward to derive the forces from nonparametric kernel methods like FCHL,³⁴ by providing a simple force calculator class, or other ML potentials. This object just takes as input the positions and box every time step and returns the external energies and forces computed with the method of choice.

For the present work, we took the featurization and atom-wise layer from SchNetPack³⁹ but rewrote entirely the training and inference parts. In particular, to allow training on multiple GPUs, the network is trained using the PyTorch lightning framework.³⁵ TorchMD can also run concurrently a set of identical simulations by just changing the random number generator seed, arranging the neural network potential into a batch for speed, thus recovering, at least partially, the efficiency of optimized molecular dynamics codes.

For the QM9 data set,³⁶ we trained the model using a standard loss function using mean square error over the energies. For the coarse-grained model, training is performed using the bottom-up "force matching" approach, focused on reproducing thermodynamics of the system from atomistic simulations, as described in previous work.^{8,9}

3. RESULTS

To demonstrate the functionalities of TorchMD, here we present some application examples. First, a set of typical MD use cases (water box, small peptide, protein, and ligand) is used mainly to assess speed and energy conservation. Second, we validate the training procedure on QM9, a data set of 134k small molecule conformations with energies.³⁶ In this case, however we cannot run any dynamical simulations as the data set only presents ground state conformations of the molecules, so we are mainly validating the training. Then, we demonstrate end-to-end differentiable capabilities of TorchMD by recovering force-field parameters from a short MD trajectory. Finally, we present a coarse-grained simulation of a miniprotein, chignolin,³⁷ using NNP trained on all-atom MD simulation data. Here, we also describe how to produce a neural network-based coarse-grained model of chignolin, although the methods exposed are general to any other protein. A step-

by-step example of the training and simulating CG model is presented in the tutorial available in the github.com/torchmd/torchmd-cg repository.

3.1. Simulations of All-Atom Systems and Performance. The performance of TorchMD is compared against ACEMD3,³¹ a high-performance molecular dynamics code. In Table 1, we can see the three different test systems comprised

Table 1. Performance Comparison for 50,000 Steps at 1 fs/timestep on Different Systems

system	atoms	TorchMD	ACEMD
water	291	6 min 56 s	7 s
alanine dipeptide	688	8 min 44 s	8 s
trypsin	3,248	13 min 2 s	16 s

of a simple periodic water box of 97 water molecules, alanine dipeptide, and trypsin with the ligand benzamidine bound to it. As it can be seen, TorchMD is around 60 times slower on the test systems than ACEMD3 running on a TITAN V NVIDIA GPU. Most of the performance discrepancy can be attributed to the lack of neighbor lists for nonbonded interactions in TorchMD and is currently prohibitive for much larger systems as the pair distances cannot fit into GPU memory. This is not a strongly limiting factor for the CG simulations conducted in this paper as the number of beads remains relatively low for the test case. However, we believe that, with an upcoming implementation of neighbor lists, TorchMD can reach a much better performance, albeit still slower than highly specialized codes as ACEMD3 due to the generic nature of PyTorch operations in addition to the PyTorch library overhead.

Despite the low performance of the current TorchMD implementation, its end-to-end differentiability allows researchers to perform experiments which would not be possible with traditional much faster MD codes as demonstrated in the following sections.

To evaluate the correctness of the TorchMD implementation of the AMBER force field, we compared it against OpenMM for 14 different systems ranging from ions, water boxes, and small molecules to whole proteins, thus testing all the different force-field terms. In all 14 test cases, the potential energy difference between OpenMM and TorchMD was lower than 10^{-3} kcal/mol when computed with the same parameters. Energy conservation was validated with TorchMD by running an NVE simulation of a periodic water box for 1 ns with a 1 fs time step. Energy conservation normalized per degree of freedom was calculated as $E_{total}/n_{dof}R$ where $n_{dof} = 870$ is the number of degrees of freedom of the system, and R is the ideal gas constant. We obtained a mean value of 1.1×10^{-5} K per degree of freedom showing a good energy conservation.

3.2. Training Validation on the QM9 Data Set. We trained and evaluated the performance on the QM9 benchmark data set³⁶ in order to validate the training procedure of TorchMD-Net. QM9 comprises 133,885 small organic molecules with up to nine heavy atoms of type C, O, N, and F reporting several thermodynamic, energetic, and electronic properties for each molecule. We trained on the energy U0 and excluded 3,054 molecules due to failed geometric consistency checks as suggested by the data set. The remaining molecules were split into a training set with 110,000 samples and a validation set with 6,541 samples (5%), leaving 14,290 samples for testing.

The network was trained using an Adam optimizer³⁸ with a learning rate scheduler on multiple GPUs by using PyTorch Lightning.³⁵ An example of the configuration file for QM9 training is presented in Figure 2. We performed multiple

```

1 batch_size: 128
2 cutoff: 5.0
3 data: ./qm9.db
4 label: energy_U0
5 lr: 0.0004
6 lr_factor: 0.8
7 lr_patience: 10
8 early_stopping_patience: 30
9 model: schnet
10 max_z: 100
11 num_epochs: 500
12 num_filters: 256
13 num_gaussians: 25
14 num_interactions: 6
15 test_ratio: 0.1
16 val_ratio: 0.05
    
```

Figure 2. An example of a training input file for training QM9.

trainings using TorchMD-Net with different amounts of training data (Figure 3). The learning rate scheduler was

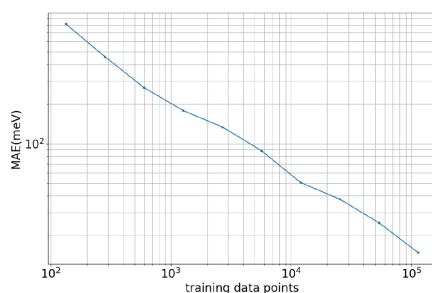


Figure 3. Learning curve for the QM9 data set.

determined with a patience of 10 on a validation subset of 5% of all data chosen at random. The performance reported is for the randomly chosen test set. The linear shape of the test performance in the log–log scale demonstrates the correctness of the training.³⁹ With the current set of hyperparameters (Figure 2), we report a best performance of 10 meV for 110,000 training points, marginally better than the reported best performance of SchNet for QM9.²⁹

3.3. Demonstration of End-to-End Differentiable Simulations. The availability of automatic differentiation (AD) within a molecular dynamics package is beneficial beyond ML applications. Being able to compute gradients for all numerical operations opens up new avenues for sensitivity analysis, force-field optimization, and steered MD simulations, as well as simulations under highly complex constraints and restraints. To demonstrate these capabilities, the present example infers force-field parameters from a short MD trajectory.

First, a small water box containing 97 water molecules and one Na⁺/Cl⁻ ion pair was simulated using the TIP3P water model with flexible bonds and angles. After energy minimization and NVT equilibration at 300 K, the simulation was run for 10 ps in the microcanonical ensemble. The simulation used a 1 fs time step, a 9 Å cutoff with a 7.5 Å switch distance, and reaction field electrostatics. Coordinates and velocities were saved every 10 steps.

Next, all partial atomic charges q in the system were annihilated (in practice, they were scaled by 0.01 to ensure nonvanishing gradients of the electrostatic potential). In order to infer q from the MD trajectory, the integrator was initialized with snapshots $r(t_i)$, $v(t_i)$ from the trajectory. Then, 10 steps of simulation were run with the modified charges, and the final positions from this short simulation were compared with the respective subsequent trajectory snapshot $r(t_{i+1})$. In other words, the simulation served as a parametrized propagator $Q: (r(t), v(t); q) \rightarrow r(t + \delta_i)$ with $\delta_i = 10$ fs. Due to the AD capabilities within TorchMD, this propagator is end-to-end differentiable.

To recover the charges, we minimized the loss function

$$L(r(t_i), v(t_i); q) = \|Q(r(t_i), v(t_i); q) - r(t_{i+1})\|_2^2$$

i.e., the mean-squared distance between the ground-truth trajectory and the propagated coordinates (taking into account periodic boundary conditions). This loss function is differentiable with respect to the charges q so that gradients can be obtained via backpropagation. Training was performed using Adam with a learning rate of 10^{-3} over one snapshot at a time. To enforce net neutrality, the positive charges (q_H and q_{Na^+}) were implicitly obtained from the oxygen and chlorine charges, and only q_O and q_{Cl^-} were explicitly optimized. Figure 4 shows the evolution of the training loss and the partial atomic charges

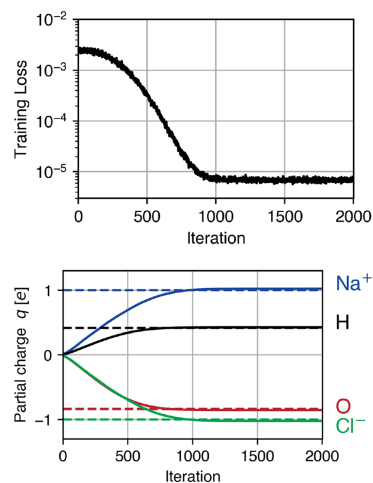


Figure 4. Inference of partial atomic charges q from a short trajectory. Training loss (top) and charges (bottom) during training.

during training. After just one epoch (1000 iterations), the original charges were recovered up to 3% accuracy.

3.4. Coarse-Graining All-Atom Systems. For our last application example, we built two coarse-grained models of chignolin: one solely based on α -carbon atoms (CA) and another one based on α -carbon and β -carbon atoms (CACB) (Figure 5).

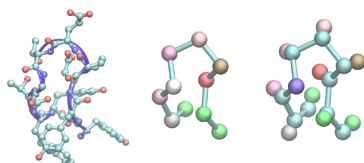


Figure 5. Miniprotein chignolin: heavy-atom representation (left) and coarse-grained representations: CA beads connected by bonds (middle) and CA and CB beads connected by bonds (right). The beads in coarse-grained representations were colored by bead type.

3.4.1. Training Data. We selected the CLN025 variant of chignolin (sequence YYDPETGTWY), which forms a β -hairpin turn while folded (Figure 5). Due to its small size (10 amino acids) and fast folding, it has been extensively studied with MD.^{40–45} Training data was obtained from an all-atom simulation of the protein in explicit solvent with ACEMD³¹ on the GPUGRID.net distributed computing network.⁴⁶ The system containing one chignolin chain was solvated in a cubic box of 40 Å, containing 1881 water molecules and two Na⁺ ions. The system was simulated at 350 K with the CHARMM22⁴⁷ force field and the TIP3P model of water.⁴⁸ A Langevin integrator was used with a damping constant of 0.1 ps⁻¹. The integration time step was set to 4 fs, with heavy hydrogen atoms (scaled up to four times the hydrogen mass) and holonomic constraints on all hydrogen-heavy atom bond terms.⁴⁹ Electrostatics were computed using Particle Mesh Ewald with a cutoff distance of 9 Å and a grid spacing of 1 Å. We used an adaptive sampling approach⁵⁰ where new simulations were started from the least explored states. As a result, we obtained a total simulation time of 180 μ s with forces and coordinates saved every 100 ps giving a total of 1.8×10^6 frames.

To obtain the training data for the CA model, the initial training set of coordinates and forces was filtered to retain only CA atoms positions and forces. In this example, a coarse-grained system contains 10 beads, built out of seven unique types of beads, one for each amino acid type. The training set for the CACB model as prepared in a similar fashion, filtering both CA and CB atoms and achieving 19 beads and 8 unique types of beads, as all CA atoms was classified as one bead type with the exception of glycine, and each CB was assigned an amino acid-specific bead type. Details of bead selection for both models are described in Supporting Methods.

3.4.2. Neural Network Potential Training. For coarse-grained simulations, it is important to provide some prior (fixed) potentials in order to limit the space that the dynamics can visit to the space sampled in the training data.⁹ All the terms of the force field could be applied, but for simplicity, we limit them to bonds and repulsions. Bonds prevent the protein polymer chain from breaking, and repulsions stop computing NNP on very close atom distances where there is no data.

