RICE UNIVERSITY

# Adaptive sampling of Conformational Dynamics

by

**Eugen Hruška**

A Thesis Submitted
in Partial Fulfillment of the
Requirements for the Degree

**Doctor of Philosophy**

Approved, Thesis Committee:

Cecilia Clementi, Thesis Director
Professor of Chemistry and Chemical &
Biomolecular Engineering

Jose Onuchic, Chair
Harry C. and Olga K. Wiess Chair of
Physics, Professor of Chemistry and
BioSciences

Jason Hafner
Professor of Physics & Astronomy and
Chemistry

Matteo Pasquali
A. J. Hartsook Professor of Chemical and
Biomolecular Engineering, Chemistry,
Material Science and NanoEngineering

Houston, Texas
April, 2020

# ABSTRACT

## Adaptive sampling of Conformational Dynamics

by

Eugen Hruška

At the core of our limited ability to understand many biophysical processes is the challenge of predicting the conformational dynamics of biomolecules. This challenge includes many open questions around the biophysical causes of many diseases or open questions in biophysics theory. *Adaptive sampling* is an approach to increase our ability to predict conformational dynamics. Adaptive sampling is a class of sampling strategies, where an ensemble of molecular dynamics trajectories is generated, where the starting points for the individual trajectories depend on the previously simulated trajectories. This approach will be investigated in this thesis.

The application of adaptive sampling to biomolecules is one example of the more general problem of accurately sampling the time-dynamics of high-dimensional stochastic systems. The high-dimensionality, combined with a complex energy landscape, impede simpler approaches. Due to the broad scope of the general challenge, this Dissertation will focus only on improving the prediction of conformational dynamics for proteins.

Many previous approaches to unravel this challenge have achieved significant improvements. In the case of proteins, the timescales where we can predict the conformational dynamics have increased by many orders of magnitudes to the millisecond scale. Despite the improvements, the current state-of-art can only predict the ac-

curate behavior for small proteins. This illustrates the magnitude of the challenge. For most of the larger biomolecules, we are not able to simulate the precise behavior. This is not only caused by the several magnitudes longer timescales for these larger systems but also an order of magnitude larger sizes of these biomolecules.

In this thesis, the adaptive sampling of conformational dynamics will be investigated in several steps. First, the prediction of the effectivity of different adaptive sampling strategies will be discussed. Due to significant stochasticity and protein-to-protein variation, the choice of adaptive sampling strategy is not apparent. The performance of different strategies for different goals varies as well.

Second, to deepen our theoretical understanding of adaptive sampling strategies, an upper limit for the performance of any adaptive sampling strategy is developed. This theoretical upper limit allows us to understand the potential and limits of adaptive sampling.

Third, adaptive sampling is heavily dependent on software due to the necessary thousands or millions of individual steps. All these steps have to be executed efficiently on a High-Performance Computer (HPC). Here we show the development of the software package *ExTASY*. This framework allows performing all the necessary steps in adaptive sampling while reducing the workload. The innovations of *ExTASY* are both the high-scalability and the modularity. The modularity allows for an easy change of the adaptive sampling strategies and better maintainability. *ExTASY* is reducing the entry barrier to utilizing adaptive sampling.

Finally, the package *ExTASY* will be applied to show the results of adaptive sampling for several proteins. Future developments to extend the investigated approaches to longer timescales will be addressed.

All the approaches mentioned above facilitate further advancements in predicting conformational dynamics of larger biomolecules.

# Acknowledgments

This dissertation is just the final product of a long path, starting when I joined Rice University in 2014. Along this long path, I have received lots of help and lots of advice. Here I want to thank everyone even though I can only mention some.

At the center of this path is Cecilia Clementi. As my advisor, she guided me through the ups and downs, helping me understand why something did or did not work. This has helped me grow professionally and personally. She consistently encouraged me to improve my results and persist when software bugs caused bad results. Without her, this work would not have been possible; she has been an incredible advisor.

All my previous and current lab mates played a significant role in helping me along this path. Jordane and Lorenzo helped me from the start to learn the crucial as well the less obvious skills for this path. Many thanks to Shantenu, Vivek, and Jayvee for being great collaborators and for being always willing to help, whatever the question.

I would also like to thank Jose Onuchic, Jason Hafner, and Matteo Pasquali for agreeing to sit on my PhD defense committee.

Last but not least, I want to thank my family for the unconditional support. Thank you.

# Contents

Abstract iii

List of Illustrations ix

List of Tables xii

**1 Motivation and Approach 1**

1.1 Adaptive sampling: Motivation . . . . . . . . . . . . . . . . . . . . . 1

1.2 Outline . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . 2

**2 Background 5**

2.1 Proteins and Molecular Dynamics . . . . . . . . . . . . . . . . . . 5

2.2 TICA dimension reduction . . . . . . . . . . . . . . . . . . . . . 7

2.3 State-free reversible VAMPnets . . . . . . . . . . . . . . . . . . . 9

2.4 Markov state models . . . . . . . . . . . . . . . . . . . . . . . . 11

2.5 Protein folding funnel . . . . . . . . . . . . . . . . . . . . . . . 11

**3 Adaptive Sampling Theory 14**

3.1 Sampling Problem . . . . . . . . . . . . . . . . . . . . . . . . . 14

3.2 Alternative sampling approaches . . . . . . . . . . . . . . . . . . 16

3.3 Adaptive sampling schema . . . . . . . . . . . . . . . . . . . . . 17

3.4 Adaptive sampling strategies . . . . . . . . . . . . . . . . . . . . 20

3.5 Restart Strategies for Adaptive Sampling . . . . . . . . . . . . . . 22

3.5.1 Adaptive sampling strategy *cmicro* . . . . . . . . . . . . . . . 23

3.5.2 Adaptive sampling strategy *cmacro* . . . . . . . . . . . . . . . 23

3.5.3 Adaptive sampling with *a priori* information . . . . . . . . . . 25

3.5.3.1 Adaptive sampling strategy $Q_f$ . . . . . . . . . . . . 25
3.5.4 Adaptive sampling strategy $Q_{f,nn}$ . . . . . . . . . . . . . . . . 26
3.5.4.1 Additional restarting strategies . . . . . . . . . . . 26
3.5.5 plain MD . . . . . . . . . . . . . . . . . . . . . . . . . . . 27
3.6 Upper limits to adaptive sampling strategies . . . . . . . . . . . . . . 27
3.6.1 Optimal strategy for exploration: $p_{esc}$ . . . . . . . . . . . . . . 27
3.6.2 Optimal strategy for protein folding: $t_{opt}$ . . . . . . . . . . . 28
3.7 Discussion . . . . . . . . . . . . . . . . . . . . . . . . . . . . . 29

**4 Adaptive Sampling Comparison 31**
4.1 Methods . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . 32
4.1.1 Reference Dataset of Simulations . . . . . . . . . . . . . . . . 32
4.1.2 Construction of Markov State Models . . . . . . . . . . . . . . 33
4.1.3 Simulating adaptive sampling with Markov Chain trajectories 34
4.2 Results . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . 35
4.2.1 Time to fold . . . . . . . . . . . . . . . . . . . . . . . . . . 36
4.2.2 Time to explore 95% of states . . . . . . . . . . . . . . . . . 39
4.2.3 Scaling . . . . . . . . . . . . . . . . . . . . . . . . . . . . . 41
4.2.4 Speedup for different proteins . . . . . . . . . . . . . . . . . 43
4.3 Discussion . . . . . . . . . . . . . . . . . . . . . . . . . . . . . 44

**5 Adaptive sampling software framework 48**
5.1 Introduction . . . . . . . . . . . . . . . . . . . . . . . . . . . . 48
5.2 Alternative software . . . . . . . . . . . . . . . . . . . . . . . . 49
5.3 Asynchronous execution . . . . . . . . . . . . . . . . . . . . . . 50
5.4 Tools and Software . . . . . . . . . . . . . . . . . . . . . . . . . 52
5.5 Pilot abstraction . . . . . . . . . . . . . . . . . . . . . . . . . . 54
5.6 Scaling . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . 54
5.7 Conclusion . . . . . . . . . . . . . . . . . . . . . . . . . . . . . 56

**6 Application of adaptive sampling 58**

6.1 Introduction . . . 58

6.2 Reference Data . . . 59

6.3 Setup . . . 60

6.4 Results . . . 62

6.4.1 Comparison of Exploration . . . 62

6.4.2 Comparison of Protein Dynamics . . . 69

6.5 Conclusion . . . 73

**7 Conclusions 77**

**Bibliography 80**

# Illustrations

2.1 Structure of proteins $\lambda$-repressor and WW domain. . . . . . . . . . . 6
2.2 Schematics of the state-free reversible VAMPnets (SRV). . . . . . . . 10
2.3 Protein folding funnel. . . . . . . . . . . . . . . . . . . . . . . . . . 12

3.1 Rare transitions barriers cause the sampling problem. . . . . . . . . . 15
3.2 Restarting method in adaptive sampling strategies. . . . . . . . . . . 18
3.3 Basic schema of adaptive sampling. The number of starting conformations, analysis strategies or the stop condition in step 3 are flexible and can be changed. . . . . . . . . . . . . . . . . . . . . . 19

4.1 Time to fold for the different sampling strategies for proteins Chignolin, WW Domain and $\alpha$3D. . . . . . . . . . . . . . . . . . . . . 36
4.2 Time to the exploration of 95% of the states for the different sampling strategies for proteins Chignolin, BBA and $\alpha$3D. . . . . . . 39
4.3 Top] Scaling of the absolute (left) or cumulative (right) number of steps required to explore 95% of all microstates. Strategies plain $MD$, $p_{esc}$, $1/C$ and $1/C^K_{M,2}$ for the protein BBA. Bottom] Scaling of the absolute folding time for sampling strategies; plain $MD$, $p_{esc}$, $t_{opt}$, $1/C$ and $1/C^K_{M,2}$ for WW Domain. . . . . . . . . . . . . . . . . . . 41
4.4 Correlation between the speedup of the folding time vs. mean first passage time for the 8 proteins and adaptive sampling strategies $1/C^K_{M,2}$, $Q_f$ and $t_{opt}$. . . . . . . . . . . . . . . . . . . . . . . . . . . 43

5.1 Asynchronous execution of an ensemble of adaptive sampling subtasks, including molecular dynamics and analysis tasks. The restart conformations for the next molecular dynamics simulation are determined by the last finished analysis task. Here steps 2 and 4 are shown combined as analysis steps. . . . . . . . . . . . . . . . . . . . . . 51
5.2 Pseudocode showing the modular software design and the EnTK API backend . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . 53
5.3 EnTK backend: This schema shows the modularity which enables ExTASY's scalability while ensuring flexibity. . . . . . . . . . . . . . 53
5.4 Scaling of ExTASY on Summit. The Efficiency is defined as the ratio between the nodehours used by all the individual tasks and the nodehours used by the software platform. . . . . . . . . . . . . . . . . 55

6.1 The recovery of the energy landscape by adaptive sampling for 4 proteins. The projection on TICA coordinates is shown. The color background shows the reference Free Energy landscape, and the black diagonal lines show the adaptive sampling explored energy landscape. The location of the folded states are shown. A) Chignolin B) Villin . 64
6.1 (cont.) C) BBA D) A3D . . . . . . . . . . . . . . . . . . . . . . . . . 65
6.2 The fraction of the explored population increases with absolute simulation time. The absolute simulation time is shown with a logarithmic scale. The vertical lines pinpoint folding events. The speedup with adaptive sampling depends on the protein and generally increases with the size of the protein. A) Chignolin B) Villin . . . . 67
6.2 (cont.) C) BBA D) A3D For protein A3D the limited computational resources interrupted the folding with plain MD. The adaptive sampling of A3D was able to fold due to the speedup of adaptive sampling. . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . 68

6.3 The evolution of the unfolding Mean First Passage Times is shown over the simulation time. Adaptive sampling is red and plain MD is blue. The reference values are shown in black. A) Chignolin B) Villin C) BBA . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . 71

6.4 The evolution of the relative entropy between the MSM transition matrices from adaptive sampling and plain MD is shown. For protein Villin the relative entropy decreases with increasing simulation time. . 73

# Tables

4.1 Proteins for reference data . . . . . . . . . . . . . . . . . . . . . . . . 32

6.1 Proteins for the comparison of adaptive sampling and plain MD. . . 59
6.2 ExTASY parameters for MD and analysis, for both plain MD and adaptive sampling. . . . . . . . . . . . . . . . . . . . . . . . . . . . . 62

# Chapter 1

# Motivation and Approach

## 1.1 Adaptive sampling: Motivation

Biomolecules fulfill many different roles in nature and have accordingly a wide range of timescales associated with the conformational dynamics of the biomolecule. Molecular dynamics (MD) simulations have become indispensable for gaining insight into the behavior of biomolecules at high spatial and temporal resolutions. However, a fundamental limitation for MD with accurate all-atom force-fields remains the computational demand for simulating processes with long timescales. In particular, many biologically relevant processes, such as protein folding and conformational changes, typically require simulation times which are orders of magnitude longer than milliseconds. In contract, atomic-resolution MD trajectories can currently reach only timescales in the order of milliseconds.

The ambition of this thesis is coming closer to solving the challenge of accurately determining the conformational dynamics of proteins. This challenge is immense due to the many orders of improvements necessary. Many approaches have been implemented which improve the sampling of proteins, including hardware advances and improved molecular dynamics software. Some of the approaches are introduced in this chapter. One particular approach this thesis studies is *adaptive sampling*. Adaptive sampling uses many shorter MD trajectories to sample the protein energy landscape and the increased efficiency of sampling is reached by selecting the starting points of the MD trajectories based on the results of previous MD trajectories. This approach allows sampling more efficiently any high-dimensional stochastic systems. Proteins are a common example of high-dimensional stochastic systems.

## 1.2 Outline

Overall this Dissertation resulted in several peer-reviewed publications [1, 2], as well one publication under review [3]. The results obtained during this Dissertation, as well as the results in the papers, will be presented in the next chapters.

Chapter 2 will summarize some of the standard techniques, which will be used frequently throughout the thesis. While some of these approaches are commonly used in this field, there are also recently developed techniques that are less commonly used. The relevance of all these techniques for adaptive sampling will be highlighted.

The theory of adaptive sampling will be discussed in Chapter 3, as well the development of an upper limit of speed up with adaptive sampling will be shown. This novel result shows that adaptive sampling can be further improved. In the next chapter 4, the current strategies will be compared in a statistically robust way, and the effectiveness of each strategy will be compared for different proteins and different goals.

Chapter 5 will introduce the computational tools and software frameworks necessary to successfully and efficiently execute adaptive sampling. Due to the many subtasks adaptive sampling contains, these practical considerations have a direct impact on the viability of adaptive sampling for solving sampling problems. This software framework will be used in Chapter 6 to show the results and the accuracy which adaptive sampling can achieve for different biomolecular systems.

A final chapter will summarize the current achievements and discuss further open questions and possible directions in further developing adaptive sampling.

# Chapter 2

# Background

## 2.1 Proteins and Molecular Dynamics

The behavior of a protein or any biomolecule can be relatively accurately represented by Newton's classical equations of motion for atoms. All 3N atom coordinates, including the surrounding water solvent, are represented by $\mathbf{x}_t$ positions and $\mathbf{v}_t$ velocities. Figure 2.1 shows the 3-dimensional structure of two of the proteins investigated in this thesis to visualize the shape of the 3D structure. The Newton's classical equations of motion are:

$$M\ddot{\mathbf{x}}_t = -\nabla U(\mathbf{x}_t)$$

The potential energy $U()$ represents the all-atom force field guiding the motion of the protein. The straightforward approach of numerically solving this equation leads to a molecular dynamics (MD) trajectory describing the motion of a protein. This potential energy $U()$ approximates the quantum-mechanical dynamics of the protein and new research increasingly improves the accuracy of this approximation. An higher accuracy than all-atom folce fields would require a quantum-mechanical approach, which additionally reduces the reachable timescales by several orders of magnitude. In this thesis, the CHARMM22* force field [4] with modified TIP3P water was utilized.

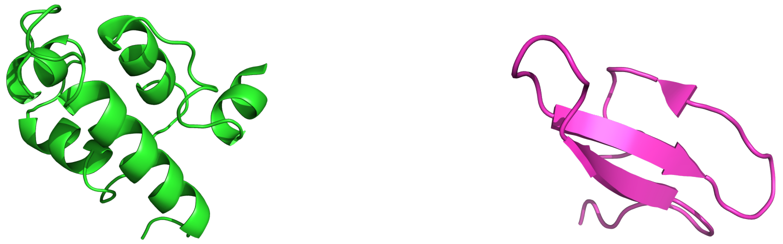


Figure 2.1 : Structure of proteins $\lambda$-repressor and WW domain.

The CHARMM force field has the following potential energy, which consists of terms for molecular bonds stretching, angle bending, dihedral bending, improper dihedral energy terms, the Urey-Bradley component, non-covalent van der Waals interactions (Lennard-Jones) and Coulomb interactions.

$$\begin{aligned} U(\mathbf{x}) &= U_{bonds} + U_{angle} + U_{dihedral} + U_{improper} + U_{UB} + U_{LJ} + U_{elec} \\ &= \sum_{bonds} K_b(b-b_0)^2 + \sum_{angles} K_\theta(\theta-\theta_0)^2 + \sum_{torsions} K_\phi\left(1+cos(n\phi-\delta)\right) \\ &+ \sum_{impropers} k_\omega\left(\omega-\omega_0\right)^2 + \sum_{Urey-Bradley} k_u\left(u-u_0\right)^2 \\ &+ \sum_{nonbonded} \epsilon\left[\left(\frac{R_{min,ij}}{r_{ij}}\right)^{12} - \left(\frac{R_{min,ij}}{r_{ij}}\right)^{6}\right] + \sum_{electric} \frac{q_i q_j}{\epsilon r_{ij}} \end{aligned} \tag{2.1}$$

In the above equation the parameters are: $k_b$ is the bond stretching force constant, $b-b_0$ is the atom distance from equilibrium, $k_\theta$ is the angle bending force constant, $\theta-\theta_0$ is the angle from the equilibrium between 3 bonded atoms, $k_\phi$ is the dihedral force constant, $n$ is the multiplicity of the dihedral function, $\phi$ is the dihedral angle, $\delta$ is the phase shift, $k_\omega$ is the improper dihedrals bending force constant, $\omega-\omega_0$ is the out of plane improper dihedrals angle, $R_{min_{ij}}$ is the constant for the Lennard-Jones potential, $r_{ij}$ is the distance between a pair of atoms and $q_i$ is the charge of an atom. The Urey-Bradley component serves the cross-term accounting for angle bending using 1,3 nonbonded interactions, where $k_u$ is the Urey-Bradley force constant, and $u - u_0$ is the distance between the 1,3 atoms in the harmonic potential.

One challenge of the straightforward approach of utilizing Newton's classical equations of motion for the prediction of protein behavior is the relatively short timestep of numerically simulating the molecular dynamics. The commonly used maximum timestep allowing for energy conservation and stability of the protein is between 2fs and 5fs. Considering most proteins fold on timescales longer than milliseconds, this requires to simulate in the order of $10^{12}$ steps. With modern GPUs, for smaller proteins, almost 1 microsecond of molecular dynamics trajectory can be simulated per day. Special purpose hardware [5] can simulate up to 100 microseconds of MD trajectory per simulation day, but the access to this special purpose hardware is limited and commonly GPUs are used. The simulation of the behavior of most biomolecules for relevant processes such as protein folding and other global structural rearrangements requires timescales of around $10^{-2}s - 10^{0}s$. These slow processes are associated with the rare crossing of high free energy barriers, representing a bottleneck for the MD simulations.

## 2.2 TICA dimension reduction

A protein molecular dynamics trajectory is high-dimensional, with 1000s or more individual atom trajectories. This high-dimensionality needs to be dimension-reduced to be effectively analyzed. One approach is the Time-lagged Independent Component Analysis (TICA) [6, 7], which includes the kinetic information of protein dynamics in the dimension reduction.

The first step for TICA is the conversion of $\mathbf{x}_t$ trajectories into features $\mathbf{f}_t$ which are rotation- and translation-invariant and mean-free:

$$\mathbf{f}_t = F\left(\mathbf{x}_t - \langle \mathbf{x}_t \rangle\right)$$

Some examples of features are atom distances or dihedral angles. The rotation- and translation-invariance is an usefull property since the protein dynamics is rotation- and translation-invariant.

The next step of TICA is the calculation of the correlation matrix $C_{00}$ and the time-lagged correlation matrix $C_{01}$:

$$C_{00} = \mathbb{E}_t\left[\mathbf{f}_t \mathbf{f}_t\right]$$

$$C_{01} = \mathbb{E}_t\left[\mathbf{f}_t \mathbf{f}_{t+\tau}\right]$$

The generalized eigenvalue problem for the correlation matrices leads to TICA, where $\mathbf{R}$ is the eigenvector matrix, and $\mathit{\Lambda}$ is the diagonal eigenvalue matrix.

$$C_{01}\mathbf{R} = C_{00}\mathbf{R}\mathit{\Lambda}$$

The eigenvalues $\lambda_i$ allow the calculation of the timescales $t_i$. Generally, the dimensions with the longest timescales are relevant for biophysical applications.

$$t_i(\tau) = -\frac{\tau}{log\left|\lambda_i\right|}$$

The TICA dimension reduced trajectory $\mathit{\Psi}(t)$ is obtained by projecting the feature trajectory on the $n$ slowest TICA eigenvectors. A kinetically meaningful state decomposition can be obtained with the commute map [8] where the slowest TICA coordinates are scaled with the corresponding timescales.

$$\Psi_{commute} = \sqrt{\frac{t_i}{t_o}}\Psi$$

The fraction of kinetic content allows the estimation of the dimensionality of the protein behavior [8].

$$c_m = \frac{\sum_{i=1}^{m} t_i}{\sum_{i=1}^{n} t_i}$$

For non-equilibrium input $\mathbf{x}_t$ trajectories, the plain TICA will lead to inaccurate results due to the non-stationary distribution of the input correlation matrices. The Koopman method [9–14] can be used to reduce the non-equilibrium effects by reweighting the correlation matrices to the stationary distribution.

## 2.3 State-free reversible VAMPnets

One disadvantage of the TICA dimension reduction is the linearity of this approach. One non-linear dimension reduction is a deep learning approach with state-free reversible VAMPnets (SRV) [15, 16]. The SRV method is closely related to the TICA methods, but with additional neuronal networks and backpropagation. The input to SRVs are the same protein trajectories $\mathbf{x}_t$ and feature trajectories $\mathbf{f}_t$ as for TICA. According to the schema in Figure 2.2 a neuronal network converts the input features $\mathbf{f}_t$ into a dimension reduced trajectory $\mathbf{o}_t$.

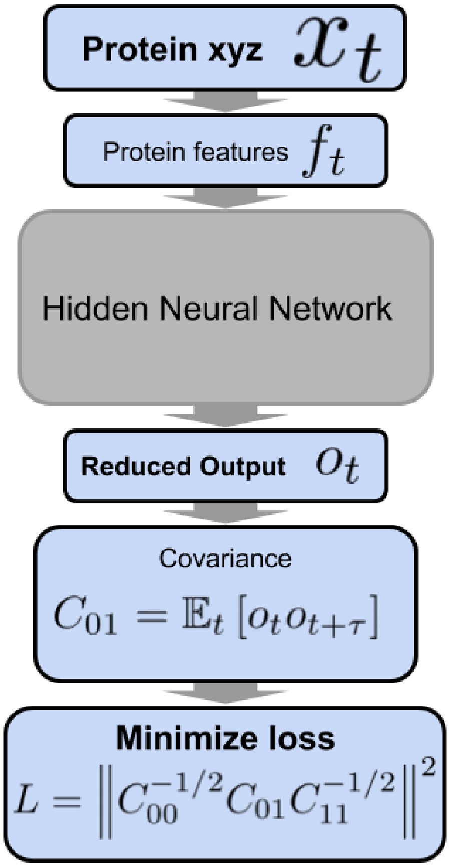

Protein xyz $x_t$
Protein features $f_t$
Hidden Neural Network
Reduced Output $o_t$
Covariance
$C_{01} = \mathbb{E}_t\left[o_t o_{t+\tau}\right]$
Minimize loss
$L = \left\| C_{00}^{-1/2} C_{01} C_{11}^{-1/2} \right\|^2$

Figure 2.2 : Schematics of the state-free reversible VAMPnets (SRV).

The loss function for optimizing the parameters of the neural network with backpropagation is the VAMP-2 score:

$$L = \left\| C_{00}^{-1/2} C_{01} C_{11}^{-1/2} \right\|^2$$

Here the modified covariance matrix is $C_{01} = \mathbb{E}_t\left[o_t o_{t+\tau}\right]$. The non-linear approach of SRVs allows reaching the same accuracy as TICA for shorter lag times than TICA. TICA performs better than SRVs in the case of limited data amounts.

## 2.4 Markov state models

The dimension reduced trajectories from TICA or SRVs are ideal for the generation of Markov state models (MSM) [17]. MSM can describe the non-linear multi-state behavior of proteins. The states are commonly defined by clustering the dimension reduced trajectory frames with k-means clustering. The clustering allows converting the dimension reduced trajectory into discrete (state) trajectories. The transitions of the discrete trajectories between states lead to the count matrix $C_{ij}$. The count matrix $C_{ij}$ indicates the number of transitions from state $i$ to state $j$ after a time lag $\tau$. The count matrix is row-normalized to obtain the transition matrix of the MSM $T_{ij}$.

The MSM behavior can be described by the eigendecomposition of the transition matrix. $\mathit{\Lambda}$ is the diagonal matrix of eigenvalues, and $\mathbf{U}$ is the eigenvector matrix. The slowest eigenvector represents the stationary distribution of state probabilities.

$$\mathbf{U}T = \mathbf{U}\mathit{\Lambda}$$

## 2.5 Protein folding funnel

The protein folding behavior can we visualized by the folding funnel of the energy landscape theory of protein folding [18]. The unfolded protein is on top of the schematic in Figure 2.3. The x-axis shows the energetic stabilization of the folded state versus the unfolded state, and the width of the folding funnel shows the conformational entropy of the states. The folded state of the protein at the bottom of the funnel corresponds to protein free energy minimum. The ”rough” energy landscape represents the high-dimensional energy landscape with many local energy minimas.

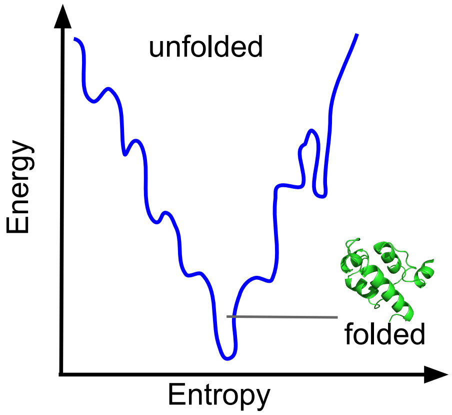

unfolded
Energy
folded
Entropy

Figure 2.3 : Protein folding funnel.

# Chapter 3

# Adaptive Sampling Theory

In this Chapter, we investigate the theory of Adaptive Sampling with the goal to better understand the possibilities and limits of adaptive sampling. The results stated here were originally published in:

[1] **Hruska, E.**; Abella, J. R.; Nüske, F.; Kavraki, L. E. & Clementi, C.; Quantitative comparison of adaptive sampling methods for protein dynamics. J. Chem. Phys. 149 (2018)

## 3.1 Sampling Problem

The stochastic behavior of biomolecules requires a good sampling of the process to understand the dynamical behavior of the proteins accurately. This sampling can be done by molecular dynamics, and for small biomolecules such as peptides, this approach reaches a sufficient sampling. Larger biomolecules pose a significantly larger challenge to sample well. One cause are the slow collective motions with long timescales. Figure 3.1 illustrates that in many cases, there are rare transitions across a rare transition barrier. Simulating a longer molecular dynamics trajectory on one side of the barrier leads to a good sampling of one side of the barrier, but a low probability of crossing the rare transition barrier. This bottleneck causes an oversampling of the area on one side on the transition barrier.