For pairwise bonded terms, we used the all-atom training data to construct distance histograms for each pair of bonded bead types. Specifically, for each bonded pair, we counted the fraction of time that the respective distance spent in an equally spaced bin in a distance range appropriate to the bead selection, 3.55 and 4.2 Å for all bonds between α carbon beads and 1.3 and 1.8 Å for all bonds between α carbon and β carbon. The distance distributions were Boltzmann-inverted to obtain free-energy profiles, and these were used to fit the equilibrium distance r_0 and the spring constant k of the respective harmonic potential

$$V_{\text{harmonic}}^{\text{(prior)}}(r) = k(r - r_0)^2 + V_0$$

where r is the distance between the beads involved in the bond.

Prior potentials for nonbonded repulsive terms were derived analogously. Distance histograms were constructed with 30 equally spaced distance bins between 3 Å and 6 Å and were used to fit the parameter ϵ of the repulsive potential

$$V_{\text{repulsive}}^{\text{(prior)}}(r) = 4\epsilon r^{-6} + V_0$$

where r , as above, is the distance between the nonbonded beads. In fitting the potential curves, we corrected for the reference state by normalizing counts of each bin by the volume of the corresponding spherical shell. Nonlinear curve fits were performed with the Levenberg–Marquardt method of the SciPy package.⁵¹

The parameters of the prior forces are stored in a YAML force-field file. Plots presenting the quality of fits are included in the Supporting Information (Figures S1–S4) as well as YAML files describing the prior force field.

Based on the resulting prior force field and input coordinates, we calculated a set of prior forces acting on the beads and then deducted them from true forces, resulting in a set of forces that we refer to as delta-forces. Along with coordinates, delta-forces were used as the input for training. In the case of the CA model, embeddings correspond to integers unique for each amino acid type. For the CACB model, all α carbons have the same embedding with the exception of glycine, and each β carbon has an embedding unique for each amino acid type.

The network was trained using a force matching approach, where a predicted force is compared to a true force from the training set.^{8,9} In the example presented here, the network consisted of 3 interaction layers, 128 filters used in continuous-filter convolution, 128 features to describe atomic environments, a 9 Å cutoff radius, and 150 Gaussian functions for the CA model and 300 Gaussians for the CACB model as the basis set of the convolutions filters. Increasing the number of Gaussian functions for the CACB model was found to provide a higher stability of the model and prevent forming collapsed nonphysical structures during the simulation. Models for simulation were selected when the validation loss reached a plateau. The training and validation loss as well as learning rates are presented in Supporting Figure 5.

3.4.3. Simulation of the NNP. The combinations of the force fields covering prior forces and the trained networks are used to simulate both CA and CACB systems with TorchMD. We introduce the parameters of the simulation as a YAML-formatted configuration file (Figure 6), although the simulation can be also started from the command line. The network is introduced to TorchMD as an external force, with the specified network's location, embeddings, and a calculator.


```

1 device: cuda:0
2 log_dir: sim_out_dir
3 output: output
4 topology: cln_ca_top.psf
5 coordinates: cln_ca_init_coords.xtc
6 replicas: 10
7 forcefield: cln_priors.yaml
8 forceterms:
9 - Bonds
10 - RepulsionCG
11 external:
12 module:
13   ↪ torchmdnet.nnp.calculators.torchmdcalc
14   embeddings: [ 4, 4, 5, 8, 6, 13,
15     ↪ 2, 13, 7, 4]
16   file: train/epoch=80.ckpt
17 langevin_gamma: 1
18 langevin_temperature: 350
19 temperature: 350
20 precision: double
21 seed: 1
22 output_period: 1000
23 save_period: 1000
24 steps: 10000000
25 timestep: 1

```

Figure 6. An example of a simulation input file.

An external force calculator class must have a “calculate” method that returns a tuple with energy and forces tensors. In our case, for both models, we run the simulation at 350 K for 10 ns with a 1 fs time step, saving the output every 1000 fs. Note that while the simulations use a small time step, the effective dynamics of the coarse-grained systems is much faster than the all-atom MD system as the coarse-grained model is supposed to reproduce the energetics but with much faster kinetics. Since TorchMD can easily handle parallel dynamics, we concurrently run ten simulations, of which five start from the folded state and five start from unfolded conformations.

The free energy surfaces obtained with a time-lagged independent component analysis (TICA)⁵² for the all-atom baseline simulations and the coarse-grained simulations obtained with TorchMD are presented in Figure 7. The energy landscapes are obtained from binning the configurations over the first two TICA dimensions and computing the average of the equilibrium probability on each bin, obtained by Markov state model analysis of the microstate of each configuration. To support TICA plots, we included plots with RMSD values for the first 2 ns of representative trajectories for both models with different starting points (Figure 8). Plots presenting full trajectories are included in the Supporting Information (Figures S6–S9). Neither SchNet nor prior energy terms can enforce chirality in the system, because they both work purely on the distances between the beads. Therefore, the RMSD plots were supplemented with RMSD values of the trajectory’s mirror image.

Results show that the coarse-grained simulations for both models were able to obtain several folding and unfolding events for chignolin. The energy landscapes for the CA model show that it captured the folded state as a global minimum of energy. The simulations also covers other minima representing unfolded and misfolded states. However, they do not recreate the energy barriers connecting these basins (as expected), which is better seen on the one-dimensional free energy surfaces (Figure S10). The CACB model also detects the global minimum correctly but fails at guessing the free energy of the unfolded region. Overall the simulation is less stable than for the CA model, and the misfolded state minimum is incorrectly located.

4. CONCLUSION

In this paper, we demonstrated TorchMD, a PyTorch-based molecular dynamics engine for biomolecular simulations with machine learning capabilities. We have shown several possible applications ranging from Amber all-atom simulations to end-to-end learning of parameters and finally a coarse-grained neural network potential for protein folding. In particular, building an NNP for protein folding requires supplementing it with asymptotic, analytical potentials for bonds and repulsions to prevent exploring conformations not visited in the training

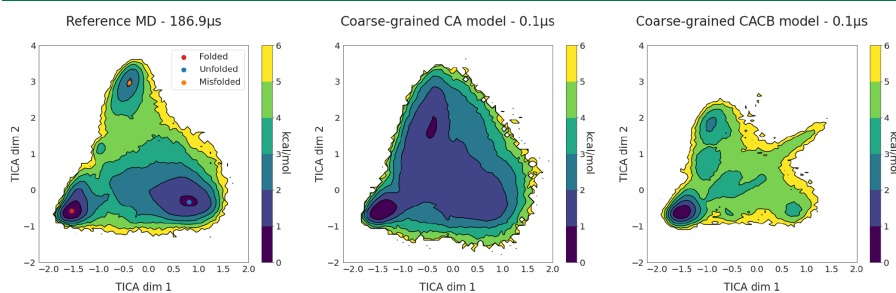


Figure 7. Two-dimensional free energy surfaces for the reference all-atom MD simulations (left) and the two coarse-grained models, CA (center) and CACB (right). The free energy surface for each simulation set was obtained by binning over the first two TICA dimensions, dividing them into a 120×120 grid, and averaging the weights of the equilibrium probability in each bin computed by the Markov state model. The reference MD simulations plot displays the locations of the three energy minima on the surface, corresponding to folded state (red dot), unfolded conformations (blue dot), and a misfolded state (orange dot). Both reference MD and coarse-grained simulations were performed at 350 K.

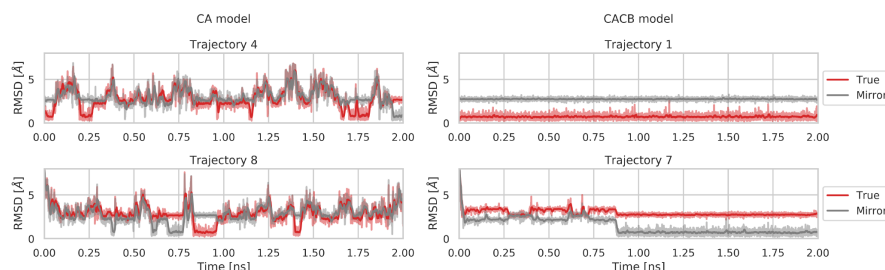


Figure 8. RMSD values across the first 2 ns of the unmodified trajectory (*True*, red) and a mirror image of the original trajectory (*Mirror*, gray) for the CA model (on the left) and the CACB model (on the right). Trajectory 4 (top left panel) and Trajectory 1 (top right panel) are examples of trajectories started from the folded state for the CA model and the CACB model, respectively. Trajectory 8 (bottom left panel) and Trajectory 7 (bottom right panel) are examples of trajectories started from the elongated chain for the CA model and the CACB model, respectively. A moving average of 100 frames is represented as darker lines. The full 10 ns of each simulation is included in Supporting Figures S6–S9.

data in which the predictions of NNP are unreliable. We have shown how to coarse-grain a protein into either α -carbon atoms or α -carbon and β -carbon atoms. Currently, the CA model seems to work the best, but future research will indicate which models are better suited for a more diverse set of targets. TorchMD end-to-end differentiability of its parameters is a feature that projects such as the Open Force Field Initiative⁵³ can potentially exploit. Furthermore, for additional speed, we plan to facilitate the integration of machine learning potentials in OpenMM⁵⁴ and ACEMD³¹ and possibly develop a plugin to extend support to more MD engines in the future. Meanwhile, we believe that TorchMD can play an important role by facilitating experimentation between ML and MD fields, speeding up the model-train-evaluate prototyping cycle, and promoting the adoption of data-based approaches in molecular simulations. All the code machinery to produce the models is made available for practitioners at github.com/torchmd.

■ ASSOCIATED CONTENT

Supporting Information

The Supporting Information is available free of charge at <https://pubs.acs.org/doi/10.1021/acs.jctc.0c01343>.

Details of bead selection for both models (Supporting Methods), plots presenting quality of fits (Figures S1–S4), training and validation loss as well as learning rates (Figure S5), plots presenting full trajectories (Figures S6–S9), and one-dimensional free energy profile (Figure S10) (PDF)

■ AUTHOR INFORMATION

Corresponding Author

Gianni De Fabritiis — *Acellera, 08005 Barcelona, Spain; Computational Science Laboratory, Universitat Pompeu Fabra, 08003 Barcelona, Spain; Institució Catalana de Recerca i Estudis Avançats, 08010 Barcelona, Spain; orcid.org/0000-0003-3913-4877; Email: g.defabritiis@gmail.com*

Authors

Stefan Doerr — *Acellera, 08005 Barcelona, Spain; orcid.org/0000-0002-8678-8657*

Maciej Majewski — *Computational Science Laboratory, Universitat Pompeu Fabra, 08003 Barcelona, Spain; orcid.org/0000-0003-2605-8166*

Adrià Pérez — *Computational Science Laboratory, Universitat Pompeu Fabra, 08003 Barcelona, Spain; orcid.org/0000-0003-2637-1179*

Andreas Krämer — *Department of Mathematics and Computer Science, Freie Universität, 14195 Berlin, Germany; orcid.org/0000-0002-7699-3083*

Cecilia Clementi — *Department of Physics, Freie Universität, 14195 Berlin, Germany; Department of Chemistry, Rice University, Houston 77005, Texas, United States; orcid.org/0000-0001-9221-2358*

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

Toni Giorgino — *Biophysics Institute, National Research Council (CNR-IBF), 20133 Milano, Italy; Department of Biosciences, Università degli Studi di Milano, 20133 Milano, Italy; orcid.org/0000-0001-6449-0596*

Complete contact information is available at: <https://pubs.acs.org/doi/10.1021/acs.jctc.0c01343>

Notes

The authors declare no competing financial interest.

■ ACKNOWLEDGMENTS

S.D. thanks the Chan Zuckerberg initiative for funding. We thank the volunteers of GPUGRID.net for donating computing time. This project has received funding from the European Union’s Horizon 2020 research and innovation programme under grant agreement No. 823712 (CompBioMed2 Project).

■ REFERENCES

- (1) Lee, E. H.; Hsin, J.; Sotomayor, M.; Comellas, G.; Schulten, K. Discovery Through the Computational Microscope. *Structure* **2009**, *17*, 1295–1306.
- (2) Ponder, J. W.; Case, D. A. *Advances in Protein Chemistry; Protein Simulations*; Academic Press: 2003; Vol. 66, pp 27–85, DOI: 10.1016/S0065-3233(03)66002-X.

<https://doi.org/10.1021/acs.jctc.0c01343>
J. Chem. Theory Comput. **2021**, *17*, 2355–2363

- (3) Martínez-Rosell, G.; Giorgino, T.; Harvey, M. J.; de Fabritiis, G. Drug Discovery and Molecular Dynamics: Methods, Applications and Perspective Beyond the Second Timescale. *Current Topics in Medicinal Chemistry* **2017**, *17*, 2617–2625.
- (4) Schütt, K.; Kindermans, P.-J.; Felix, H. E. S.; Chmiela, S.; Tkatchenko, A.; Müller, K.-R. SchNet: A continuous-filter convolutional neural network for modeling quantum interactions. *Advances in neural information processing systems* **2017**, 991–1001.
- (5) 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.
- (6) Ruza, J.; Wang, W.; Schwalbe-Koda, D.; Axelrod, S.; Harris, W. H.; Gómez-Bombarelli, R. Temperature-transferable coarse-graining of ionic liquids with dual graph convolutional neural networks. *J. Chem. Phys.* **2020**, *153*, 164501.
- (7) Duvenaud, D. K.; Maclaurin, D.; Iparraguirre, J.; Bombarell, R.; Hirzel, T.; Aspuru-Guzik, A.; Adams, R. P. Convolutional networks on graphs for learning molecular fingerprints. *Advances in neural information processing systems* **2015**, 2224–2232.
- (8) Husic, B. E.; Charron, N. E.; Lemm, D.; Wang, J.; Pérez, A.; Majewski, M.; Krämer, A.; Chen, Y.; Olsson, S.; de Fabritiis, G.; Noé, F.; et al. Coarse Graining Molecular Dynamics with Graph Neural Networks. *J. Chem. Phys.* **2020**, *153*, 194101.
- (9) 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 central science* **2019**, *5*, 755–767.
- (10) Nüske, F.; Boninsegni, L.; Clementi, C. Coarse-graining molecular systems by spectral matching. *J. Chem. Phys.* **2019**, *151*, 044116.
- (11) Wang, J.; Chmiela, S.; Müller, K.-R.; Noé, F.; Clementi, C. Ensemble learning of coarse-grained molecular dynamics force fields with a kernel approach. *J. Chem. Phys.* **2020**, *152*, 194106.
- (12) Zhang, L.; Han, J.; Wang, H.; Car, R.; E, W. DeePCG: constructing coarse-grained models via deep neural networks. **2018**, arXiv:1802.08549. [arXiv.org e-Print archive. https://arxiv.org/abs/1802.08549](https://arxiv.org/abs/1802.08549) (accessed 2021-03-14).
- (13) Wang, W.; Gómez-Bombarelli, R. Coarse-graining autoencoders for molecular dynamics. *npj Computational Materials* **2019**, *5*, 125.
- (14) Wang, W.; Yang, T.; Harris, W. H.; Gómez-Bombarelli, R. Active learning and neural network potentials accelerate molecular screening of ether-based solvate ionic liquids. *Chem. Commun.* **2020**, *56*, 8920–8923.
- (15) Marrink, S. J.; Tieleman, D. P. Perspective on the Martini model. *Chem. Soc. Rev.* **2013**, *42*, 6801–6822.
- (16) Machado, M. R.; Barrera, E. E.; Klein, F.; Sónora, M.; Silva, S.; Pantano, S. The SIRAH 2.0 Force Field: Altius, Fortius, Citius. *J. Chem. Theory Comput.* **2019**, *15*, 2719–2733.
- (17) Saunders, M. G.; Voth, G. A. Coarse-Graining Methods for Computational Biology. *Annu. Rev. Biophys.* **2013**, *42*, 73–93.
- (18) Izvekov, S.; Voth, G. A. A Multiscale Coarse-Graining Method for Biomolecular Systems. *J. Phys. Chem. B* **2005**, *109*, 2469–2473.
- (19) Noid, W. G. Perspective: Coarse-grained models for biomolecular systems. *J. Chem. Phys.* **2013**, *139*, 090901.
- (20) Clementi, C. Coarse-grained models of protein folding: Toy-models or predictive tools? *Curr. Opin. Struct. Biol.* **2008**, *18*, 10–15.
- (21) Paszke, A.; Gross, S.; Massa, F.; Lerer, A.; Bradbury, J.; Chanan, G.; Killeen, T.; Lin, Z.; Gimelshein, N.; Antiga, L.; Desmaison, A.; Kopf, A.; Yang, E.; DeVito, Z.; Raison, M.; Tejani, A.; Chilamkurthy, S.; Steiner, B.; Fang, L.; Bai, J.; Chintala, S. In *Advances in Neural Information Processing Systems* 32; Wallach, H., Larochelle, H., Beygelzimer, A., d'Alché-Buc, F., Fox, E., Garnett, R., Eds.; Curran Associates, Inc.: 2019; pp 8024–8035.
- (22) Gao, X.; Ramezanghorbani, F.; Isayev, O.; Smith, J.; Roitberg, A. TorchANI: A Free and Open Source PyTorch Based Deep Learning Implementation of the ANI Neural Network Potentials. **2020**, *ChemRxiv*. https://chemrxiv.org/articles/preprint/TorchANI_A_Free_and_Open_Source_PyTorch_Based_Deep_Learning_Implementation_of_the_ANI_Neural_Network_Potentials/12218294 (accessed 2021-03-14).
- (23) Wang, Y.; Fass, J.; Chodera, J. D. End-to-End Differentiable Molecular Mechanics Force Field Construction. **2020**, preprint arXiv:2010.01196. *arXiv.org e-Print archive. https://arxiv.org/abs/2010.01196* (accessed 2021-03-14).
- (24) Greener, J. G.; Jones, D. T. Differentiable molecular simulation can learn all the parameters in a coarse-grained force field for proteins. **2021**, *bioRxiv*. <https://www.biorxiv.org/content/10.1101/2021.02.05.429941v1> (accessed 2021-03-14).
- (25) Bradbury, J.; Frostig, R.; Hawkins, P.; Johnson, M. J.; Leary, C.; Maclaurin, D.; Wanderman-Milne, S. JAX: composable transformations of Python+NumPy programs. **2018**, *GitHub*. <http://github.com/google/jax> (accessed 2021-03-14).
- (26) Schoenholz, S. S.; Cubuk, E. D. JAX M. D. End-to-End Differentiable, Hardware Accelerated, Molecular Dynamics in Pure Python. **2019**, <https://arxiv.org/abs/1912.04232> (accessed 2021-03-16).
- (27) Zhao, Y. Time Machine. **2020**, *GitHub*. <https://github.com/proteneer/timemachine> (accessed 2021-03-14).
- (28) Yao, K.; Herr, J. E.; Toth, D. W.; Mckintyre, R.; Parkhill, J. The TensorMol-0.1 model chemistry: a neural network augmented with long-range physics. *Chem. Sci.* **2018**, *9*, 2261–2269.
- (29) Schütt, K. T.; Kessel, P.; Gastegger, M.; Nicoli, K. A.; Tkatchenko, A.; Müller, K.-R. SchNetPack: A Deep Learning Toolbox For Atomistic Systems. *J. Chem. Theory Comput.* **2019**, *15*, 448–455.
- (30) Tironi, I. G.; Sperb, R.; Smith, P. E.; van Gunsteren, W. F. A generalized reaction field method for molecular dynamics simulations. *J. Chem. Phys.* **1995**, *102*, 5451–5459.
- (31) Harvey, M. J.; Giupponi, G.; Fabritiis, G. D. ACEMD: accelerating biomolecular dynamics in the microsecond time scale. *J. Chem. Theory Comput.* **2009**, *5*, 1632–1639.
- (32) Shirts, M. R.; Klein, C.; Swails, J. M.; Yin, J.; Gilson, M. K.; Mobley, D. L.; Case, D. A.; Zhong, E. D. Lessons learned from comparing molecular dynamics engines on the SAMPL5 dataset. *J. Comput.-Aided Mol. Des.* **2017**, *31*, 147–161.
- (33) Maier, J. A.; Martinez, C.; Kasavajhala, K.; Wickstrom, L.; Hauser, K. E.; Simmerling, C. ff14SB: Improving the Accuracy of Protein Side Chain and Backbone Parameters from ff99SB. *J. Chem. Theory Comput.* **2015**, *11*, 3696–3713.
- (34) Christensen, A. S.; Bratholm, L. A.; Faber, F. A.; Anatole von Lilienfeld, O. FCHL revisited: Faster and more accurate quantum machine learning. *J. Chem. Phys.* **2020**, *152*, 044107.
- (35) Falcon, W. PyTorch Lightning. **2019**, *GitHub*. <https://github.com/PyTorchLightning/pytorch-lightning> (accessed 2021-03-14).
- (36) Ramakrishnan, R.; Dral, P. O.; Rupp, M.; von Lilienfeld, O. A. Quantum chemistry structures and properties of 134 kilo molecules. *Scientific Data* **2014**, *1*, 140022.
- (37) Honda, S.; Akiba, T.; Kato, Y. S.; Sawada, Y.; Sekijima, M.; Ishimura, M.; Ooishi, A.; Watanabe, H.; Odahara, T.; Harata, K. Crystal structure of a ten-amino acid protein. *J. Am. Chem. Soc.* **2008**, *130*, 15327–15331.
- (38) Kingma, D. P.; Ba, J. Adam: A method for stochastic optimization. **2014**, preprint arXiv:1412.6980. *arXiv.org e-Print archive. https://arxiv.org/abs/1412.6980* (accessed 2021-03-14).
- (39) Vapnik, V. *The nature of statistical learning theory*; Springer Science & Business Media: 2013; DOI: 10.1007/978-1-4757-3264-1.
- (40) Lindorff-Larsen, K.; Piana, S.; Dror, R. O.; Shaw, D. E. How fast-folding proteins fold. *Science* **2011**, *334*, 517–520.
- (41) Beauchamp, K. A.; McGibbon, R.; Lin, Y.-S.; Pande, V. S. Simple few-state models reveal hidden complexity in protein folding. *Proc. Natl. Acad. Sci. U. S. A.* **2012**, *109*, 17807–17813.
- (42) 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.
- (43) McKiernan, K. A.; Husic, B. E.; Pande, V. S. Modeling the mechanism of CLN025 beta-hairpin formation. *J. Chem. Phys.* **2017**, *147*, 104107.