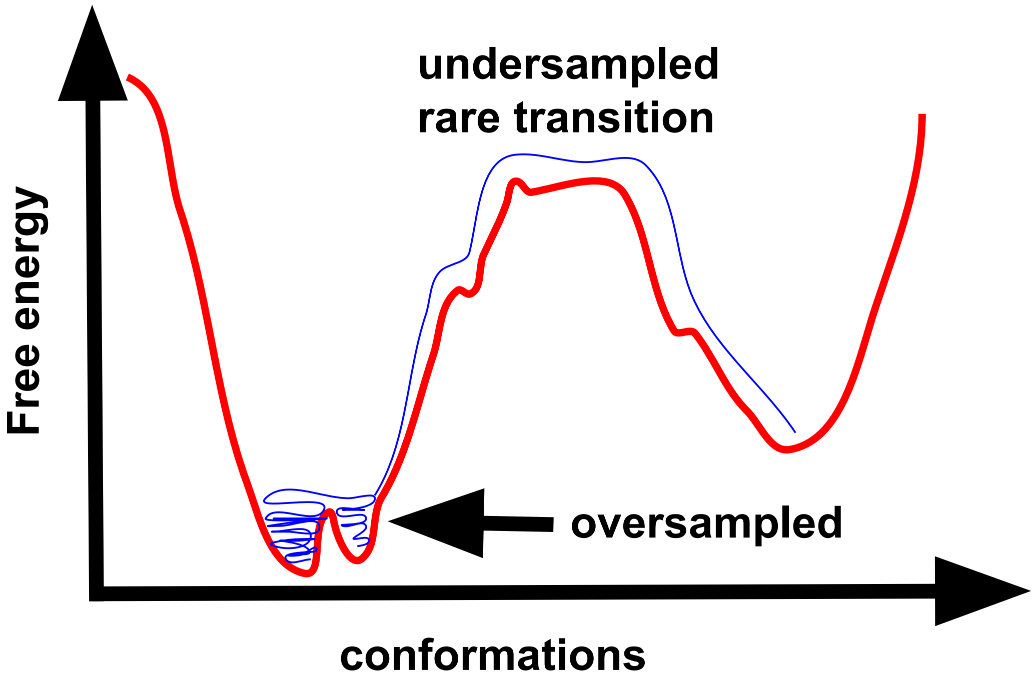


Figure 3.1 : Rare transitions barriers cause the sampling problem.

Figure 3.1 shows the free energy landscape along on a reaction coordinate. The high sampling at the bottom of the free energy wells is caused by the Boltzman distribution, which describes the equilibrium probability density of conformation space for biomolecules. The equilibrium probability density $\pi$ is:

$$\pi(x) \; e^{-F(x)/(k_B T)}$$

Here $F(x)$ is the free energy at reaction coordinate point $x$, $k_B$ is the Boltzmann constant and $T$ is the temperature. The low probability density at transition barrier leads to a slow sampling of the transition. Single long simulation would require long simulation times to sample the whole phase space of the biomolecules. The excess computing power sampling the bottom of the free energy landscape could be redirected to sample the undersampled side of the transition barrier. For proteins, the rare transition barrier could be protein folding or other large-scale conformational changes.

## 3.2 Alternative sampling approaches

The sampling problem, as well the desire to accurately sample the dynamics of high-dimensional stochastic systems has led to many approaches besides adaptive sampling. These efforts can be broadly categorized, but this summary can not exhaustively discuss all approaches due to a large number of these approaches. One approach is to simulate longer molecular dynamics trajectories by software and hardware approaches. In hardware advancements include the utilization of Graphics processing units (GPUs) and special-purpose hardware [5]. The graphics cards can currently simulate almost 1 microsecond of MD trajectory per day of simulation time for smaller proteins. The special-purpose hardware can simulate up to 100 microseconds of MD trajectory per simulation day, but the major limitation of the special-purpose hardware is the minimal access to these machines. An important role play software advances that can effectively utilize this hardware. Despite all the improvements in speed, these approaches perform simple molecular dynamics simulations with a time step of 2-5fs.

Another approach, incentivized by the increase in parallelization of High-Performance Computers, is the simulation of many simultaneous trajectories [19, 20]. The sampling is performed in parallel for the same biomolecular system, and by analyzing the resulting trajectories, a better sampled result is obtained. As shown in Chapter 4, the total efficiency of sampling is reduced due to the independent nature of the sampling, but the time to solution is reduced.

Monte Carlo is commonly used to improve the simulations of many stochastic systems. Here the sampling is not constrained to the small continuous steps common to molecular dynamics. This allows Monte Carlo to perform large jumps in conformation space, which is impossible in molecular dynamics. The high dimensionality of biomolecular systems reduces the effectivity of Monte Carlo Simulations, which doesn't represent well the collective motions of biomolecules. The approach of Hybrid Monte Carlo attempts to combine the strength of both Monte Carlo and MD, by including information of the intermolecular forces in the Monte Carlo moves. This approach is promising but wasn't yet able to reach longer timescales for biomolecules than other approaches.

Modifying the Hamiltonian of the biomolecules is another common approach to solving the sampling problem. Here the Free energy barrier is reduced by adding terms to the original Hamiltonian of the biomolecules. The reduced free energy barrier leads to faster sampling of the modified Hamiltonian landscape. Different approaches such as metadynamics [21] or accelerated MD [22] are possible. The stationary probability distribution can be recovered according to the Boltzmann distribution. Accurate kinetic information or conformational dynamics cannot be measured directly. Recent methods in recovering the kinetic information [23–26] are investigated but haven't been widely used.

## 3.3 Adaptive sampling schema

The general idea of *adaptive sampling* [1,27–37] is the "divide and conquer" approach. Simulating shorter molecular dynamics trajectories allows changing the restarting points for the next short molecular dynamics trajectories. The choice of this restarting points is a key element of this thesis. Figure 3.2 illustrates that the free choice of restarting points allows to clone conformations and sample certain areas of the energy landscape better. The black crosses in the figure indicate areas where sampling is reduced or blocked, reducing the utilization of computational resources in some areas. Adaptive sampling allows the use of multiple parallel simulations and is therefore effective on High-Performance Computers (HPC).

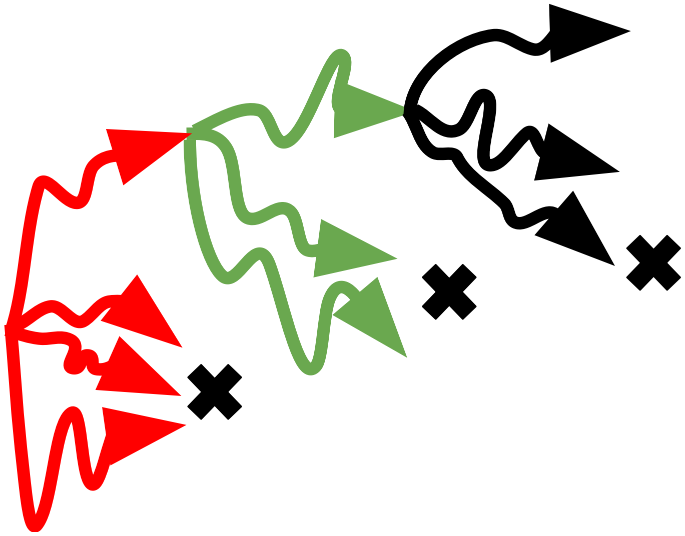


Figure 3.2 : Restarting method in adaptive sampling strategies.

In the iterative process of adaptive sampling, the previously generated molecular dynamics trajectories are analyzed, and the analysis results inform the restarting points for the next batch of molecular dynamics trajectories. Figure 3.3 shows the initialization and the 4 steps of adaptive sampling. Different adaptive sampling strategies have different implementations of the analysis steps, steps 2 and 4.

The individual adaptive sampling steps are summarized as follows:

- Start: Initialization of the start conformations. Commonly an unfolded structure of the selected proteins is chosen.

- Step 1: Simulating an ensemble of MD trajectories. The conformations are

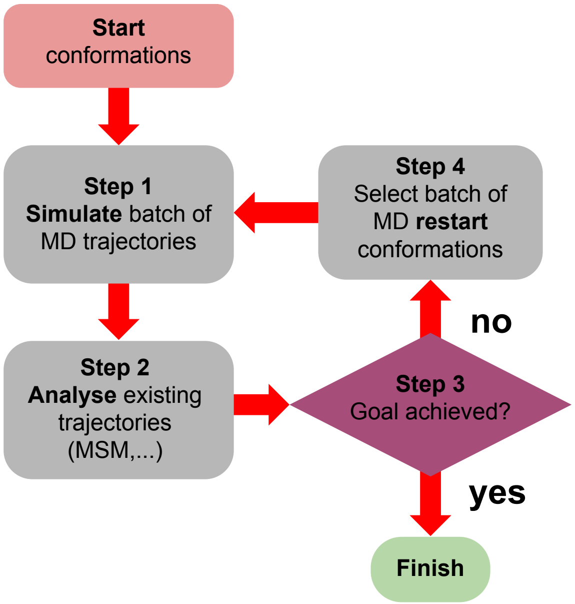

Start
conformations
Step 1
Simulate batch of
MD trajectories
Step 2
Analyse existing
trajectories
(MSM,...)
Step 3
Goal achieved?
no
yes
Step 4
Select batch of
MD restart
conformations
Finish

Figure 3.3 : Basic schema of adaptive sampling. The number of starting conformations, analysis strategies or the stop condition in step 3 are flexible and can be changed.

chosen either by Initialization or Step 4.

- Step 2: Analysis of all previously generated MD trajectories. The analysis varies between adaptive sampling strategies.
- Step 3: Stop condition. Automatic determination if the objective of adaptive sampling is achieved. When the objective is not achieved, proceed iteratively to Step 4.
- Step 4: Selection of restart conformations for the next set of MD trajectories based on the analysis results.

The beginning step of any adaptive sampling is the initialization. The start conformation from where adaptive sampling is launching is chosen. For each replica, which depends on the parallelization of the computational resource, a conformation is generated. The start conformations can be all identical or disparate. For proteins commonly an unfolded state is chosen. In step 1, the MD simulations for all replicas are performed in a parallelized mode. Here most of the computational resources are utilized, and scalability of this step is crucial. Following the MD simulation is the analysis of the MD trajectories in step 2. In the Chapters 5 and 6, the adaptive sampling strategies *cmicro* and *cmacro* with Markov State Models [38] are used as described in this Chapter.

The results of Step 2 allow determining in Step 3 if the objectives of adaptive sampling are reached. Examples for the objectives are protein folding, achieving a certain accuracy in protein dynamics, or exploration of a part of the energy landscape corresponding to a smaller-scale motion of the protein. If the objective is not reached, Step 4 creates the restarting conformation for the next MD trajectories corresponding to Step 1. The adaptive sampling strategies to determine the restarting points in Step 4 in Figure 3.3 are easily exchangeable, such as the strategies discussed below. Once the objective in Step 3 is reached, adaptive sampling finishes and all trajectories can be further analyzed. Once the sampling is finished, the dimension reduction and analysis methods are utilized to extract accurate stationary and dynamics information about the biomolecule.

## 3.4 Adaptive sampling strategies

The general advantage of adaptive sampling compared to other non-adaptive sampling strategies are the resulting accurate conformation dynamics results and the increased timescales reachable. While the success of adaptive sampling has been demonstrated in applications [39–41], the success of adaptive sampling depends on the chosen adaptive sampling strategy. Different methods have been proposed and investigated [1, 27–37, 42–47].

Despite the range of adaptive sampling methods, there is no consensus which adaptive sampling strategies perform best for which sampling goal. This lack of comparability between different adaptive sampling strategies is partially caused by the number of disparate ways the individual adaptive sampling methods are implemented. In Chapter 5, a flexible software framework is developed, which allows the utilization of any of the adaptive sampling strategies and improves the comparability between the adaptive sampling strategies.

One common difference between the adaptive sampling strategies is the method to dimension reduce the molecular dynamics trajectories. The dimension reduced molecular dynamics trajectories are generally easier to analyze further. Some adaptive sampling strategies such as CoCo-MD [46,48,49] use PCA. Other dimension reduction methods are TICA [6, 7] , Diffusion Maps [50–53], likelihood-based approaches [54], cut-based free energy profiles [55], or neural networks such as VAMPnets (SRV) [15, 16, 56, 57].

Multiple layers of dimension reduction such as combining TICA with Markov State Models (MSMs) [38, 58–61] are common. MSMs are able to process many short trajectories and do not require equilibrium sampling to recover thermodynamics and kinetic properties. The ability of analyzing non-equilibrium sampled data is essential for adaptive sampling, due to the increased sampling of parts of the protein energy landscape. The most common dimension reduction methods are TICA in combination with Markov State Models (MSMs) due to the validation of this approach, but new approaches such as VAMPnets offer some benefits such as the ability to analyze shorter trajectories at same accuracy. Shorter trajectories allow potentially a more frequent restarting for adaptive sampling.

The restarting strategies based on the dimension reduced trajectories show significant variety, making a comparison complex. Some of the common restarting strategies are the *cmicro* and *cmacro* strategy, and adaptive sampling with *a priori* information. These strategies are described in the following section.

## 3.5 Restart Strategies for Adaptive Sampling

Different adaptive sampling strategies can be broadly categorized into strategies with and without *a priori* information. In the following section, the adaptive sampling strategies *cmicro* and several types of *cmacro* are introduced, which all don't utilize *a priori* information. These strategies select restarting positions only according to the iteratively updated analysis results.

To contrast, two strategies utilizing *a priori* information, $Q_f$ and $Q_{f,nn}$, are introduced. These strategies take advantage of the previously known slow collective modes for the studied protein or utilize the intuition of a domain experts on the behavior of proteins. The introduced strategies are not exhaustive but cover the more commonly used adaptive sampling strategies.

Common to these strategies is the use of MSM for the analysis, due to the effective analysis of the non-equilibrium trajectories generated by adaptive sampling. Chapter 2 described the steps standard in generating an MSM from the adaptive sampling. The MSM is repeatedly generated for each iteration of the adaptive sampling to include the additional molecular dynamics trajectories.

### 3.5.1 Adaptive sampling strategy *cmicro*

One less complex adaptive sampling strategy is the *cmicro* strategy, also called $1/C$ strategy. This strategy chooses the restart states based on the count of molecular dynamics trajectories that have visited each MSM state [29, 30, 32, 33]. By restarting in less visited states, the undersampled areas of the proteins will be sampled more. Generally, for a given MSM state $i$, the restart probability is proportional inversely to the count for this state. The c in *cmicro* strategy stands for the count of times that molecular dynamics trajectories have reached a particular state. The *cmicro* strategy is particularly effective in exploring, but less effective in crossing rare transition regions [1].

### 3.5.2 Adaptive sampling strategy *cmacro*

A further development of the *cmicro* strategy is the *cmacro* strategy, also called $1/C_M$ strategy. To improve the crossing of rare transition regions, additional kinetic information of the MSM states is taken into account. A higher restart probability is chosen for states which are kinetically disconnected. This is implemented by clustering the MSM into kinetically connected states. Different clustering methods can be used, such as PCCA+ [62], or k-means clustering of the MSM in the MSM eigenvector dimensions [31, 32]. Each of the resulting clusters is one macrostate. Any MSM states not connected in the MSM is treated as an additional macrostate. The restarting probability is proportional to $1/C_{macro})$, where $C_{macro}$ is the count of molecular dynamics trajectories have visited each macrostate. For each restart macrostate, the restart states are selected inversely proportional to the micro state count. In this work, we investigated the effects of 4 variants of the *cmacro* strategy. One difference is the approach to setting the number of macrostates. The number of macrostates affects the kinetically connected states and affects the effectivity of this adaptive sampling strategy. One approach is to set the number of macrostates to a constant number such as 30. The other approach is to determine the number of macrostates based on the number of slow collective motions of the protein. One measure of the number of slow collective motions is the number of dimensions which reach comunatively 50% of the total kinetic content of the system.

An additional variation of the *cmacro* strategy is designed to reduce the effect of the non-equilibrium data on adaptive sampling. In the first step of the analysis, the dimension reduction of the molecular dynamics trajectories is affected by the non-equilibrium sampling of the trajectories. This non-equilibrium sampling is caused the different restarting probabilities across the protein landscape. One approach to reduce the effect of this non-equilibrium sampling on the dimension reduction is the Koopman reweighting method [9–14]. Here first, the stationary distribution is estimated, and then this stationary distribution is utilized to reweight the input molecular dynamics trajectories towards the stationary distribution. For incomplete adaptive sampling, the estimate of stationary distribution will only be partially correct due to only partially sampling the whole protein energy landscape. The improvement in dimension reduction leads to an improved MSM and better restarting probabilities.

The 4 variants of the *cmacro* strategy are the following:

$1/C^{C}_{M,1}$ The MSM is not adjusted, and the number of macrostates is fixed at 30.

$1/C^{C}_{M,2}$ The MSM is not adjusted, and the number of macrostates is based on the number of timescales with a 50% kinetic content.

$1/C^{K}_{M,1}$ The MSM is corrected to represent the equilibrium MSM, and the number of macrostates is fixed at 30.

$1/C^K_{M,2}$ The MSM is corrected to represent the equilibrium MSM, and the number of macrostates is based on the number of timescales with a 50% kinetic content.

The effectivity of setting the number of macrostates based on the number of timescales depends on the individual proteins and sampling of the protein behavior. For this reason, the more robust approach and the default variation of the *cmacro* strategy is $1/C^K_{M,1}$ with a constant number of macrostate together with the Koopman correction. All 4 variants have the goal of crossing slow transition barriers of proteins faster [1].

### 3.5.3 Adaptive sampling with *a priori* information

*A priori* information about the protein can significantly speedup the adaptive sampling by guiding the direction in which to sample. One approach [34] proposed restarting based on the number of folded contacts based on the evolutionary coupling analysis. The FAST method [35] utilizes the distance to a target structure. The effectivity of this approach also depends on the quality of the *a priori* information. This general class of adaptive sampling strategies will be studied in this work by two different strategies, $Q_f$ and $Q_{f,nn}$.

#### 3.5.3.1 Adaptive sampling strategy $Q_f$

The $Q_f$ adaptive sampling strategy utilizes the native contacts to guide the sampling. Using the *a priori* known folded structure of the protein, for each state $i$, the number of native contacts $Q_i$ is calculated. To adaptively sample the folded state from the unfolded state, states with a higher number of native contacts are preferred. The restarting probability is proportional to $exp(-k * |Q_i - Q_{max}|)$. Here, $Q_{max}$ is the total number of native contacts. The parameter $k$ controls the strength of guidance by this reaction coordinate and is in the range of 0.1 to 1, with a smaller number generally for larger proteins. This strategy works effectively when the folding path of the protein is simple. For some proteins, this strategy leads to misfolded states with high $Q_i$. Return to the folding path is slow with this strategy since some contacts would have to be broken, which corresponds with a low restarting probability.

### 3.5.4 Adaptive sampling strategy $Q_{f,nn}$

The strategy $Q_{f,nn}$ is designed to prevent the misfolded states from strategy $Q_f$. Here both native and non-native contacts are utilized in calculating the restarting probabilities. Restarting in states with a higher number of non-native contacts is reduced. The restarting probability in state $i$ is proportional to $exp(-d_i)$, where $d_i = \sqrt{k_1^2 * (Q_i - Q_{max})^2 + k_2^2 * N_i^2}$. The additional parameters to strategy $Q_f$ are $N_i$, which is the number of non-native contacts, and $k_1, k_2$ are setting the strength of the sampling pressure along the native and non-native contact dimensions. This strategy avoids misfolded states and is faster and more robust than $Q_f$.

#### 3.5.4.1 Additional restarting strategies

Some additional restarting strategies are the following: DeepDriveMD [63] restarts from outlier points detected by the Density-based spatial clustering methods DBSCAN. This approach depends on the effectivity of the dimension reduction of the molecular dynamics trajectories. CoCo-MD extrapolates the restarting points outside of the already sampled conformations based on a PCA projection [46,48,49].

All these adaptive sampling restart strategies have similar, but not identical goals. Some of these goals are the faster crossing of rare transition barriers, such as protein folding, or accurate conformational dynamics results, or sampling of a small part of the energy landscape. The effectiveness of the individual adaptive sampling strategies for each of these goals is, in many cases, unclear. Studies [28–31] report speedup of protein folding with adaptive sampling between 2 and 10, but in most cases, these studies have a small sample size. Chapter 4 evaluates some of the adaptive sampling strategies for two different goals: crossing a rare transition barrier, such as the folding of a protein and the speedup of the exploration of large regions of the conformational space of a protein.

### 3.5.5 plain MD

While plain MD is not an adaptive sampling strategy, it is used as a comparison for all the adaptive sampling results. To illustrate the main difference with adaptive sampling, for plain MD no analysis is performed after each iteration, Steps 2 and 4 are missing. Instead, each MD trajectory is continued in the next iteration. To allow comparison with adaptive sampling, the parallelization of plain MD is chosen to be equal to the parallelization of adaptive sampling.

## 3.6 Upper limits to adaptive sampling strategies

### 3.6.1 Optimal strategy for exploration: $p_{esc}$

One objective for adaptive sampling is the optimal exploration of all states. This objective is relevant when the whole energy landscape has to be explored with no dominant slow processes. An upper limit can be derived when assuming that the full transition matrix of the protein investigated is known. The full transition matrix is not generally known and limits this strategy only as an upper limit for the speed of exploration. This approach assumes that the transition matrix represents the protein accurately. First, we calculate for each visited microstate $i$ the probability of transition to any microstate, which is not explored yet.

$$p_{esc}[i] = \sum_{j \in unexplored} T[i,j]$$

For this greedy approach on optimal strategy, the next restarting conformation is chosen from the state with the highest $p_{esc}$. The performance of this upper limit on exploration speed is investigated in Chapter 4.

### 3.6.2 Optimal strategy for protein folding: $t_{opt}$

Many biomolecules have significant transition barriers, which cause slow collective motions. Here the objective is to cross the transition barrier optimally. The problem can be rephrased as the mean optimal time to reach any state in the ensemble $s_1$. Frequently this ensemble of states is the folded state. Similarly to the $p_{esc}$ strategy we assume the full transition matrix for the protein is known. As the first step we designate the mean optimal time to reach $s_1$ from state $i$ as $t_{opt}[i]$. For every state in $s_1$ the value of $t_{opt}[i]$ is zero. All these $t_{opt}[i]$ values have interdependencies caused by our knowledge of the full transition matrix:

$$t_{opt}[i] = \sum_{j \in states} T[i,j] min(t_{opt}[i], t_{opt}[j]) + 1$$

This equation has two parts. The first part represents the restarting of the adaptive sampling in the current state $i$. According to the transition probability, the next steps would be state $j$. The new state $j$ is only accepted if the mean optimal time $t_{opt}$ of state $j$ is lower than the current state. All previously explored states with higher $t_{opt}$ are insignificant since the optimal strategy chooses only a state with a smaller mean optimal time $t_{opt}$. The addition of 1 is caused by the one restarting iteration in the first part. The result is in units of restarting steps. The above equation can be solved numerically. We use the $t_{opt}[i]$ values to define a benchmark restart strategy by selecting the restart state among the ones explored that has the lowest $t_{opt}$ value, representing the state that is the closest to the folded state. Note again that this strategy is impossible to implement in practice since the full MSM is not known a priori, but this strategy is still a useful benchmark for adaptive sampling strategies. The performance of this strategy $t_{opt}$ is setting an upper limit on the folding speed up with adaptive sampling and is investigated in Chapter 4.

## 3.7 Discussion

The two new upper limits on adaptive sampling speedup allow for an increased understanding of adaptive sampling strategies, continued in Chapter 4. Without any theoretical upper limit, the potential of adaptive sampling is unknown. The equations show that the potential of adaptive sampling depends on the transition matrix, which changes for each protein. Adaptive sampling strategies will be more effective when following the restarting decisions of $t_{opt}$ benchmark under the constrain of no *a priori* knowledge of the transition probabilities.

# Chapter 4

# Adaptive Sampling Comparison

One major challenge in developing, investigating, and comparing adaptive sampling strategies are the vast computational resources required to run adaptive sampling with the necessary molecular dynamics trajectories. To reach statistically significant sample sizes requires even considerably more computational resources, which are rarely available. The investigation of adaptive sampling for toy systems or small peptides requires a lower amount of computational resources. The lower dimensionality of these smaller test systems reduces the transferability of the resulting conclusions to biomolecules, which have orders of magnitude more degrees of freedom and a more complex energy landscape. Some strategies which work well for toy systems of small peptides face significant challenges to scale up to larger proteins. An alternative approach to investigating adaptive sampling strategies for larger biomolecules is the utilization of Markov Chain trajectories. These approximations of molecular dynamics trajectories can be generated many orders of magnitudes faster than molecular dynamics trajectories, allowing a better sample size than generating molecular dynamics trajectories for adaptive sampling. The requirement for the generation of Markov Chain trajectories is the knowledge of the Markov State Model of a protein. In this Chapter, adaptive sampling strategies were investigated with Markov Chain trajectories; the material in this Chapter was first published in:

[1] **Hruska, E.**; Abella, J. R.; Nüske, F.; Kavraki, L. E. & Clementi, C.; Quantitative comparison of adaptive sampling methods for protein dynamics. J. Chem.

Phys. 149 (2018)

## 4.1 Methods

### 4.1.1 Reference Dataset of Simulations

To generate accurate Markov State Models well-sampled simulations are necessary. Here we used published long all-atom molecular dynamics trajectories from the Anton supercomputer [64]. The Table4.1 shows the 8 proteins utilized. The 8 proteins are ranging from 10 to 80 residues. These proteins have different topologies, both $\alpha$-helices, $\beta$ sheets, and a mix of both. The folding times in simulation are ranging from 0.6 to 49 $\mu$s. These proteins are relatively fast-folding since only fast-folding proteins can currently be simulated with multiple folding and unfolding events.

Table 4.1 : Proteins for reference data

| Protein Name | PDB ID of Folded Structure | Size (# residues) | Folding Time ($\mu$s) [64] |
|---|---|---|---|
| Chignolin | 2RVD | 10 | 0.6 |
| Trp-cage | 2JOF | 20 | 14 |
| BBA | 1FME | 28 | 18 |
| WW Domain | 2F21 | 35 | 21 |
| Protein B | 1PRB | 47 | 3.9 |
| Homeodomain | 2P6J | 52 | 3.1 |
| $\alpha$3D | 2A3D | 73 | 27 |
| $\lambda$-repressor | 1LMB | 80 | 49 |

### 4.1.2 Construction of Markov State Models

Markov Chain trajectories can approximate molecular dynamics trajectories only when the Markov State Models are accurate. The MSM represents the stationary and dynamic behavior of the biomolecules. We have used the standard procedures to generate MSM. The trajectories were dimension-reduced with Time-lagged Independent Component Analysis (TICA) [6, 7], where the output dimension were scaled according to the commute map [8]. The input features for TICA were all pairwise inter-residue distances and all dihedral angles of the protein. The pairwise inter-residue distance is the closest heavy-atom distance between a pair of residues. For the smaller proteins, additional features of inverse inter-residue distances were used to increase the accuracy. This dimensionally reduced space was clustered with k-means clustering into 1000 or 2000 states, depending on the size and sampling of the protein. This larger number of microstates is selected to increase the accuracy of the Markov Chain trajectories and is only possible due to the good sampling of the reference trajectories. The disconnected microstates were removed, and the folding-unfolding process was set as the slowest MSM eigenvector.

For each protein, a set of folded and unfolded states are defined. The native contacts are extracted from the folded structures shown in Table 4.1. For each state, the median number of native contacts is determined. States with the median number of native contacts above a folding threshold are assigned as the folded states. States with native contacts above an unfolding threshold are set as the unfolded states.

The clustering results in microstate trajectories. The reference MSMs are generated by maximum-likelihood estimation under detailed balance constraint. The lag time $\tau$ for MSM generation was selected based on the plateau of implied timescales. The Markov property of the resulting MSM was tested by Chapman-Kolmogorov validation. All this analysis was executed with the PyEMMA package [65]. Details of the MSM generation are described in the paper [1]. To generate Markov Chain trajectories, synthetic microstate trajectories are generated according to the transition probabilities of the MSM transition matrix.

### 4.1.3 Simulating adaptive sampling with Markov Chain trajectories

The simulation of adaptive sampling with Markov Chain trajectories follows the schematic in Chapter 3 Figure 3.3. Instead of simulating an ensemble of $n$ MD trajectories an ensemble of $n$ Markov Chain trajectories is simulated. The required time for simulating Markov Chain trajectories is orders of magnitude smaller than simulating equivalent MD trajectories. This reduced time allows to simulate many repetitions of adaptive sampling and for multiple proteins.

The investigated sampling strategies are plain MD as reference, and *cmicro* strategy, the 4 *cmacro* strategy variants, the two adaptive sampling strategies with *a priori* information $Q_f$ and $Q_{f,nn}$ and the two optimal strategies for adaptive sampling performance $p_{esc}$ and $t_{opt}$. These strategies were described in Chapter 3. At the end of each iteration of each strategy, a new set of $n$ restarting points are chosen. The restarting points were chosen based on the adaptively sampled MSM. The iteratively updated adaptively sampled MSM represents only the explored transitions captured in the sampled Markov Chain trajectories.

Two of the *cmacro* strategy variants include the Koopman method to improve the accuracy of the Markov State model. The Koopman method applies directly to the molecular dynamics trajectory, which is not available here. To estimated the effect of the Koopman methods on adaptive sampling, the adaptively sampled MSM is corrected to the transition probabilities from the reference MSMs. This approach gives an upper limit on the improvement of adaptive sampling the Koopman method can achieve.

## 4.2 Results

Once the adaptive sampling is generated for each protein, each sampling strategy and is repeated 100 times, the sampling can be evaluated according to different sampling objectives. Here we quantify the adaptive sampling performance by two sampling objectives. The first objective is the time to cross the rare transition barrier or fold the protein. The second objective is the time to explore 95% of the states, representing the exploration of the whole energy landscape. In each case, the average and the 20% and 80% percentiles are measured and compared with the other sampling strategies, including plain MD. A parallelization $n$ of 100 replicas is chosen, except for the scaling comparison where the parallelization $n$ ranges from 1 to 5000.