- (44) Sultan, M. M.; Pande, V. S. Automated design of collective variables using supervised machine learning. *J. Chem. Phys.* **2018**, *149*, 094106.
- (45) Scherer, M. K.; Husic, B. E.; Hoffmann, M.; Paul, F.; Wu, H.; Noé, F. Variational selection of features for molecular kinetics. *J. Chem. Phys.* **2019**, *150*, 194108.
- (46) Buch, L.; Harvey, M. J.; Giorgino, T.; Anderson, D. P.; De Fabritiis, G. High-throughput all-atom molecular dynamics simulations using distributed computing. *J. Chem. Inf. Model.* **2010**, *50*, 397–403.
- (47) Piana, S.; Lindorff-Larsen, K.; Shaw, D. E. How robust are protein folding simulations with respect to force field parameterization? *Biophys. J.* **2011**, *100*, L47.
- (48) Jorgensen, W. L.; Chandrasekhar, J.; Madura, J. D.; Impey, R. W.; Klein, M. L. Comparison of simple potential functions for simulating liquid water. *J. Chem. Phys.* **1983**, *79*, 926–935.
- (49) Feenstra, K. A.; Hess, B.; Berendsen, H. J. Improving efficiency of large time-scale molecular dynamics simulations of hydrogen-rich systems. *J. Comput. Chem.* **1999**, *20*, 786–798.
- (50) Doerr, S.; De Fabritiis, G. On-the-Fly Learning and Sampling of Ligand Binding by High-Throughput Molecular Simulations. *J. Chem. Theory Comput.* **2014**, *10*, 2064.
- (51) Virtanen, P.; Gommers, R.; Oliphant, T. E.; Haberland, M.; Reddy, T.; Cournapeau, D.; Burovski, E.; Peterson, P.; Weckesser, W.; Bright, J.; van der Walt, S. J.; Brett, M.; Wilson, J.; Millman, K. J.; Mayorov, N.; Nelson, A. R. J.; Jones, E.; Kern, R.; Larson, E.; Carey, C. J.; Polat, I.; Feng, Y.; Moore, E. W.; VanderPlas, J.; Laxalde, D.; Perktold, J.; Cimrman, R.; Henriksen, I.; Quintero, E. A.; Harris, C. R.; Archibald, A. M.; Ribeiro, A. H.; Pedregosa, F.; van Mulbregt, P. SciPy 1.0: fundamental algorithms for scientific computing in Python. *Nat. Methods* **2020**, *17*, 261–272.
- (52) 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.
- (53) Mobley, D. L.; Bannan, C. C.; Rizzi, A.; Bayly, C. I.; Chodera, J. D.; Lim, V. T.; Lim, N. M.; Beauchamp, K. A.; Shirts, M. R.; Gilson, M. K.; Eastman, P. K. Open Force Field Consortium: Escaping atom types using direct chemical perception with SMIRNOFF v0.1. 2018, *Biorxiv preprint*. <https://www.biorxiv.org/content/10.1101/286542v2> (accessed 2021-03-14), DOI: 10.1101/286542.
- (54) 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*, No. e1005659.

Supplementary Information

TorchMD: A deep learning framework for molecular simulations

Stefan Doerr,[†] Maciej Majewski,[‡] Adrià Pérez,[‡] Andreas Krämer,[¶] Cecilia
Clementi,^{§,||} Frank Noe,^{¶,§,||} Toni Giorgino,^{⊥, #} and Gianni De Fabritiis^{*, †, ‡, @}

[†]*Acellera, Barcelona, Spain*

[‡]*Computational Science Laboratory, Universitat Pompeu Fabra, Barcelona, Spain*

[¶]*Department of Mathematics and Computer Science, Freie Universität, Berlin, Germany*

[§]*Department of Physics, Freie Universität, Berlin, Germany*

^{||}*Department of Chemistry, Rice University, Houston, Texas*

[⊥]*Biophysics Institute, National Research Council (CNR-IBF), Italy*

[#]*Department of Biosciences, Università degli Studi di Milano, Italy*

[@]*Institució Catalana de Recerca i Estudis Avançats, Barcelona, Spain*

E-mail: g.defabritiis@gmail.com

Supporting Methods

Coarse-Graining

Coarse-grained models were constructed based on only α -carbons (CA model) or both α - and β -carbons (CACB model) of chignolin. For CA model each α -carbon was assigned a bead_name based on the amino acid of origin, resulting in 7 unique bead types (Tab.S1). For CACB model all α -carbons, except for glycine, were classified as the same bead type and each β -carbon was assigned a bead_name based on the amino acid of origin, resulting in the total of 8 unique bead types (Tab.S2). These selections of beads were then used to filter the coordinates and forces from full-atomistic simulations and to compute parameters of the prior energy terms in the simulation force field. Each bead_name has an embedding associated with it that is later used as an input for the neural network.

Table S1: A set of coarse-grained beads building CA model, along with the embeddings required for the network.

index	atom_name	residue	bead_name	embedding
1	CA	TYR	CAY	4
2	CA	TYR	CAY	4
3	CA	ASP	CAD	5
4	CA	PRO	CAP	8
5	CA	GLU	CAE	6
6	CA	THR	CAT	13
7	CA	GLY	CAG	2
8	CA	THR	CAT	13
9	CA	TRP	CAW	7
10	CA	TYR	CAY	4

Table S2: A set of coarse-grained beads building CACB model, along with the embeddings required for the network.

index	atom_name	residue	bead_name	embedding
1	CA	TYR	CA	30
2	CB	TYR	CBY	4
3	CA	TYR	CA	30
4	CB	TYR	CBY	4
5	CA	ASP	CA	30
6	CB	ASP	CBD	5
7	CA	PRO	CA	30
8	CB	PRO	CBP	8
9	CA	GLU	CA	30
10	CB	GLU	CBE	6
11	CA	THR	CA	30
12	CB	THR	CBT	13
13	CA	GLY	CAG	31
14	CA	THR	CA	30
15	CB	THR	CBT	13
16	CA	TRP	CA	30
17	CB	TRP	CBW	7
18	CA	TYR	CA	30
19	CB	TYR	CBY	4

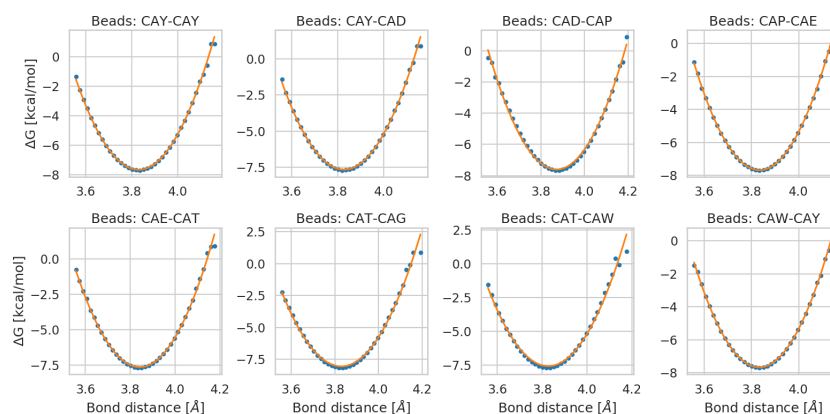


Figure S1: Each plot presents a fit of a function representing harmonic potential (orange line) to a free energy profile obtained from the distribution of distances of bonded α -carbon beads in full atom training data (blue dots). The fits were used to obtain force field parameters for bonded interactions in CA model.

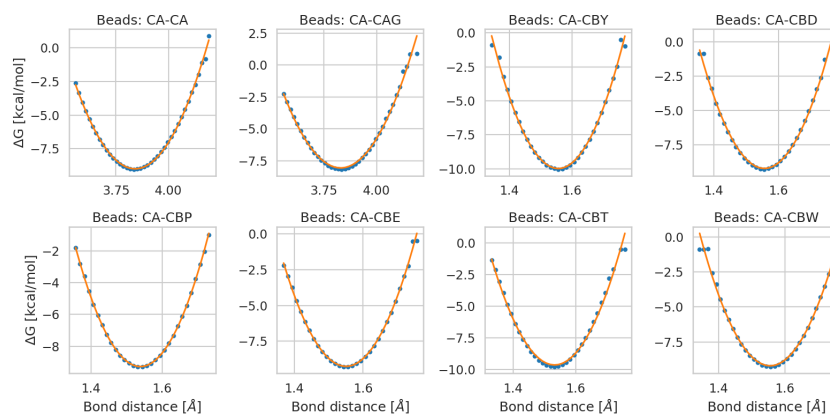


Figure S2: Each plot presents a fit of a function representing harmonic potential (orange line) to a free energy profile obtained from the distribution of distances of bonded α -carbon and β -carbon beads in full atom training data (blue dots). The fits were used to obtain force field parameters for bonded interactions in CACB model.

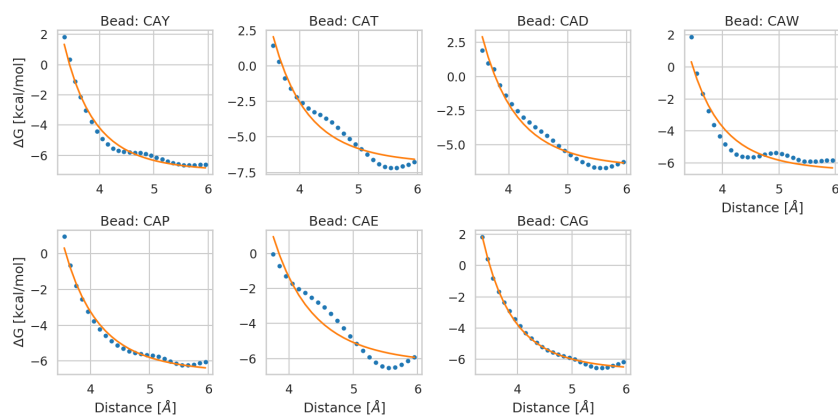


Figure S3: Each plot presents a fit of a function representing repulsive potential (orange line) to a free energy profile obtained from the distribution of distances between a given α -carbon atom and all non-bonded beads in full atom training data (blue dots). The fits were used to obtain force field parameters for non-bonded interactions in CA model.

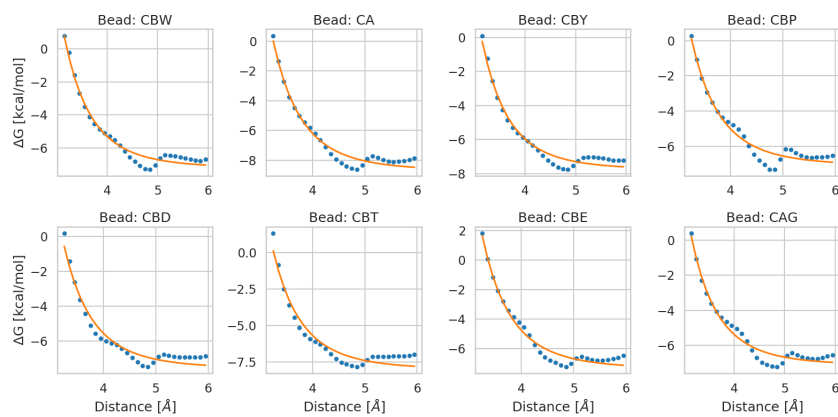


Figure S4: Each plot presents a fit of a function representing repulsive potential (orange line) to a free energy profile obtained from the distribution of distances between a given α -carbon or β -carbon atom and all non-bonded beads in full atom training data (blue dots). The fits were used to obtain force field parameters for non-bonded interactions in CACB model.

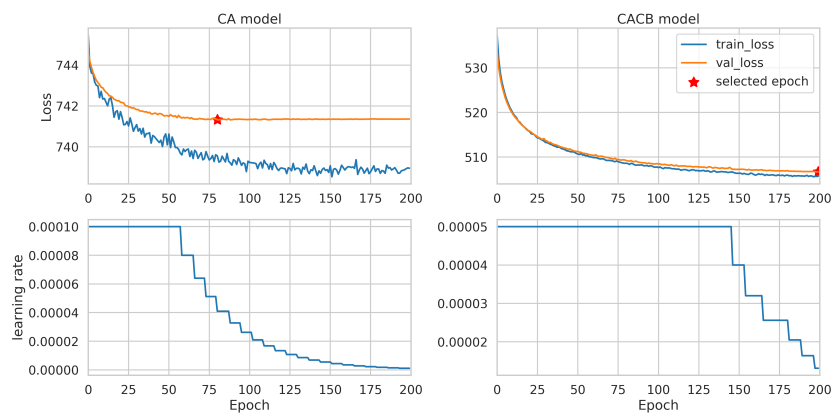


Figure S5: Top panel: Loss curves for CA model (left) and CACB model (right) of chignolin. Blue curves represent training loss values (*train_loss*) and orange curves represent validation loss values (*val_loss*). The models selected are marked with a red star. Bottom panel: learning rate values across the training of the corresponding models.