### 4.2.1 Time to fold

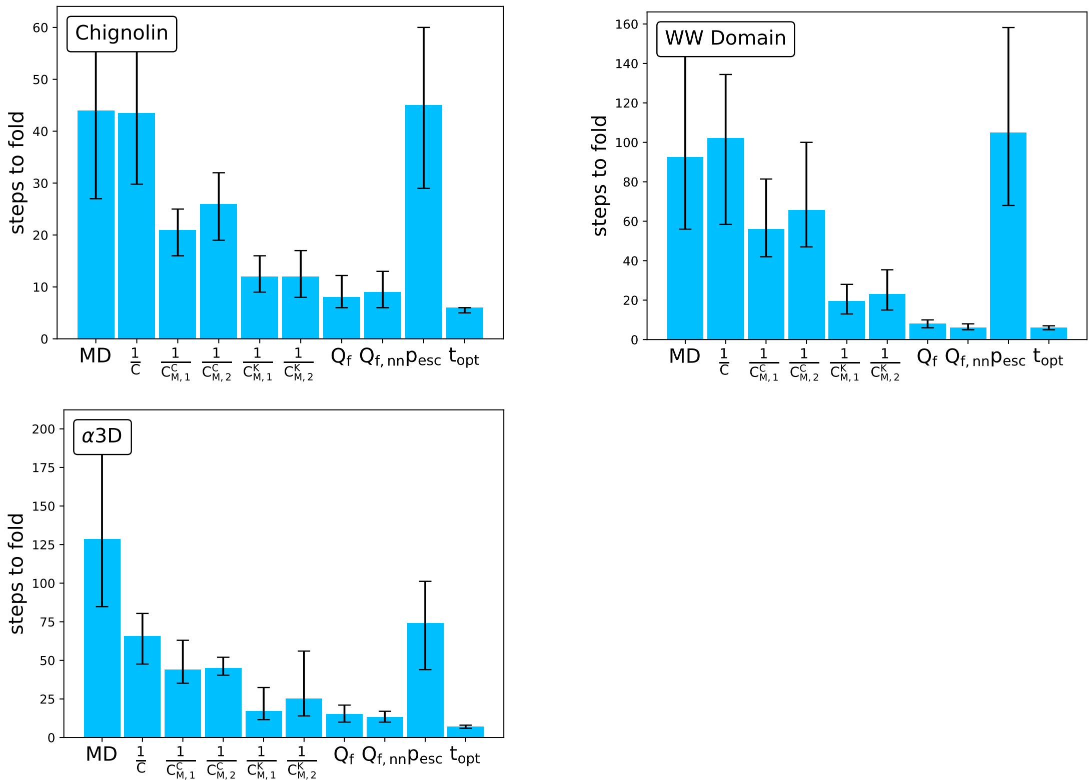


Figure 4.1 : Time to fold for the different sampling strategies for proteins Chignolin, WW Domain and $\alpha$3D.

The time to fold the protein is the most common objective of adaptive sampling. Figure 4.1 shows the performance of different adaptive sampling strategies with Markov Chain trajectories. Each sampling strategy should be compared with plain MD, which is the reference sampling strategy. The simple $cmicro/1/C$ strategy shows no speedup for Chignolin, WW Domain, and only a small 2x speedup for $\alpha$3D. This shows that the $cmicro$ strategy is not optimal for folding proteins. The reason for the low efficiency lies in the oversampling of states which are not on the folding path but have a low stationary probability and high restart probability.

The *cmacro* strategies show a significant speedup compared to plain MD for all the proteins, but the 4 variants of the *cmacro* strategies show significant performance differences. Between the 4 variants of the *cmacro* strategies, it seems the correction of the MSM due to non-equilibrium $1/C_{K,1}$ improves the results most. While the approximation of the Koopman method with the Markov Chain trajectories is only an upper limit for the performance improvement, the improvement is 2-3x. The improved performance of the corrected MSMs is caused by the improved estimation of the collective motions of the proteins. The estimated macrostates are kinetically more accurate. The 2 variants in setting the number of macrostates don't show a substantial performance effect. The constant number of macrostates $1/C_{M,1}$ is slightly faster. The underperformance of the kinetically set number of macrostates could be caused by the challenge of determining the timescales for partially sampled energy landscapes accurately

The two adaptive sampling strategies with *a priori* information $Q_f$ and $Q_{f,nn}$ show significant improvement over $1/C_{K,1}$, which is the best strategy without *a priori* information. If the *a priori* information is accurate, then this can speedup the folding of the protein significantly. Depending on the protein the improvement with the *a priori* information can be less than 50 % or a factor 2-3x. The effectivity of including additional *a priori* information to prevent misfolded states in $Q_{f,nn}$ varies on the protein, but no significant performance penalties are observed.

The two benchmark strategies $p_{esc}$ and $t_{opt}$ show very different performance. The strategy optimized to fold proteins $t_{opt}$ shows the upper limit for protein folding speedup with adaptive sampling. This benchmark shows a 2x or more improvement compared to strategies with no *a priori* information, allowing a gap for the improvement of adaptive sampling strategies. Previously the upper limit for performance of protein folding with adaptive sampling was unknown. The $t_{opt}$ strategy is in some proteins close to the performance of $Q_{f,nn}$, the best strategy with *a priori* information. The similar performance can be caused due to the simple protein folding pathway. For other proteins, the $t_{opt}$ strategy is significantly faster than $Q_{f,nn}$, showing that adaptive sampling strategies with *a priori* information have opportunities to improve. The strategy optimized to explore all states with an greedy algorithm $p_{esc}$ is not effective in folding proteins. The performance of the adaptive sampling strategies is consistent between different proteins, allowing to conclude that adaptive sampling is effective for different topologies, both $\alpha$-helices and $\beta$-sheets.

### 4.2.2 Time to explore 95% of states

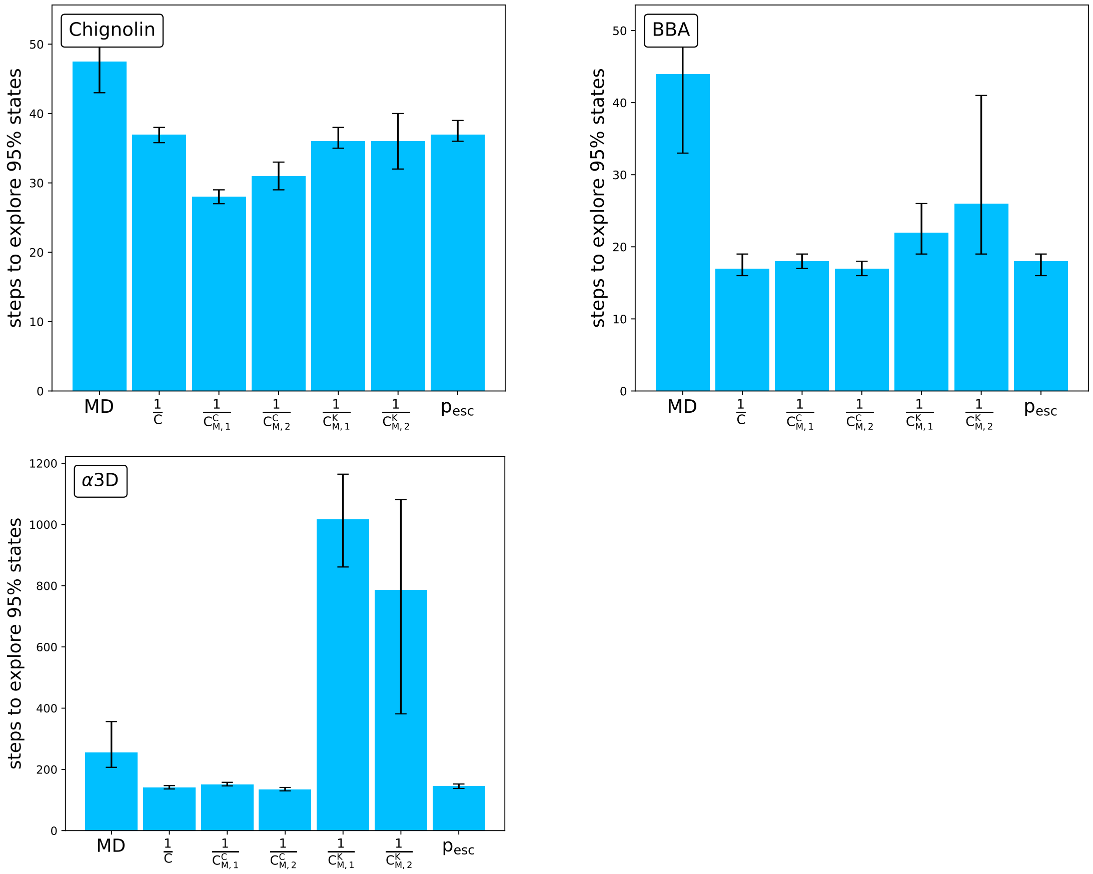


Figure 4.2 : Time to the exploration of 95% of the states for the different sampling strategies for proteins Chignolin, BBA and $\alpha$3D.

The time to explore the whole energy landscape is an alternative metric to evaluate the effectiveness of adaptive sampling. Figure 4.2 shows the average time needed to explore 95% of the microstates for different adaptive sampling strategies with Markov Chain trajectories. The results show that the adaptive sampling strategies show a different performance for this metric. Comparing to the protein folding metric, the *cmicro* strategy performs better for the exploration metric, and the *cmacro* strategy performs relatively worse for the exploration metric. Despite the worse performance of the *cmacro* strategy, for some proteins like Chignolin, it's the fastest strategy.

For some proteins such as Chignolin, the $p_{esc}$ strategy is slower than the $(1/C_M^C)$ strategy. This is caused by the rare transition barrier in the folding landscape. The $p_{esc}$ is a greedy algorithm and doesn't take the rare transition barrier into account. The $t_{opt}$ strategy was not shown in Fig. 4.2, since this strategy never explores 95% of the states. This is expected since this strategy is optimized for a different objective. Surprisingly, the *cmacro* strategies with regular MSM $(1/C_M^C)$ outperform the strategies with corrected MSM $(1/C_M^K)$. This unexpected result is probably caused by the correction, which biases the sampling towards slow processes and away from general exploration. It's unclear which variant of the *cmacro* strategy for the number of macrostates performs better due to the variance of performance between the proteins. For the metric of exploration, the strategies *cmicro* or $(1/C_M^C)$ are preferable.

### 4.2.3 Scaling

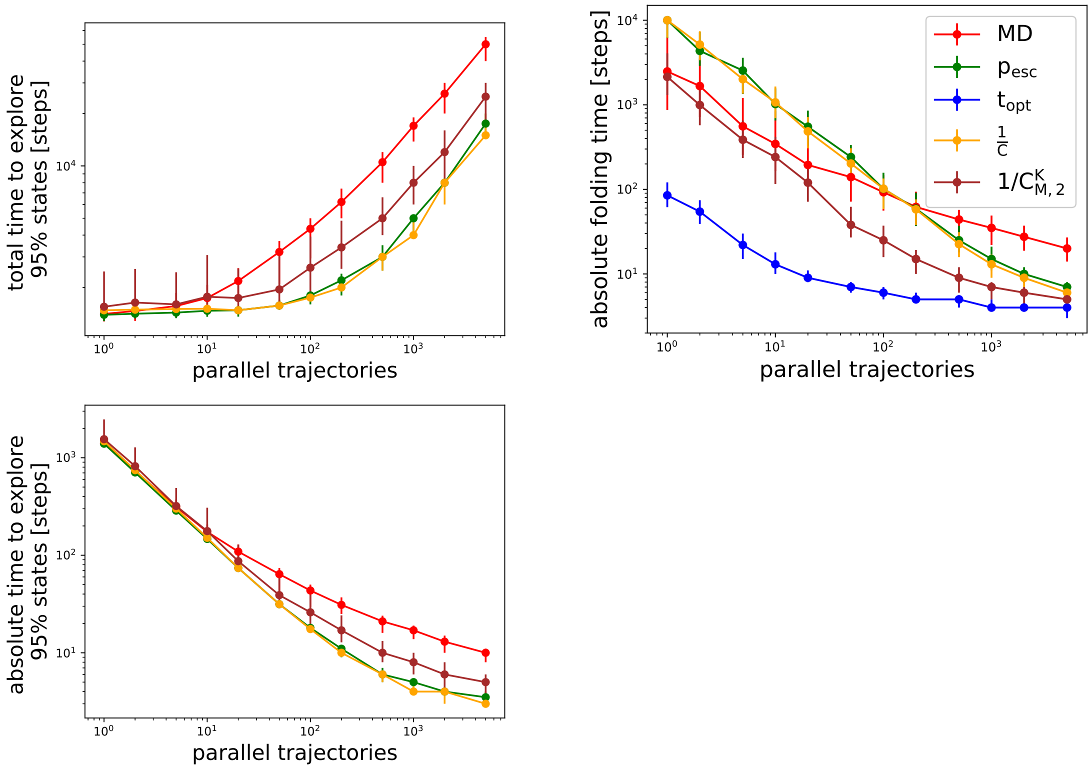


Figure 4.3 : Top] Scaling of the absolute (left) or cumulative (right) number of steps required to explore 95% of all microstates. Strategies plain $MD$, $p_{esc}$, $1/C$ and $1/C^{K}_{M,2}$ for the protein BBA. Bottom] Scaling of the absolute folding time for sampling strategies; plain $MD$, $p_{esc}$, $t_{opt}$, $1/C$ and $1/C^{K}_{M,2}$ for WW Domain.

The scalability of adaptive sampling is essential for both the application and understanding of adaptive sampling. Adaptive sampling is usually deployed on High-Performance Computers, which have a high parallelization available. Parallelization allows to shorten the time to solution by dividing the work among different GPUs, but this approach is only effective when adaptive sampling scales well. The metric absolute time represents the time to solution, but this metric is hardware and software independent and measures the length of consecutive MD simulations. The cumulative/total time represents the computational resources necessary and is hardware and software independent as well and measures the sum of all lengths of MD simulations. Figure 5.4 shows how the time to folding and time to exploration depends on the parallelization. The absolute time decreases rapidly up to parallelization of 100-1000. Any higher parallelization will not decrease the time to solution. This scaling limit depends on the protein; for larger proteins, it's slightly higher. All adaptive sampling strategies scale similarly, but plain MD scales worse. This worse scaling of plain MD is explained by the lack of communication between the individual MD trajectories for plain MD. In contrast, adaptive sampling shares information between the different replicas, which allows adaptive sampling to scale better. The total time to explore shows that the computational resources necessary to sample a protein increase rapidly above the scaling limit of 100-1000. This increase of total time indicates a lower efficiency for utilization of the GPUs but could be traded off to reduce the time to solution. For plain MD, the computational resources necessary increase above 10 replicas or GPUs.

### 4.2.4 Speedup for different proteins

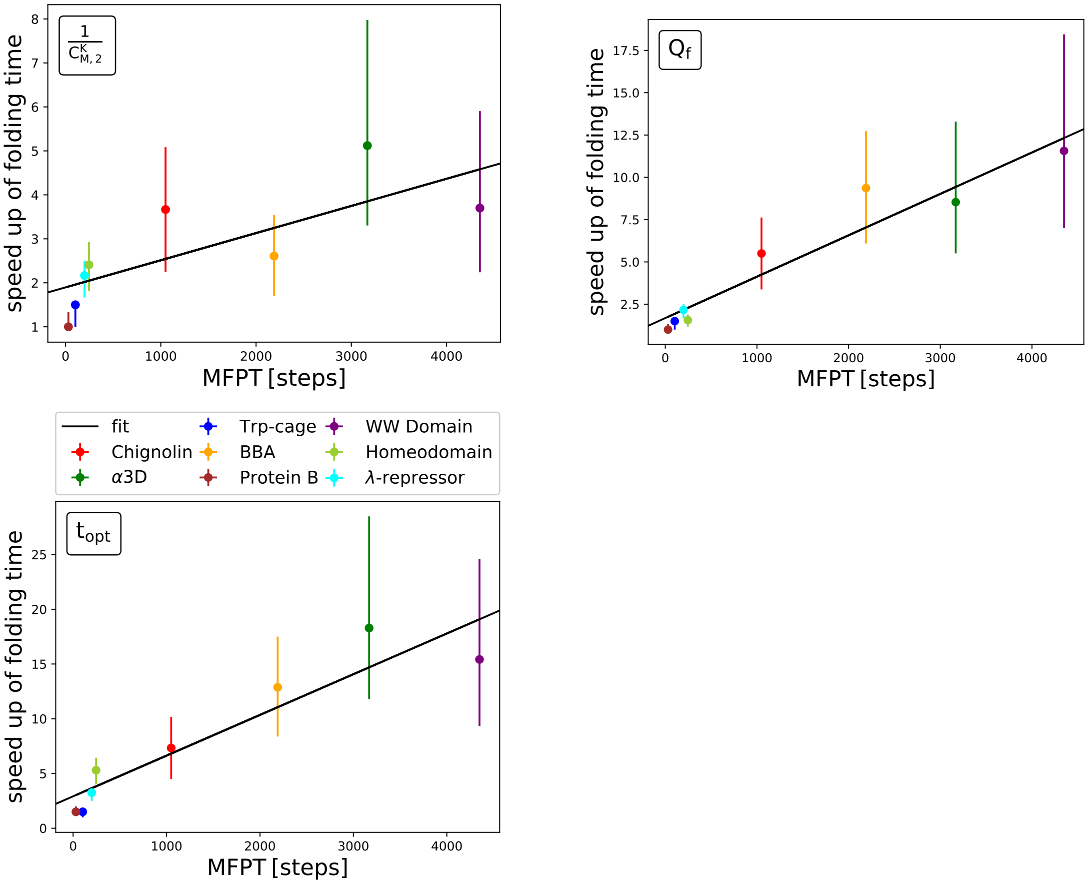


Figure 4.4 : Correlation between the speedup of the folding time vs. mean first passage time for the 8 proteins and adaptive sampling strategies $1/C^{K}_{M,2}$, $Q_f$ and $t_{opt}$.

The previous figures showed a significant variance in performance between individual proteins. Plotting the speedup of folding time relative to plain MD in Figure 4.1 allows understanding how protein size and folding time affect the adaptive sampling speedup. Despite the limited number of proteins investigated due to limited reference data, a relationship for adaptive sampling speedup is visible. The strategies shown, $1/C^{K}_{M,2}$, $Q_f$ and $t_{opt}$, increase the speedup with folding time for the protein. This dependency is roughly linear, the Pearson correlation coefficients are 0.82 for $1/C^K_{M,2}$, 0.95 for $Q_f$ and 0.93 for $t_{opt}$. The optimal strategy $t_{opt}$ and strategy with *a priori* information have a higher correlation coefficient. The upper limit strategy for folding $t_{opt}$ reaches a speedup about 5 times of the strategy $1/C^K_{M,2}$ and about 50% higher than $Q_f$. This shows an opportunity for improved adaptive sampling strategies. All the proteins are relatively small due to limited reference data, but the increasing speedup for longer folding and larger protein allows to predict that larger proteins can reach speedups above 10x with $1/C^K_{M,2}$. The maximal possible speedup with adaptive sampling is not known.

## 4.3 Discussion

This systematic analysis of adaptive sampling strategies allows comparing different strategies with an improved statistical sampling. The effectivity of correcting the MSM for non-equilibrium sampling was demonstrated by the performance of $1/C^K_{M,2}$ for folding proteins. This result is limited by the accuracy of the approximation, which was used to estimate the performance of correcting the MSM with the Koopman method. For folding a protein, the $1/C^K_{M,2}$ strategy is preferable, reaching a speedup of up to 5x for the investigated small proteins. The most efficient adaptive sampling strategy is based on the iterative identification of kinetically disconnected macrostates with corrections for non-equilibrium effects.

Another result is that a different goal of the sampling, such as exploring all the states, dramatically affects the speedup with adaptive sampling strategies. For exploration *cmicro* or $(1/C^C_M)$ strategies are preferable. An unexplored consideration is the utilization of multiple strategies. At the beginning of adaptive sampling, the folding of the protein could be prioritized; later the exploration or increasing the accuracy of the kinetic results could be prioritized.

The increased effectivity of adaptive sampling strategies with *a priori* knowledge about the biomolecules has been confirmed. In some cases, the speedup reaches the upper limit determined by the $t_{opt}$ strategy. These strategies can only be used when the *a priori* information is accurate. Inaccurate or imperfect intuitive reaction coordinates can lead to a slower sampling of the biomolecule.

The newly developed upper limit strategy for folding time $t_{opt}$ shows the maximal speedup with adaptive sampling, which is around 15x for the investigated small proteins. Increased speedups are predicted for larger and slower folding proteins. The second upper limit strategy for exploration $p_{esc}$ is not a good benchmark since, in some cases, the *cmacro* strategy performed better. This is caused by the greedy algorithm nature of this upper limit strategy.

The significant performance gap between the $1/C^{K}_{M,2}$ strategy, and the upper limit $t_{opt}$ shows an opportunity for more effective adaptive sampling strategies.

Unexpectedly, plain MD simulations with a larger number of replicas than 100 increases the required computational resources, which shows a disadvantage of plain MD compared to adaptive sampling. Adaptive sampling scales well to 100-1000 replicas; this extends the results of [28].

The results in this Chapter are limited by the approximation of molecular dynamics trajectories by Markov chain trajectories. This approximation allowed us to increase the sample size of adaptive sampling by several orders of magnitude but also introduces the inaccuracies of the generated MSM model. These inaccuracies of the MSM models were limited by the careful generation of the MSM models. Additionally, the effect of the Koopman method could only be approximated.

These results are encouraging adaptive sampling for a higher number of replicas and larger, more complex proteins. Currently, adaptive sampling is limited by the significant computational resources to fold larger proteins even with the speed up of adaptive sampling. The cautious extrapolation of the adaptive sampling speedup for protein folding allows predicting speedup in excess of 10-times for larger proteins.

# Chapter 5

# Adaptive sampling software framework

## 5.1 Introduction

One of the motivations of utilizing *Adaptive sampling* is reaching longer timescales in the practical application of sampling biomolecules and obtaining accurate kinetic behavior. This requires to execute a hierarchical ensemble of tasks in an efficient fashion on every level. Without efficient execution of all the subtasks of adaptive sampling, the goal of reaching longer timescales wouldn't be realistic. In practice, the higher complexity of executing adaptive sampling compared to plain molecular dynamics reduces the practical utilization of the advantages of adaptive sampling. Here we design and develop a software framework that reduces the entry barrier to implement adaptive sampling by domain experts. Some of the objectives establishing the software frameworks are scalability, maintainability, flexibility, and extensibility. The scalability objective is fundamental to efficiently utilize the limited computational resources on High-Performance Computers (HPC). Easy maintainability should reduce the entry barrier to execute adaptive sampling. The flexibility and extensibility of this software framework should enable adjusting for new sampling methods, individual molecular dynamics software choices, and different HPCs or supercomputers. In this Chapter we will discuss the development of the *ExTASY* software package and how ExTASY implements the beforementioned objectives.

The material in this Chapter was first published in the following papers:

[2] Balasubramanian, V.; Bethune, I.; Shkurti, A.; Breitmoser, E.; **Hruska, E.**; Clementi, C.; Laughton, C.; Jha, S.; Extasy: Scalable and flexible coupling of md simulations and advanced sampling techniques. Proceedings of the 2016 IEEE 12th International Conference on e-Science 361370 (2016).

[3] **Hruska, E.**; Balasubramanian, V.; Ossyra, J. R.; Jha, S.; Clementi, C.; Extensible and scalable adaptive sampling on supercomputers. arXiv (2019). URL: https://arxiv.org/abs/1907.06954.

## 5.2 Alternative software

Previously multiple groups have developed software packages that strive to achieve the same objectives for the execution of adaptive sampling. Still, all these frameworks have some limitations which we attempt to overcome with *ExTASY*. The package HTMD [32] has shown the effective adaptive sampling performance for small proteins, including effective retrieval of kinetic information. The performance of HTMD for larger proteins and it's scalability have not yet been shown. The entry barrier for the utilization of HTMD is increased by not being open-sourced. Further, HTMD supports only one single molecular dynamics engine and a limited number of adaptive sampling strategies.

The recent package DeepDriveMD [63] allows to utilize deep learning in adaptive sampling, and it is able to execute on heterogeneous software and hardware environments scalably. Still, DeepDriveMD has not yet released or open-sourced the code yet, or shown application to larger proteins.

The SSAGES (Software Suite for Advanced General Ensemble Simulations) [66] is open-source and flexible, not bound to specific molecular dynamics engines. Still, the effectivity and scalability for large proteins remain to be shown. Multiple software frameworks [67–69] show the effective sampling of toy systems and small peptides with the ability to utilize strategies with neural networks but haven't shown the extensibility to larger systems or the scalability to perform efficiently for larger applications.

## 5.3 Asynchronous execution

One crucial feature of ExTASY is the ability for asynchronous execution of the individual subtasks. As far as we know, no other adaptive sampling platform enables asynchronous execution. This significantly improves the efficiency of adaptive sampling in practice. In the standard, synchronous execution, all previous steps have to be concluded to begin the next step. The Steps 2-4 mentioned in Chapter 3 Figure 3.3 require very low parallelization compared to Step 1. Steps 2-4 shows the analysis, and Step 1 shows the execution of all molecular dynamics trajectories. In a typical use case, Step 1 utilizes a large number of GPUs due to the large number of replicas simultaneously simulated. Once Step 1 finishes, then Steps 2-4 are run, which typically utilizes a fraction of nodes or GPUs compared to Step 1. When running on a constant number of nodes or GPUs, which is the most common case on HPCs, this low utilization of nodes for Step 2-4 is reduced. This performance penalty is more substantial when the Steps 2-4 take a longer time to finish, for example, for a more complex strategy or larger biomolecules. The asynchronous execution shown in Figure 5.1 allows to increase the utilization of computational resources and consequently to reach longer timescales by continuously utilizing all nodes or GPUs and removing the downtimes for MD workers. The MD tasks require new restarting conformations to start the simulations. These new restarting conformations are continuously updated by repeatedly running the adaptive sampling analysis steps. This generated new restarting conformations, which take into account the recently partially unfinished molecular dynamics tractories. Due to the time delay of finishing adaptive sampling analysis, a small fraction at the end of MD trajectories won't be analyzed, but in the next restarting point, these small parts will be analyzed. The first version of ExTASY [2] had only synchronous execution enabled, but the current version [3] includes asynchronous execution. In chapters 5 and 6, all the adaptive sampling is executed asynchronously.

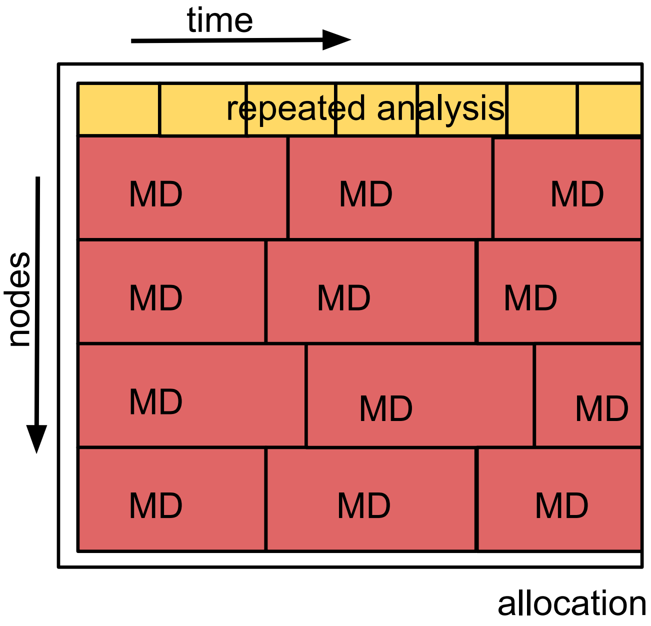


Figure 5.1 : Asynchronous execution of an ensemble of adaptive sampling subtasks, including molecular dynamics and analysis tasks. The restart conformations for the next molecular dynamics simulation are determined by the last finished analysis task. Here steps 2 and 4 are shown combined as analysis steps.