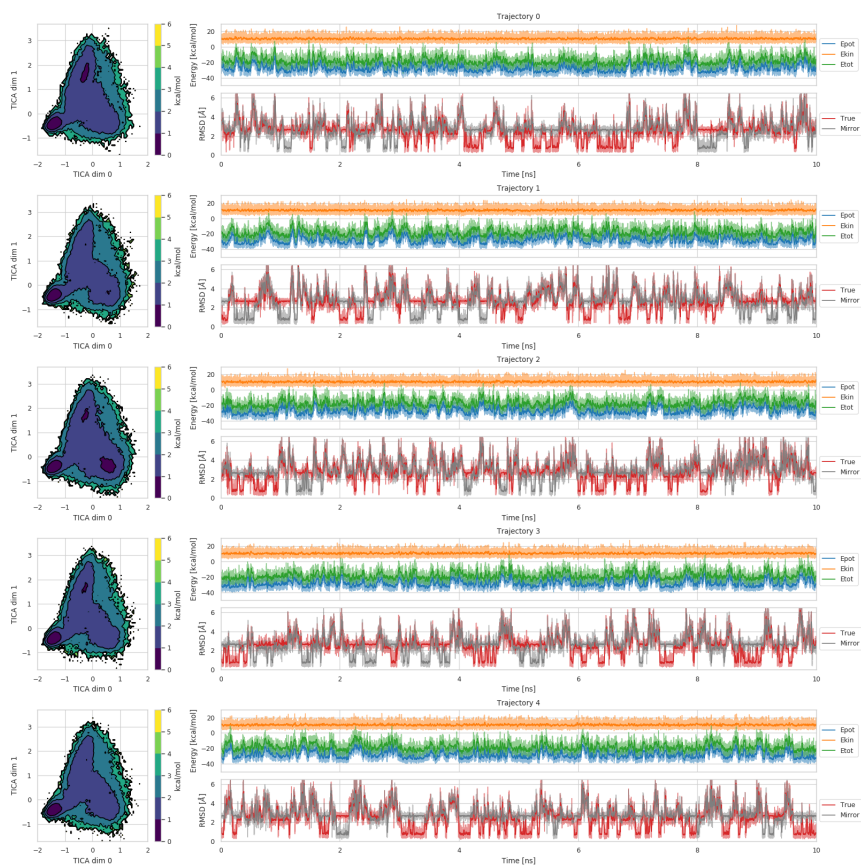


Figure S6: Full trajectories simulated with neural network for CA model starting from folded conformation. Each one of five panels represents a full analysis of energy (top right plot) and RMSD (bottom right plot) across the 10 ns simulation. The energy plot presents potential energy (E_{pot} , blue), kinetic energy (E_{kin} , orange) and total energy (E_{tot} , green). The RMSD plot presents RMSD values across the simulation for the unmodified trajectory ($True$, red) and a mirror image of the original trajectory ($Mirror$, gray). A moving average of 100 frames is represented as darker lines. The left plot on each panel presents a two-dimensional free energy surfaces for the trajectory.

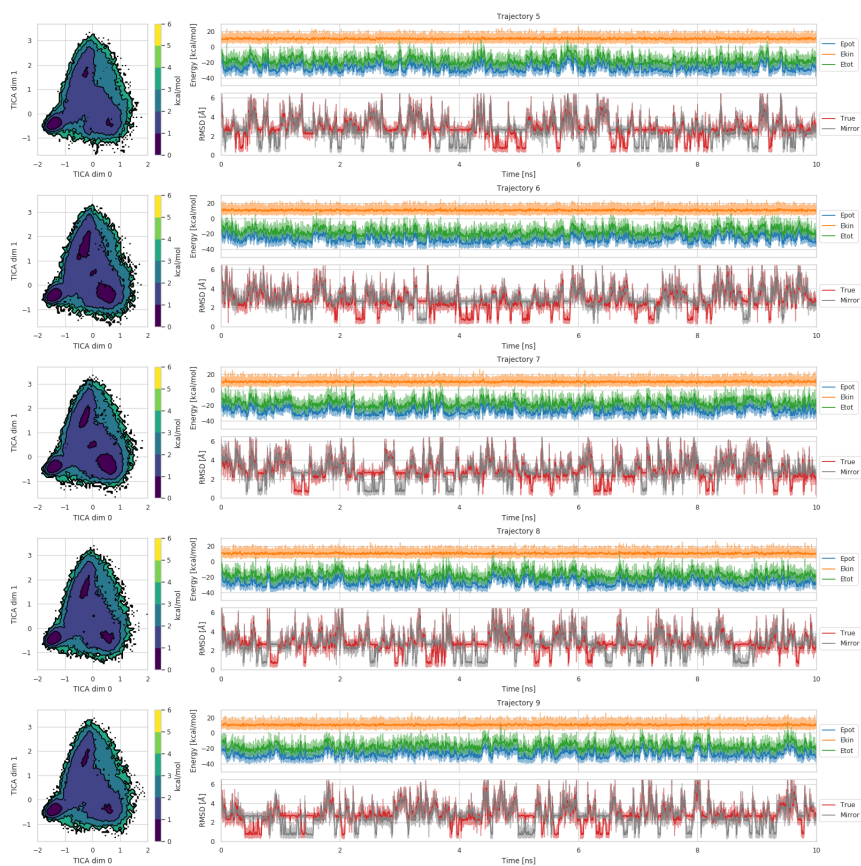


Figure S7: Full trajectories simulated with neural network for CA model starting from an elongated chain. Each one of five panels represents a full analysis of energy (top right plot) and RMSD (bottom right plot) across the 10 ns simulation. The energy plot presents potential energy (E_{pot} , blue), kinetic energy (E_{kin} , orange) and total energy (E_{tot} , green). The RMSD plot presents RMSD values across the simulation for the unmodified trajectory (*True*, red) and a mirror image of the original trajectory (*Mirror*, gray). A moving average of 100 frames is represented as darker lines. The left plot on each panel presents a two-dimensional free energy surfaces for the trajectory.

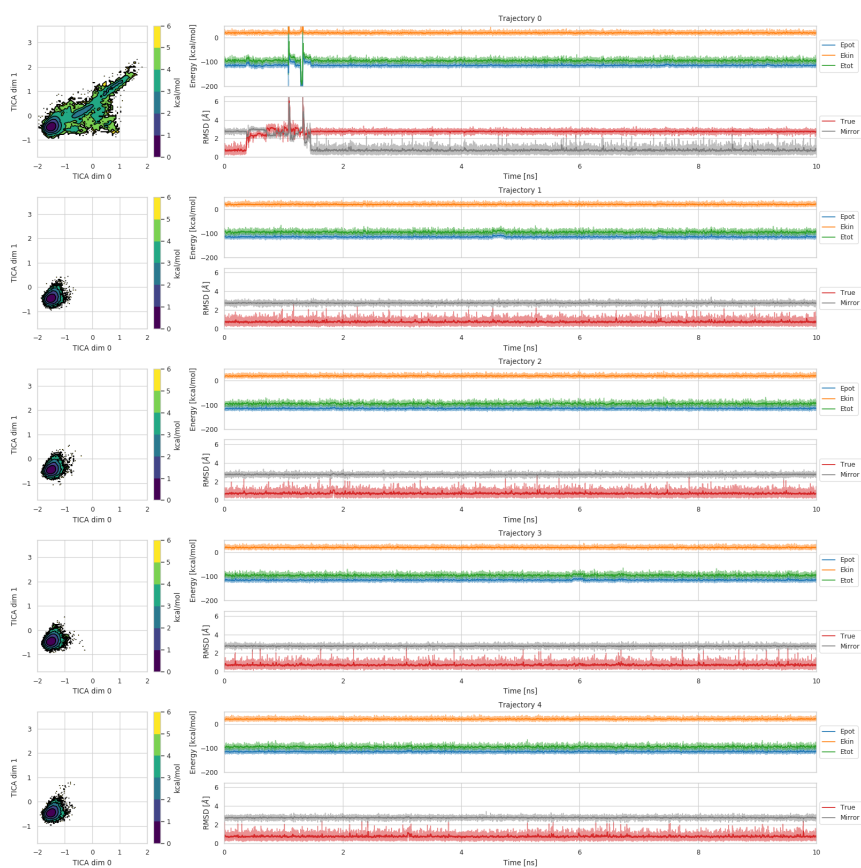


Figure S8: Full trajectories simulated with neural network for CACB model starting from folded conformation. Each one of five panels represents a full analysis of energy (top right plot) and RMSD (bottom right plot) across the 10 ns simulation. The energy plot presents potential energy (E_{pot} , blue), kinetic energy (E_{kin} , orange) and total energy (E_{tot} , green). The RMSD plot presents RMSD values across the simulation for the unmodified trajectory (*True*, red) and a mirror image of the original trajectory (*Mirror*, gray). A moving average of 100 frames is represented as darker lines. The left plot on each panel presents a two-dimensional free energy surfaces for the trajectory.

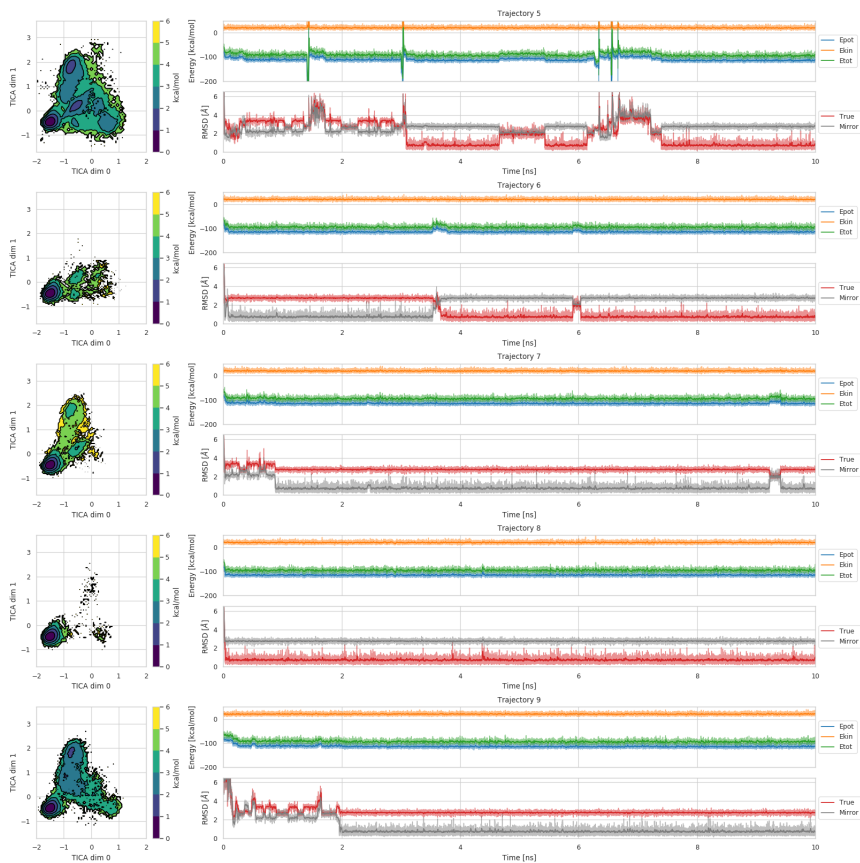


Figure S9: Full trajectories simulated with neural network for CACB model starting from an elongated chain. Each one of five panels represents a full analysis of energy (top right plot) and RMSD (bottom right plot) across the 10 ns simulation. The energy plot presents potential energy (E_{pot} , blue), kinetic energy (E_{kin} , orange) and total energy (E_{tot} , green). The RMSD plot presents RMSD values across the simulation for the unmodified trajectory ($True$, red) and a mirror image of the original trajectory ($Mirror$, gray). A moving average of 100 frames is represented as darker lines. The left plot on each panel presents a two-dimensional free energy surfaces for the trajectory.

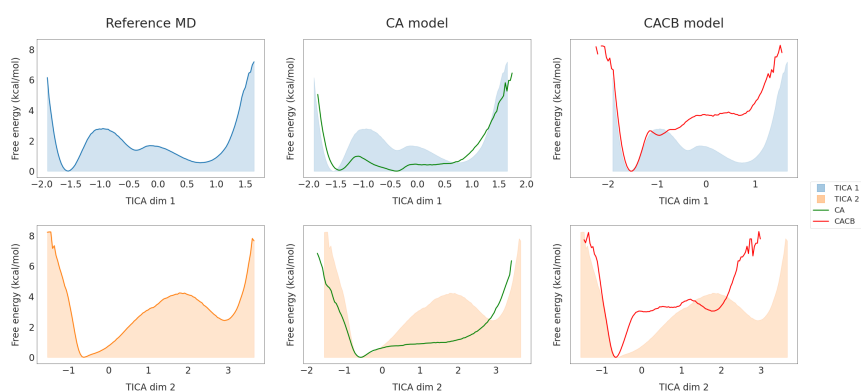


Figure S10: One-dimensional free energy surfaces of the first two TICA dimensions for the reference MD simulations (left), CA model (center) and CACB model (right). First row corresponds to the first TICA dimension while the second row corresponds to the second TICA dimension. The plot was performed the same way as the two-dimensional one (Fig.6), but only binning a single dimension. Coarse-grained models surface lines are depicted with a specific color (green for CA model, red for CACB model). Surfaces for the reference MD simulations are displayed at each plot as a shade for comparison with the coarse-grained surfaces.

Chapter 4

DISCUSSION

In the work presented in this thesis, we have successfully applied a data-driven approach for MD simulations, where we leveraged the latest machine learning advances to learn from simulation data. By using a reinforcement learning approach, we provide a strong framework for adaptive sampling algorithms, which enabled us to characterize a slow and complex event, the coupled binding and folding of the disordered protein cMyb with the KIX domain. We have also designed TorchMD, a deep learning framework to perform and interact with MD simulations. We used it to train and simulate successfully a coarse-grained potential for protein folding. Here we discuss the implications and future prospects of the previously presented work.

4.1 AdaptiveBandit

The results of the various tests performed show that AdaptiveBandit is a flexible algorithm, performing as good as fully exploratory methods even in exploration-limited scenarios and better in others. The 2D toy models were important for the development and validation of the algorithm. The experiments comparing AdaptiveBandit with other algorithms using external knowledge of the system show an interesting comparison between the different strategies, where we can appreciate different be-

haviours and results regarding the exploration versus exploitation decisions made. One of the interesting conclusions we can extract from these results is that no single algorithm is going to perform optimally for all types of systems. By providing a more flexible approach like AdaptiveBandit, we can adapt to different types of systems and sampling requirements.

By recasting adaptive sampling in terms of a multi-armed bandit problem, we can bring a more solid approach to the adaptive sampling problem. The key factor of AdaptiveBandit is that it defines an intrinsic reward from the simulations, not a predefined one. Rewards are computed based on the outcomes of actions instead of being fixed throughout the run. This is useful if, for example, the system gets stuck in a local minima. An external, static reward will keep wasting resources there, and the action will still be valued equally, while AdaptiveBandit is able to update on-the-fly the estimated reward for a certain action. This is exemplified in the toy models using external knowledge and very clearly on the villin experiment. However, AdaptiveBandit is highly sensible to the MSM estimation. If the MSM used at each action-picking stage is poorly estimated, it will influence the action-picking strategy and affect its performance. Moreover, AdaptiveBandit’s performance at its current state also depends on the c hyperparameter, which is not very intuitive to define. Overall, the essential result of AdaptiveBandit is the successful establishment of the adaptive sampling algorithm under a robust mathematical framework, with proofs of optimality, which hopefully will inspire and incentivize better algorithms, more prepared for specific types of systems and MD experiments.

With the simulations for KIX and cMyb, we bring AdaptiveBandit to a real case application. This system features a complex binding event coupled with the folding process of the disordered peptide cMyb. We started simulating the system way before we developed AdaptiveBandit, and we did not have any success, being unable to sample even conformations close to the bound pose. It is a solid proof of AdaptiveBandit’s power to be able to recover the entire folding-and-binding event. By means of MSM modelling, we are able to describe the whole binding and folding

process with detail, identifying the intermediate states and the flux between them. Results show a combined mechanism between induced fit and conformational selection. While an initial folding of the cMyb peptide is required for the first binding to occur, the complete formation of the α -helix is only possible when bound to KIX. We are also able to detect secondary binding modes, some entirely new.

The constructed model is validated by comparing the predicted thermodynamic and kinetic parameters with experimental values. However, some of the predictions are dependant on the manual bulk state definition. We believe there might be several factors for that. First, it can be a pure discretization problem. Because the whole process contains a lot of degrees of freedom, as cMyb can have a lot of different conformations if compared to a small molecule, its hard to get the optimal features and a correct discretization of the bound and unbound states. Second, the simulation box used might have been too small, and therefore its hard for the model to automatically separate the bulk state.

4.2 TorchMD

With TorchMD, we provide an accessible framework to train machine learning based potentials and perform MD simulations with them. Because the simulation code is entirely written in PyTorch, it facilitates the interface between simulations and any deep learning model based on the same library. This is very useful when developing neural network potentials due to the constant iteration between training, testing and adjusting the model’s architecture or hyperparameters. TorchMD has already proven to be very useful inside our group, where we are using it to develop a coarse-grained potential for protein folding, testing new architectures for neural network potentials for MD, or evaluating the potential of end-to-end differential simulations to re-train existing potentials. The only caveat of the current TorchMD implementation is its simulation speed when compared to common MD simulation codes, mostly due to the lack of neighbor lists for non-bonded interactions and the generic

nature of PyTorch operations. TorchMD was not made with speed as an objective, and that is left for future work on machine learning potentials integration with specialized MD codes.

Chapter 5

CONCLUSIONS

1. AdaptiveBandit is able to perform equally or better than previous adaptive sampling algorithms in a diverse set of systems, and it has demonstrated the ability to learn from simulation results.
2. We were able to simulate the binding and folding of c-Myb with KIX by using unbiased full-atom MD simulations, where AdaptiveBandit had a crucial role in resolving this type of folding and binding process.
3. Analysis of the simulation data provides a detailed molecular description of binding of c-Myb to the primary interface of KIX, summarized as a two-step process, where initially the N-terminal region of c-Myb binds to KIX via conformational selection, and finally folds into an α -helix by binding of the C-terminal via induced-fit.
4. The deep learning framework provided by TorchMD can play an important role in facilitating experimentation between machine learning and MD fields, promoting the adoption of data-based approaches in molecular simulations.
5. We have shown several possible applications for TorchMD, ranging from all-atom MD simulations, end-to-end learning of parameters,

training of neural network potentials and finally a coarse-grained neural network potential for protein folding.

Chapter 6

APPENDIX: OTHER PUBLICATIONS

In this chapter you can find other publications where I had minor contributions, but that are still relevant to the thesis.

6.1 Machine Learning of Coarse-Grained Molecular Dynamics Force Fields

Wang J, Olsson S, Wehmeyer C, Pérez A, Charron NE, De Fabritiis G, Noé F, Clementi C. [Machine Learning of Coarse-Grained Molecular Dynamics Force Fields](#) ACS Central Science. 2019;5(5):755-767.

Summary

In this paper, we introduce CGnet, a deep learning based coarse-grained potential trained by a force-matching scheme. We demonstrate that CGnet can reproduce the coarse-grained mean force and free energy for a 2D toy model, alanine dipeptide and the fast-folding protein chignolin. CGnet is able to capture all-atom explicit-solvent free energy surfaces with a potential defined with only a few coarse-grained beads and

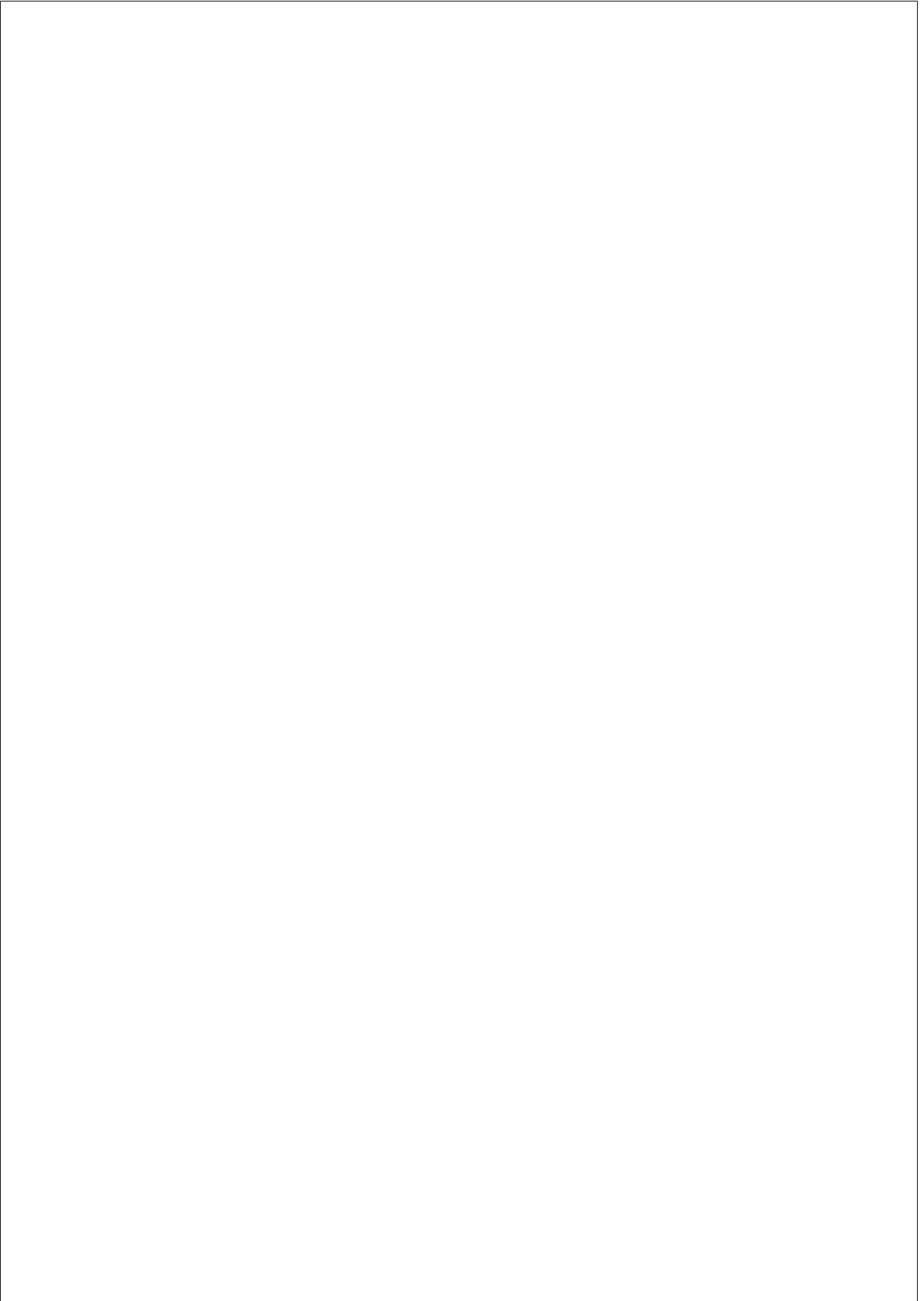
no solvent, while classical coarse-graining methods fail to capture crucial features of the free energy surface.

6.2 Coarse graining molecular dynamics with graph neural networks

Husic BE, Charron NE, Lemm D, Wang J, Pérez A, Majewski M, Krämer A, Chen Y, Olsson S, De Fabritiis G, Noé F, Clementi C. [Coarse graining molecular dynamics with graph neural networks](#). *The Journal of Chemical Physics*. 2021;153(19):194101.