## 5.4 Tools and Software

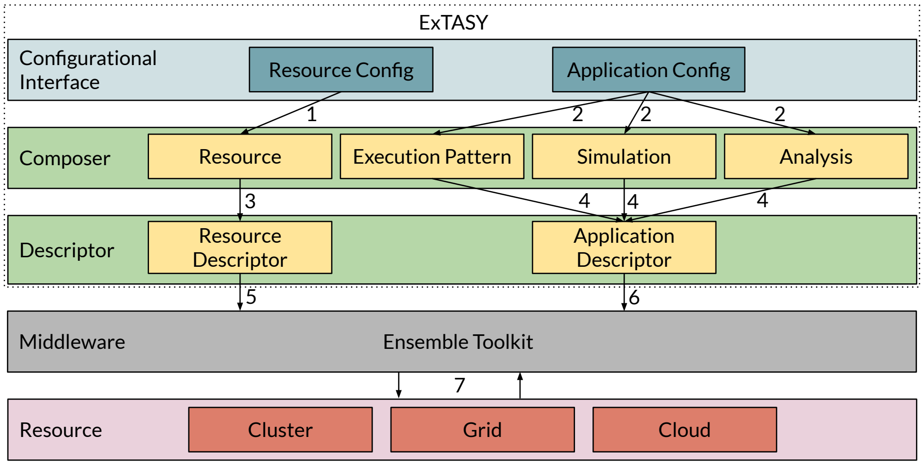
ExTASY was built by utilizing the RADICAL-Cybertools [70]. The RADICAL-Cybertools are software systems designed to execute ensembles of tasks on HPC platforms in a scalable and modular approach.

ExTASY uses the RADICAL-Cybertools component EnTK (Ensemble Toolkit) [71, 72] to communicate with RADICAL-Cybertools. The EnTK layer enables to abstract the execution of individual tasks from the explicit resource management. In the background, hidden from ExTASY, RADICAL-Cybertools performs the explicit resource acquisition and resource management in a scalable fashion as well as error tracking and analytics. The whole RADICAL-Cybertools was built in a blocks approach [73], allowing modularity and flexibility for ExTASY.

Figure 5.2 shows pseudo-code illustrating the ExTASY implementation of adaptive sampling with EnTK API. The individual separation of simulation and analysis tasks is recognizable. Figure 5.3 shows the modular integration between ExTASY and EnTK. ExTASY describes adaptive sampling as a set of executable tasks. ExTASY determines the parameters for the Resource description (event 1) and the MD and analysis tasks (event 2). ExTASY then passes these parameters to the EnTK's interface, which defines the resource and application (events 3 and 4). Once defined, EnTK starts the execution of the executable ExTASY tasks on the target resource (events 5, 6, and 7).

```
from radical.entk import Task, Stage, Pipeline
p = Pipeline()

sim_stage = Stage()
sim_task = Task()
sim_task.executable = <executable> #example openmm
sim_task.arguments = <args> #example openmm args
<add other task properties>
sim_stage.add_tasks(sim_task)

ana_stage = Stage()
ana_task = Task()
ana_task.executable = <executable> #example pyemma
ana_task.arguments = <args> #example pyemma args
<add other task properties>

ana_stage.add_tasks(ana_task)
ana_stage.post_exec = {
    'condition': eval_sims(),
    'on_true':   add_sims(),
    'on_false':  terminate()
    }

p.add_stages([sim_stage, ana_stage])
```

Figure 5.2 : Pseudocode showing the modular software design and the EnTK API backend


ExTASY
Configurational Interface
Resource Config
Application Config
1
2
2
2
Composer
Resource
Execution Pattern
Simulation
Analysis
3
4
4
4
Descriptor
Resource Descriptor
Application Descriptor
5
6
Middleware
Ensemble Toolkit
7
Resource
Cluster
Grid
Cloud

## 5.5 Pilot abstraction

One reason for ExTASY to utilize RADICAL-Cybertools is the improved ability to execute tasks on computational resources. Traditionally multiple HPC tasks can be executed either as individual jobs or with message-passing interface (MPI) as part of a single job. The first approach is effective for independent tasks, but for interdependent tasks as in adaptive sampling, this approach is suboptimal. The second method is suboptimal for the heterogeneous and adaptive tasks in adaptive sampling. The pilot abstraction in RADICAL-Cybertools [74] overcomes these limitations. The pilot abstraction separates the initial computational resource acquisition from individual task-to-resource assignments. First, the computational resources are requested with placeholder jobs without any tasks. In the second step, while the job is running, individual tasks are assigned to the placeholder jobs. RADICAL-Cybertools is engineered to support scalable pilot abstraction for launching heterogeneous tasks across different platforms. This pilot abstraction of the RADICAL-Cybertools allows ExTASY to execute all the individual subtasks of adaptive sampling effectively.

## 5.6 Scaling

One of the main metrics to deploy tools to Supercomputers is the scalability. Only algorithms with sufficient parallelization can utilize these computational resources effectively. The scalability of adaptive sampling depends both on the scalability of the software platform and the scalability of the individual MD engines and analysis tools. To abstract only the scalability of the software platform, we define the efficiency as the ratio between the nodehours used by all the individual tasks and the nodehours used by the software platform. This efficiency was measured for increasing parallelizations up to 2000 GPUs on Summit, shown in Figure 5.4. With the asynchronous execution, one GPU was tasked to run analysis tasks; all other GPUs were running MD tasks. 6 GPUs per node and 2 hour-long computational jobs were utilized. The efficiency of ExTASY above 2000 GPUs is reduced due to the time delay caused by communication between individual RADICAL components. RADICAL-Cybertools adapts to the specific software environments of the HPCs [75], improving the scalability and illustrating the advantage of platform-agnostic execution and portability across heterogeneous HPC resources. The asynchronous version of ExTASY allows scaling to a higher number of nodes than alternative adaptive sampling software platforms.

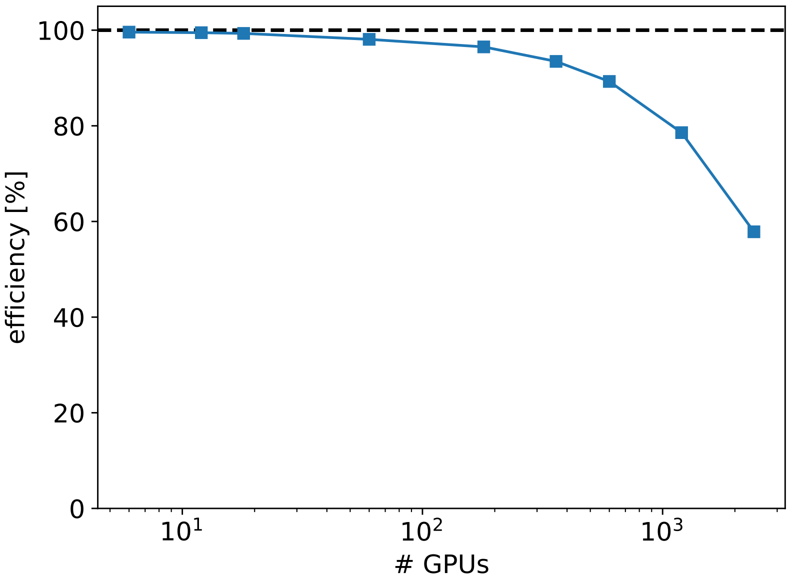


Figure 5.4 : Scaling of ExTASY on Summit. The Efficiency is defined as the ratio between the nodehours used by all the individual tasks and the nodehours used by the software platform.

## 5.7 Conclusion

This thesis shows the development of the ExTASY framework, which achieves several objectives to enable effective execution of adaptive sampling workloads. Scalability of over 1000 GPUs and 1000 replicas was demonstrated on the supercomputer Summit. This scalability reduces the technical barrier to adaptive sampling for larger proteins. Notably, this scalability doesn't reduce the flexibility of the platform.

The open-source nature of the ExTASY framework is an important feature. The code is available at https://github.com/ClementiGroup/ExTASY. By releasing the platform open-source, the implementation of new adaptive sampling strategies is simplified. This flexibility allowed ExTASY to be the first open-source, adaptive sampling platform that supports deep learning or asynchronous execution. The asynchronous execution significantly improves the utilization of computational resources for larger proteins or more complex adaptive sampling strategies.

The modularity of this software platform enables multiple objectives. One objective is the agnosticism from the HPC platform, preventing code working only for specific software and hardware environments. The modularity also improves the maintainability, reliability, and reproducibility of adaptive sampling.

# Chapter 6

# Application of adaptive sampling

The development of an effective, scalable software platform in Chapter 5 allows us to execute adaptive sampling for more extensive applications and larger proteins. In this chapter, adaptive sampling will be used to determine the folding and conformational dynamics of 4 proteins. These results allow us to compare the efficiency and accuracy of adaptive sampling compared with plain molecular dynamics. These 4 proteins range from 10 residues to 73 residues. The computational resources available limited the application to these small, fast folding proteins.

The material in this chapter was first published in the following paper:

[3] **Hruska, E.**; Balasubramanian, V.; Ossyra, J. R.; Jha, S.; Clementi, C.; Extensible and scalable adaptive sampling on supercomputers. arXiv (2019). URL: https://arxiv.org/abs/1907.06954.

## 6.1 Introduction

In Chapter 4, the efficiency and reliability of adaptive sampling strategies were compared in a statistically significant approach. In that chapter, Markov Chain trajectories were utilized to approximate the behavior of molecular dynamics trajectories. Since Markov Chain trajectories can be generated much faster than molecular dynamics trajectories, statistically more significant results could be achieved. The limitation of this approach is that Markov Chain trajectories are only approximations, and full molecular dynamics trajectories are necessary to confirm the performance of adaptive sampling. From Chapter 4, we expect a higher speedup with adaptive sampling [1] compared to plain MD for larger proteins.

## 6.2 Reference Data

To compare the speedup and accuracy of the adaptive sampling methods compared to plain MD we utilize reference proteins and reference trajectories to compare the results. The 4 small proteins have long reference simulations generated on the Anton supercomputer [64], which fold and unfold each of the proteins multiple times. The multiple folding events ensure a well-sampled energy landscape and a good accuracy for the reference comparison. To fold and unfold proteins multiple times requires large computational resources, leaving only a very limited number of proteins as well sampled reference data with well-sampled energy landscapes and conformational dynamics.

| Protein | PDB ID | # of Residues | Folding Time ($\mu$s) [64] | Unfolding Time ($\mu$s) |
|---|---|---|---|---|
| Chignolin | 5AWL | 10 | 0.6 | 2.2 |
| Villin | 2F4K | 35 | 2.8 | 0.9 |
| BBA | 1FME | 28 | 18 | 5 |
| A3D | 2A3D | 73 | 27 | 31 |

Table 6.1 : Proteins for the comparison of adaptive sampling and plain MD.

The folding and unfolding times of the 4 proteins are shown in Table 4.1. All the folding times are below 40 $\mu$s, a timescale reachable with the available resources. To compare the adaptive sampling and plain MD we ran for all proteins both methods with ExTASY on the Summit supercomputer. To ensure comparability, the parallelization was the same for adaptive sampling and plain MD.

## 6.3 Setup

The initialization of both plain MD and adaptive sampling was identical. For the 4 proteins, a single start frame was chosen from the reference dataset. To ensure this frame was unfolded, random selection was only among the 20% frames with the highest Root Mean Square Deviation to the protein crystal structure. The initial coordinates for ExTASY were obtained after a short energy equilibration (1-2ns) in an NPT ensemble.

The parameters for the MD simulations were following. The setup of the reference trajectories [64] was replicated, with a CHARMM22* force field [4] and the modified TIP3P water. Table 6.2 shows the identical temperature of the MD simulation to the reference simulations. One difference was the use of the Particle Mesh Ewald method for long-range electrostatics in OpenMM. The MD stepsize was 5fs and 2fs for A3D. The MD was run on OpenMM 7.5 [76] using CUDA 9.1.

For both adaptive sampling and plain MD, 50 replicas were used, each replica simulated on one GPU. The MD trajectory lengths were 50ns for Chignolin and Villin, 10ns for BBA, and 40 ns for A3D. The resulting trajectories were strided to reduce the data volume. Two copies of each trajectory were generated. One trajectory stored all information, including water and one trajectory stored only the protein coordinates. The protein-only trajectories were used to speed up the analysis, which was limited by the data transfer rate.

The analysis for adaptive sampling was asynchronously executed on one GPU, as shown in Chapter 5. The parameters for the analysis step of the adaptive sampling strategies were the following. For dimension reduction, deep learning with SRVs was utilized, as introduced in Chapter 2. The SRV features were the distances between all pairs of C-alpha atoms. For protein A3D, only the distance pairs between every second C-alpha atom were selected due to the larger size of the protein. For proteins Chignolin, Villin, and BBA, the SRVs dimension reduction was run with 8 deep learning epochs, 6 hidden layers, size of hidden layers 50, activation tanh, learning rate 1e-4, dropout rate 0.1 and latent space noise 0.1. For the larger protein A3D, the SRV was sped up with the following parameters: 1 deep learning epoch, 2 hidden layers, size of hidden layers 160, activation function elu (Exponential linear unit), learning rate 1e-3, dropout rate 0.01 and latent space noise 0.1. The number of SRV output dimensions was 10 for Chignolin and 4 for Villin, BBA, and A3D.

The SRV lag times and MSM lag times are shown in Table 6.2. For BBA, short lag times were chosen to investigate the effect of shorter trajectory lengths. These shorter trajectory lengths were enabled by using SRVs instead of TICA. SRVs reach identical accuracy with shorter time lags than TICA dimension reduction [16]. The number of k-means clustering states for the MSMs was 500 and 200 for protein A3D. For Chignolin and A3D, the chosen adaptive sampling strategy was *cmacro*, for Villin and BBA *cmicro* adaptive sampling strategy was used. For the *cmacro* strategy, 20 macrostates were used.

Once both the adaptive sampling and plain MD was concluded, the exploration and the protein dynamics were compared with the help of the Anton MD reference trajectories. For the final analysis, TICA dimension reduction with kinetic map scaling and Koopman correction was used with a lag time of 10ns or 1.5ns for BBA. The TICA input features were the distances and inverse distances between all pairs of C-alpha atoms. After the projection of both the adaptive sampling and plain

MD trajectories on the reference trajectories in the TICA space, the exploration and protein dynamics were compared.

| Protein Name | MD Temperature [K] | MD [ns]/iteration | MD step size [fs] | SRV lag [ns] | MSM lag [ns] |
|---|---|---|---|---|---|
| Chignolin | 340 | 50 | 5 | 1.5 | 10 |
| Villin | 360 | 50 | 5 | 1.5 | 10 |
| BBA | 325 | 10 | 5 | 1.5 | 1 |
| A3D | 370 | 40 | 2 | 10 | 10 |

Table 6.2 : ExTASY parameters for MD and analysis, for both plain MD and adaptive sampling.

## 6.4 Results

The three primary considerations when comparing adaptive sampling and plain MD are the speedup achieved, exploration of the whole energy landscape, and the accuracy of the conformational dynamics. The exploration of the whole energy landscape was shown by projecting the adaptive sampling onto the reference trajectories in the TICA space. By measuring the explored fraction of the population vs. simulation time, the speedup compared to plain MD could be determined. The accuracy of the protein dynamics was determined by analyzing the entropy of the MSM transition matrices and the Mean First Passage Time (MFPT).