Summary

In this paper, we leverage the work performed in Publication 6.1 and the inherently transferable SchNet [112] to bring up CGSchNet, a transferable coarse-grained potential that is able to learn their own features.



Bibliography

- [1] Nelson DL, Lehninger AL, Cox MM. *Lehninger principles of biochemistry*. Macmillan; 2008.
- [2] Ross CA, Poirier MA. Protein aggregation and neurodegenerative disease. *Nature medicine*. 2004;10(7):S10–S17.
- [3] Bloemendal H, de Jong W, Jaenicke R, Lubsen NH, Slingsby C, Tardieu A. Ageing and vision: structure, stability and function of lens crystallins. *Progress in Biophysics and Molecular Biology*. 2004;86(3):407–485.
- [4] Kendrew JC, Bodo G, Dintzis HM, Parrish R, Wyckoff H, Phillips DC. A three-dimensional model of the myoglobin molecule obtained by x-ray analysis. *Nature*. 1958;181(4610):662–666.
- [5] Chandonia JM, Brenner SE. The impact of structural genomics: expectations and outcomes. *Science*. 2006;311(5759):347–351.
- [6] McCammon J. Protein dynamics. *Reports on Progress in Physics*. 1984;47(1):1.
- [7] Henzler-Wildman K, Kern D. Dynamic personalities of proteins. *Nature*. 2007;450(7172):964–972.
- [8] Frauenfelder H, Sligar SG, Wolynes PG. The energy landscapes and motions of proteins. *Science*. 1991;254(5038):1598–1603.

- [9] Diez M, Zimmermann B, Börsch M, König M, Schweinberger E, Steigmiller S, et al. Proton-powered subunit rotation in single membrane-bound F₀F₁-ATP synthase. *Nature Structural & Molecular Biology*. 2004;11(2):135–141.
- [10] Eisenmesser EZ, Millet O, Labeikovsky W, Korzhnev DM, Wolf-Watz M, Bosco DA, et al. Intrinsic dynamics of an enzyme underlies catalysis. *Nature*. 2005;438(7064):117–121.
- [11] Wright PE, Dyson HJ. Intrinsically unstructured proteins: reassessing the protein structure-function paradigm. *Journal of Molecular Biology*. 1999;293(2):321–331.
- [12] van Der Lee R, Buljan M, Lang B, Weatheritt RJ, Daughdrill GW, Dunker AK, et al. Classification of intrinsically disordered regions and proteins. *Chemical Reviews*. 2014;114(13):6589–6631.
- [13] Romero P, Obradovic Z, Kissinger C, Villafranca J, Dunker A. Identifying disordered regions in proteins from amino acid sequence. In: *Proceedings of International Conference on Neural Networks (ICNN'97)*. vol. 1. IEEE; 1997. p. 90–95.
- [14] Dunker AK, Lawson JD, Brown CJ, Williams RM, Romero P, Oh JS, et al. Intrinsically disordered protein. *Journal of Molecular Graphics and Modelling*. 2001;19(1):26–59.
- [15] Iakoucheva LM, Brown CJ, Lawson JD, Obradović Z, Dunker AK. Intrinsic disorder in cell-signaling and cancer-associated proteins. *Journal of Molecular Biology*. 2002;323(3):573–584.
- [16] Liu J, Perumal NB, Oldfield CJ, Su EW, Uversky VN, Dunker AK. Intrinsic disorder in transcription factors. *Biochemistry*. 2006;45(22):6873–6888.
- [17] Kussie PH, Gorina S, Marechal V, Elenbaas B, Moreau J, Levine AJ, et al. Structure of the MDM2 oncoprotein bound

to the p53 tumor suppressor transactivation domain. *Science*. 1996;274(5289):948–953.

- [18] Radhakrishnan I, Pérez-Alvarado GC, Parker D, Dyson HJ, Montminy MR, Wright PE. Solution structure of the KIX domain of CBP bound to the transactivation domain of CREB: a model for activator: coactivator interactions. *Cell*. 1997;91(6):741–752.
- [19] De Guzman RN, Martinez-Yamout MA, Dyson HJ, Wright PE. Interaction of the TAZ1 Domain of the CREB-Binding Protein with the Activation Domain of CITED2 Regulation by competition between intrinsically unstructured ligand for non-identical binding sites. *Journal of Biological Chemistry*. 2004;279(4):3042–3049.
- [20] Zor T, De Guzman RN, Dyson HJ, Wright PE. Solution structure of the KIX domain of CBP bound to the transactivation domain of c-Myb. *Journal of Molecular Biology*. 2004;337(3):521–534.
- [21] De Guzman RN, Goto NK, Dyson HJ, Wright PE. Structural basis for cooperative transcription factor binding to the CBP coactivator. *Journal of Molecular Biology*. 2006;355(5):1005–1013.
- [22] Bah A, Vernon RM, Siddiqui Z, Krzeminski M, Muhandiram R, Zhao C, et al. Folding of an intrinsically disordered protein by phosphorylation as a regulatory switch. *Nature*. 2015;519(7541):106.
- [23] Krois AS, Ferreon JC, Martinez-Yamout MA, Dyson HJ, Wright PE. Recognition of the disordered p53 transactivation domain by the transcriptional adapter zinc finger domains of CREB-binding protein. *Proceedings of the National Academy of Sciences*. 2016;113(13):E1853–E1862.
- [24] Amaro RE, Mulholland AJ. Multiscale methods in drug design bridge chemical and biological complexity in the search for cures. *Nature Reviews Chemistry*. 2018;2(4):1–12.

- [25] Berman HM, Westbrook J, Feng Z, Gilliland G, Bhat TN, Weissig H, et al. The protein data bank. *Nucleic Acids Research*. 2000;28(1):235–242.
- [26] Levantino M, Yorke BA, Monteiro DC, Cammarata M, Pearson AR. Using synchrotrons and XFELs for time-resolved X-ray crystallography and solution scattering experiments on biomolecules. *Current Opinion in Structural Biology*. 2015;35:41–48.
- [27] Neutze R, Brändén G, Schertler GF. Membrane protein structural biology using X-ray free electron lasers. *Current Opinion in Structural Biology*. 2015;33:115–125.
- [28] Carpenter EP, Beis K, Cameron AD, Iwata S. Overcoming the challenges of membrane protein crystallography. *Current Opinion in Structural Biology*. 2008;18(5):581–586.
- [29] Gibbs EB, Cook EC, Showalter SA. Application of NMR to studies of intrinsically disordered proteins. *Archives of Biochemistry and Biophysics*. 2017;628:57–70.
- [30] Dyson HJ, Wright PE. Perspective: the essential role of NMR in the discovery and characterization of intrinsically disordered proteins. *Journal of Biomolecular NMR*. 2019;73(12):651–659.
- [31] Kühlbrandt W. The resolution revolution. *Science*. 2014;343(6178):1443–1444.
- [32] Nogales E, Scheres SH. Cryo-EM: a unique tool for the visualization of macromolecular complexity. *Molecular Cell*. 2015;58(4):677–689.
- [33] Cheng Y. Single-particle cryo-EM at crystallographic resolution. *Cell*. 2015;161(3):450–457.
- [34] Lyumkis D. Challenges and opportunities in cryo-EM single-particle analysis. *Journal of Biological Chemistry*. 2019;294(13):5181–5197.

- [35] Yip KM, Fischer N, Paknia E, Chari A, Stark H. Atomic-resolution protein structure determination by cryo-EM. *Nature*. 2020;587(7832):157–161.
- [36] Nakane T, Kotecha A, Sente A, McMullan G, Masiulis S, Brown PM, et al. Single-particle cryo-EM at atomic resolution. *Nature*. 2020;587(7832):152–156.
- [37] Okamoto K, Sako Y. Recent advances in FRET for the study of protein interactions and dynamics. *Current Opinion in Structural Biology*. 2017;46:16–23.
- [38] Vestergaard B. Analysis of biostructural changes, dynamics, and interactions—small-angle X-ray scattering to the rescue. *Archives of Biochemistry and Biophysics*. 2016;602:69–79.
- [39] Kikhney AG, Svergun DI. A practical guide to small angle X-ray scattering (SAXS) of flexible and intrinsically disordered proteins. *FEBS letters*. 2015;589(19):2570–2577.
- [40] Grimaldo M, Roosen-Runge F, Zhang F, Schreiber F, Seydel T. Dynamics of proteins in solution. *Quarterly Reviews of Biophysics*. 2019;52.
- [41] Jumper J, Evans R, Pritzel A, Green T, Figurnov M, Ronneberger O, et al. Highly accurate protein structure prediction with AlphaFold. *Nature*. 2021;596(7873):583–589.
- [42] Dirac PAM. Quantum mechanics of many-electron systems. *Proceedings of the Royal Society of London Series A, Containing Papers of a Mathematical and Physical Character*. 1929;123(792):714–733.
- [43] Schrödinger E. An undulatory theory of the mechanics of atoms and molecules. *Physical Review*. 1926;28(6):1049.
- [44] Kohn W, Sham LJ. Self-consistent equations including exchange and correlation effects. *Physical Review*. 1965;140(4A):A1133.

- [45] Parr RG. Density functional theory of atoms and molecules. Springer; 1980.
- [46] Kulik HJ, Luehr N, Ufimtsev IS, Martinez TJ. Ab initio quantum chemistry for protein structures. *The Journal of Physical Chemistry B*. 2012;116(41):12501–12509.
- [47] Ponder JW, Case DA. Force fields for protein simulations. *Advances in Protein Chemistry*. 2003;66:27–85.
- [48] Lindorff-Larsen K, Maragakis P, Piana S, Eastwood MP, Dror RO, Shaw DE. Systematic validation of protein force fields against experimental data. *PloS one*. 2012;7(2):e32131.
- [49] Durrant JD, McCammon JA. Molecular dynamics simulations and drug discovery. *BMC Biology*. 2011;9(1):1–9.
- [50] Noid WG. Perspective: Coarse-grained models for biomolecular systems. *The Journal of Chemical Physics Modelling*. 2013;139(9):09B201_1.
- [51] Marrink SJ, Risselada HJ, Yefimov S, Tieleman DP, De Vries AH. The MARTINI force field: coarse grained model for biomolecular simulations. *The Journal of Physical Chemistry B*. 2007;111(27):7812–7824.
- [52] Monticelli L, Kandasamy SK, Periole X, Larson RG, Tieleman DP, Marrink SJ. The MARTINI coarse-grained force field: extension to proteins. *Journal of Chemical Theory and Computation*. 2008;4(5):819–834.
- [53] Das R, Baker D. Macromolecular modeling with rosetta. *Annu Rev Biochem*. 2008;77:363–382.
- [54] Davtyan A, Schafer NP, Zheng W, Clementi C, Wolynes PG, Papoian GA. AWSEM-MD: protein structure prediction using

- coarse-grained physical potentials and bioinformatically based local structure biasing. *The Journal of Physical Chemistry B*. 2012;116(29):8494–8503.
- [55] Koliński A, et al. Protein modeling and structure prediction with a reduced representation. *Acta Biochimica Polonica*. 2004;51.
- [56] Ołdziej S, Czaplewski C, Liwo A, Chinchio M, Nancias M, Vila J, et al. Physics-based protein-structure prediction using a hierarchical protocol based on the UNRES force field: assessment in two blind tests. *Proceedings of the National Academy of Sciences*. 2005;102(21):7547–7552.
- [57] Rojas AV, Liwo A, Scheraga HA. Molecular dynamics with the united-residue force field: Ab initio folding simulations of multichain proteins. *The Journal of Physical Chemistry B*. 2007;111(1):293–309.
- [58] Periolo X, Knepp AM, Sakmar TP, Marrink SJ, Huber T. Structural determinants of the supramolecular organization of G protein-coupled receptors in bilayers. *Journal of the American Chemical Society*. 2012;134(26):10959–10965.
- [59] Domański J, Marrink SJ, Schäfer LV. Transmembrane helices can induce domain formation in crowded model membranes. *Biochimica et Biophysica Acta (BBA)-Biomembranes*. 2012;1818(4):984–994.
- [60] Sambriski E, Schwartz D, De Pablo J. Uncovering pathways in DNA oligonucleotide hybridization via transition state analysis. *Proceedings of the National Academy of Sciences*. 2009;106(43):18125–18130.
- [61] The Nobel Prize in Chemistry 2013. Nobel Prize Outreach AB; 2013. Available from: <https://www.nobelprize.org/prizes/chemistry/2013/summary/>.

- [62] Rapaport DC, Rapaport DCR. The art of molecular dynamics simulation. Cambridge university press; 2004.
- [63] McCammon JA, Gelin BR, Karplus M. Dynamics of folded proteins. *Nature*. 1977;267(5612):585–590.
- [64] Harvey M, De Fabritiis G. A survey of computational molecular science using graphics processing units. *Wiley Interdisciplinary Reviews: Computational Molecular Science*. 2012;2(5):734–742.
- [65] Shaw DE, Deneroff MM, Dror RO, Kuskin JS, Larson RH, Salmon JK, et al. Anton, a special-purpose machine for molecular dynamics simulation. *Communications of the ACM*. 2008;51(7):91–97.
- [66] Shaw DE, Adams PJ, Azaria A, Bank JA, Batson B, Bell A, et al. Anton 3: twenty microseconds of molecular dynamics simulation before lunch. In: *Proceedings of the International Conference for High Performance Computing, Networking, Storage and Analysis*; 2021. p. 1–11.
- [67] Harvey MJ, Giupponi G, Fabritiis GD. ACEMD: accelerating biomolecular dynamics in the microsecond time scale. *Journal of Chemical Theory and Computation*. 2009;5(6):1632–1639.
- [68] Gotz AW, Williamson MJ, Xu D, Poole D, Le Grand S, Walker RC. Routine microsecond molecular dynamics simulations with AMBER on GPUs. 1. Generalized born. *Journal of Chemical Theory and Computation*. 2012;8(5):1542–1555.
- [69] Salomon-Ferrer R, Gotz AW, Poole D, Le Grand S, Walker RC. Routine microsecond molecular dynamics simulations with AMBER on GPUs. 2. Explicit solvent particle mesh Ewald. *Journal of Chemical Theory and Computation*. 2013;9(9):3878–3888.
- [70] Bowers KJ, Chow DE, Xu H, Dror RO, Eastwood MP, Gregersen BA, et al. Scalable algorithms for molecular dynamics simulations on commodity clusters. In: *SC06: Proceedings of the 2006*

- ACM/IEEE Conference on Supercomputing. IEEE; 2006. p. 43–43.
- [71] Eastman P, Swails J, Chodera JD, McGibbon RT, Zhao Y, Beauchamp KA, et al. OpenMM 7: Rapid development of high performance algorithms for molecular dynamics. *PLoS Computational Biology*. 2017;13(7):e1005659.
- [72] Lindorff-Larsen K, Piana S, Dror RO, Shaw DE. How fast-folding proteins fold. *Science*. 2011;334(6055):517–520.
- [73] Plattner N, Doerr S, De Fabritiis G, Noé F. Complete protein–protein association kinetics in atomic detail revealed by molecular dynamics simulations and Markov modelling. *Nature Chemistry*. 2017;9(10):1005.
- [74] Prinz JH, Wu H, Sarich M, Keller B, Senne M, Held M, et al. Markov models of molecular kinetics: Generation and validation. *The Journal of Chemical Physics Modelling*. 2011;134(17):174105.
- [75] Chodera JD, Noé F. Markov state models of biomolecular conformational dynamics. *Current Opinion in Structural Biology*. 2014;25:135–144.
- [76] Sarich M, Noé F, Schütte C. On the approximation quality of Markov state models. *Multiscale Modeling & Simulation*. 2010;8(4):1154–1177.
- [77] Djurdjevac N, Sarich M, Schütte C. Estimating the eigenvalue error of Markov state models. *Multiscale Modeling & Simulation*. 2012;10(1):61–81.
- [78] Prinz JH, Chodera JD, Noé F. Spectral rate theory for two-state kinetics. *Physical Review X*. 2014;4(1):011020.

- [79] Pérez-Hernández G, Paul F, Giorgino T, De Fabritiis G, Noé F. Identification of slow molecular order parameters for Markov model construction. *The Journal of Chemical Physics Modelling*. 2013;139(1):07B604_1.
- [80] Singhal N, Pande VS. Error analysis and efficient sampling in Markovian state models for molecular dynamics. *The Journal of Chemical Physics modelling*. 2005;123(20):204909.
- [81] Hinrichs NS, Pande VS. Calculation of the distribution of eigenvalues and eigenvectors in Markovian state models for molecular dynamics. *The Journal of Chemical Physics Modelling*. 2007;126(24):244101.
- [82] Pronk S, Larsson P, Pouya I, Bowman GR, Haque IS, Beauchamp K, et al. Copernicus: A new paradigm for parallel adaptive molecular dynamics. In: *Proceedings of 2011 International Conference for High Performance Computing, Networking, Storage and Analysis*. ACM; 2011. p. 60.
- [83] Doerr S, De Fabritiis G. On-the-fly learning and sampling of ligand binding by high-throughput molecular simulations. *Journal of Chemical Theory and Computation*. 2014;10(5):2064–2069.
- [84] Sabbadin D, Moro S. Supervised molecular dynamics (SuMD) as a helpful tool to depict GPCR–ligand recognition pathway in a nanosecond time scale. *Journal of Chemical Information and Modeling*. 2014;54(2):372–376.
- [85] Perez A, MacCallum JL, Dill KA. Accelerating molecular simulations of proteins using Bayesian inference on weak information. *Proceedings of the National Academy of Sciences*. 2015;112(38):11846–11851.
- [86] Zimmerman MI, Bowman GR. FAST conformational searches by balancing exploration/exploitation trade-offs. *Journal of Chemical Theory and Computation*. 2015;11(12):5747–5757.

- [87] Zimmerman MI, Hart KM, Sibbald CA, Frederick TE, Jimah JR, Knoverek CR, et al. Prediction of new stabilizing mutations based on mechanistic insights from Markov state models. *ACS Central Science*. 2017;3(12):1311–1321.
- [88] Cruz MA, Frederick TE, Singh S, Vithani N, Zimmerman MI, Porter JR, et al. Discovery of a cryptic allosteric site in Ebola’s ‘undruggable’ VP35 protein using simulations and experiments. *bioRxiv*. 2020;.
- [89] Huber GA, Kim S. Weighted-ensemble Brownian dynamics simulations for protein association reactions. *Biophysical Journal*. 1996;70(1):97–110.
- [90] Dickson A, Brooks III CL. WExplore: hierarchical exploration of high-dimensional spaces using the weighted ensemble algorithm. *The Journal of Physical Chemistry B*. 2014;118(13):3532–3542.
- [91] Donyapour N, Roussey NM, Dickson A. REVO: Resampling of ensembles by variation optimization. *The Journal of Chemical Physics Modelling*. 2019;150(24):244112.
- [92] Zimmerman MI, Porter JR, Sun X, Silva RR, Bowman GR. Choice of adaptive sampling strategy impacts state discovery, transition probabilities, and the apparent mechanism of conformational changes. *Journal of Chemical Theory and Computation*. 2018;14(11):5459–5475.
- [93] Shamsi Z, Cheng KJ, Shukla D. Reinforcement learning based adaptive sampling: REAPing rewards by exploring protein conformational landscapes. *The Journal of Physical Chemistry B*. 2018;122(35):8386–8395.
- [94] Martinez-Rosell G, Giorgino T, Harvey MJ, de Fabritiis G. Drug discovery and molecular dynamics: methods, applications and perspective beyond the second timescale. *Current Topics in Medicinal Chemistry*. 2017;17(23):2617–2625.

- [95] Glielmo A, Husic BE, Rodriguez A, Clementi C, Noé F, Laio A. Unsupervised Learning Methods for Molecular Simulation Data. *Chemical Reviews*. 2021;.
- [96] LeCun Y, Bengio Y, Hinton G. Deep learning. *Nature*. 2015;521(7553):436–444.
- [97] Schmidhuber J. Deep learning in neural networks: An overview. *Neural Networks*. 2015;61:85–117.
- [98] Krizhevsky A, Sutskever I, Hinton GE. Imagenet classification with deep convolutional neural networks. *Advances in Neural Information Processing Systems*. 2012;25:1097–1105.
- [99] Jozefowicz R, Vinyals O, Schuster M, Shazeer N, Wu Y. Exploring the limits of language modeling. *arXiv preprint arXiv:1602.02410*. 2016;.
- [100] Hornik K. Approximation capabilities of multilayer feedforward networks. *Neural Networks*. 1991;4(2):251–257.
- [101] Andoni A, Panigrahy R, Valiant G, Zhang L. Learning polynomials with neural networks. In: *International Conference on Machine Learning*. PMLR; 2014. p. 1908–1916.
- [102] Noé F, Tkatchenko A, Müller KR, Clementi C. Machine learning for molecular simulation. *Annual Review of Physical Chemistry*. 2020;71:361–390.
- [103] Noé F, De Fabritiis G, Clementi C. Machine learning for protein folding and dynamics. *Current Opinion in Structural Biology*. 2020;60:77–84.
- [104] Noé F, Olsson S, Köhler J, Wu H. Boltzmann generators: Sampling equilibrium states of many-body systems with deep learning. *Science*. 2019;365(6457).

- [105] Schneider E, Dai L, Topper RQ, Drechsel-Grau C, Tuckerman ME. Stochastic neural network approach for learning high-dimensional free energy surfaces. *Physical Review Letters*. 2017;119(15):150601.
- [106] Wu H, Noé F. Variational approach for learning Markov processes from time series data. *Journal of Nonlinear Science*. 2020;30(1):23–66.
- [107] Mardt A, Pasquali L, Wu H, Noé F. VAMPnets for deep learning of molecular kinetics. *Nature Communications*. 2018;9(1):1–11.
- [108] Chmiela S, Tkatchenko A, Sauceda HE, Poltavsky I, Schütt KT, Müller KR. Machine learning of accurate energy-conserving molecular force fields. *Science Advances*. 2017;3(5):e1603015.
- [109] Behler J, Parrinello M. Generalized neural-network representation of high-dimensional potential-energy surfaces. *Physical Review Letters*. 2007;98(14):146401.
- [110] Rupp M, Tkatchenko A, Müller KR, Von Lilienfeld OA. Fast and accurate modeling of molecular atomization energies with machine learning. *Physical Review Letters*. 2012;108(5):058301.
- [111] Smith JS, Isayev O, Roitberg AE. ANI-1: an extensible neural network potential with DFT accuracy at force field computational cost. *Chemical Science*. 2017;8(4):3192–3203.
- [112] Schütt KT, Sauceda HE, Kindermans PJ, Tkatchenko A, Müller KR. Schnet—a deep learning architecture for molecules and materials. *The Journal of Chemical Physics Modelling*. 2018;148(24):241722.
- [113] Schutt K, Kessel P, Gastegger M, Nicoli K, Tkatchenko A, Müller KR. SchNetPack: A deep learning toolbox for atomistic systems. *Journal of Chemical Theory and Computation*. 2018;15(1):448–455.

- [114] Boninsegna L, Banisch R, Clementi C. A data-driven perspective on the hierarchical assembly of molecular structures. *Journal of Chemical Theory and Computation*. 2018;14(1):453–460.
- [115] Wang W, Gómez-Bombarelli R. Coarse-graining auto-encoders for molecular dynamics. *Computational Materials*. 2019;5(1):1–9.