### 6.4.1 Comparison of Exploration

The high-dimensional nature of the whole energy landscape of biomolecules prevents the visualization and comparison of the whole energy landscape. Only the dimension reduced free energy landscape of the proteins in TICA coordinates was compared, shown in Figure 6.1. The background in color shows the reference free energy landscape of the proteins, and the shaded foreground is the area which adaptive sampling successfully explored. For all four proteins, Chignolin, Villin, BBA, and A3D, the whole energy landscapes were explored by adaptive sampling, including the folded state. This shows that adaptive sampling not only folded these proteins but correctly recovered the energy landscape for these proteins. The small differences compared to the reference energy landscapes predominantly occur in the rare sampled areas of the energy landscapes. This discrepancy can be explained by the stochasticity of both the adaptive sampling and reference MD simulation. Despite the relatively long data size, the stochasticity caused sparse sampling on the energy landscape in some low probability areas of the energy landscapes.

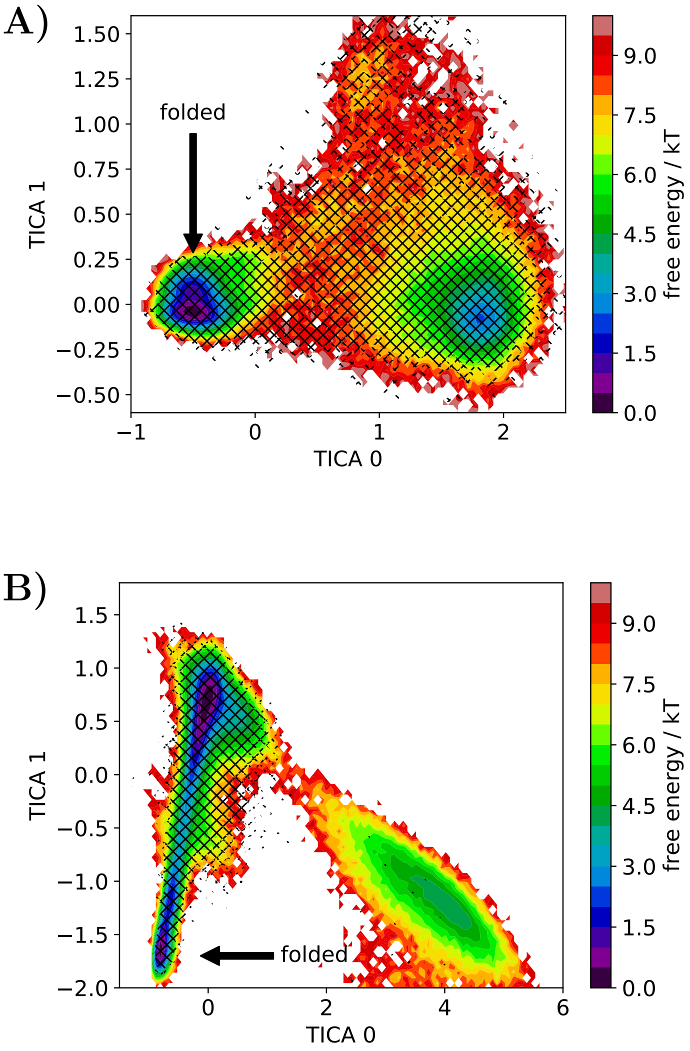


Figure 6.1 : The recovery of the energy landscape by adaptive sampling for 4 proteins. The projection on TICA coordinates is shown. The color background shows the reference Free Energy landscape, and the black diagonal lines show the adaptive sampling explored energy landscape. The location of the folded states are shown. A) Chignolin B) Villin

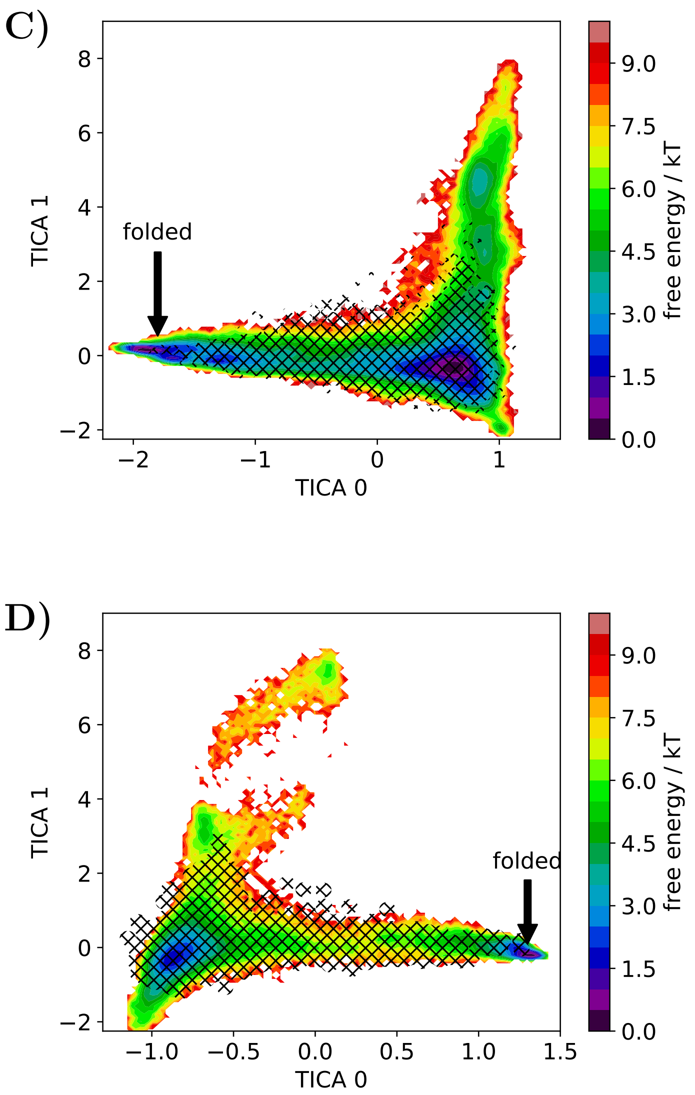


Figure 6.1 : (cont.) C) BBA D) A3D

To compare the speed of exploration between adaptive sampling and plain MD, the fraction of explored population over time can be compared, as shown in Figure 6.2. The time here is measured in absolute simulation time. The absolute simulation time is the length of consecutive molecular dynamics simulations, omitting any parallel simulations. For example, the absolute simulation time is the length of one single trajectory multiplied by the number of iterations. This metric is directly related to the time to solution, which is in most cases the relevant criterium if an adaptive sampling can be performed. The absolute simulation length is hardware-independent, which simplifies the comparison, in contrast to time to the solution, which heavily depends on hardware and software choices. Chapter 4 explored the effect of parallelization on the absolute simulation time; in this chapter the parallelization is held constant.

The fraction of explored population is measured as a function of increasing absolute simulation time during an exploration, either for adaptive sampling or for plain MD. The populations and states for the calculation of the explored population are defined by the reference datasets. The fraction of the explored population is calculated by determining which states were explored and weighting with the stationary weight from the reference dataset.

ExTASY can flexibly implement different adaptive sampling strategies. To illustrate the flexibility of ExTASY, the proteins Chignolin and A3D were adaptively sampled with the *cmacro* strategy, and protein Villin and BBA were adaptively sampled with the *cmicro* strategy. The strategies are described in Chapter 3.

Figure 6.2 shows that adaptive sampling both explores the landscape faster and also folds the 4 proteins faster than plain MD. The speedup of folding with adaptive sampling is 170% for Chignolin, 20% for Villin, 380% for BBA, and above 690% for A3D. For A3D, this value couldn't be accurately determined due to limited computational resources to fold A3D with plain MD. A3D shows that even while the speedup with adaptive sampling increases for larger proteins, the computational resources necessary to fold the larger proteins increase too. The increased speedup with adaptive sampling for A3D follows the predictions discussed in Chapter 4.

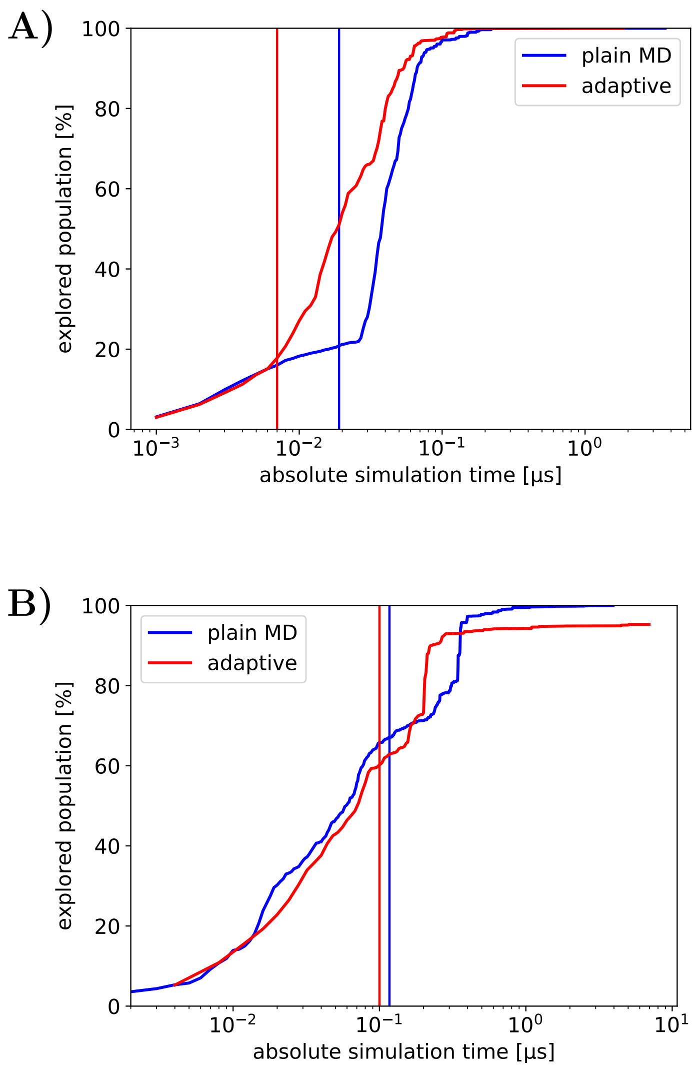


Figure 6.2 : The fraction of the explored population increases with absolute simulation time. The absolute simulation time is shown with a logarithmic scale. The vertical lines pinpoint folding events. The speedup with adaptive sampling depends on the protein and generally increases with the size of the protein. A) Chignolin B) Villin

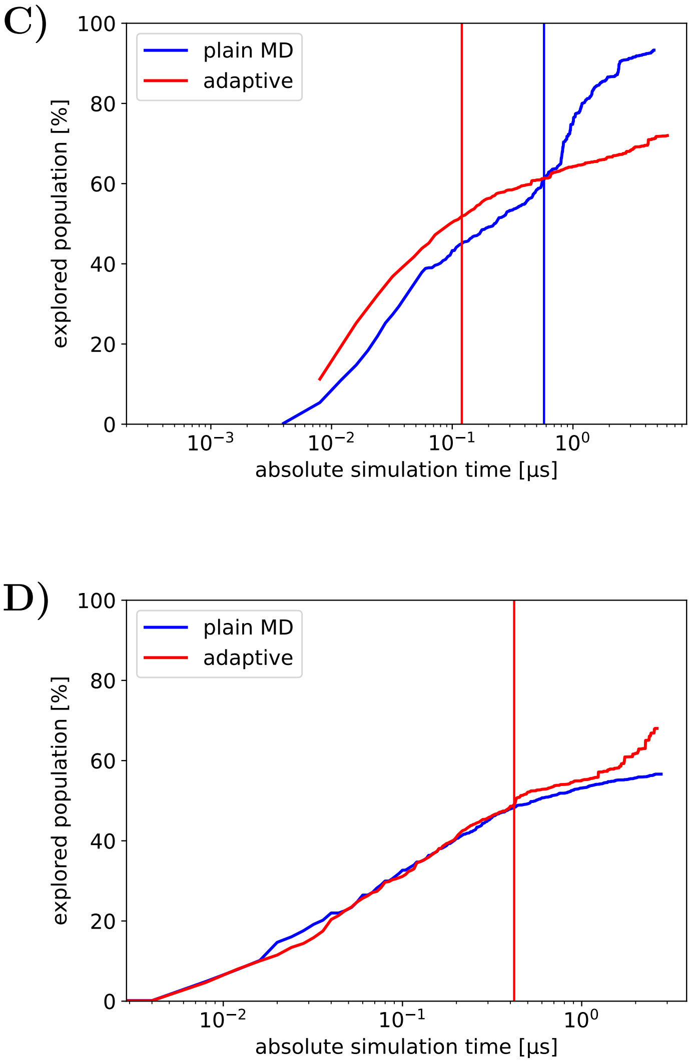


Figure 6.2 : (cont.) C) BBA D) A3D For protein A3D the limited computational resources interrupted the folding with plain MD. The adaptive sampling of A3D was able to fold due to the speedup of adaptive sampling.

### 6.4.2 Comparison of Protein Dynamics

A popular and robust metric to measure the kinetic behavior of proteins is the Mean First Passage Time (MFPT). The MFPT measures the mean time a trajectory takes to reach state B for the first time while starting from state A. In this chapter, MFPT from the folded states to the unfolded states is measured. This metric is a relatively robust measure of conformational dynamics. A low error in the measured MFPT can confirm that the conformational dynamics are sampled accurately. The ensemble of folded and unfolded states are defined for each protein by the TICA coordinates of the states.

Figure 6.3 shows the evolution of mfpt with the absolute simulation time for individual proteins. At the beginning of the simulations, when the folded state is not explored yet, the MFPT can not be calculated. The MFPT for both plain MD and adaptive sampling converges towards the reference dataset value. The remaining errors of simulated MFPT compared to the reference MFPT are about one order of magnitude for both adaptive sampling and plain MD. When comparing the error of adaptive sampling MFPT with plain MD, the protein BBA shows a smaller adaptive sampling error; Villin shows a similar error, and Chignolin shows a larger adaptive sampling error. The interpretation of the relative size of the MFPT errors is limited due to the small sample size and large stochasticity inherent to molecular dynamics. This allows us to conclude that adaptive sampling reaches similar errors of conformational dynamics in a much short time due to the speedup. Novel adaptive sampling strategies could reach higher accuracies than plain MD. The magnitude of MFPT errors matches the results obtained with the HTMD framework [32] for the protein Villin. With ExTASY, the conformational dynamics of larger proteins such as BBA was investigated. The results for A3D are limited due to the limited computational resources preventing the folding of A3D with plain MD.

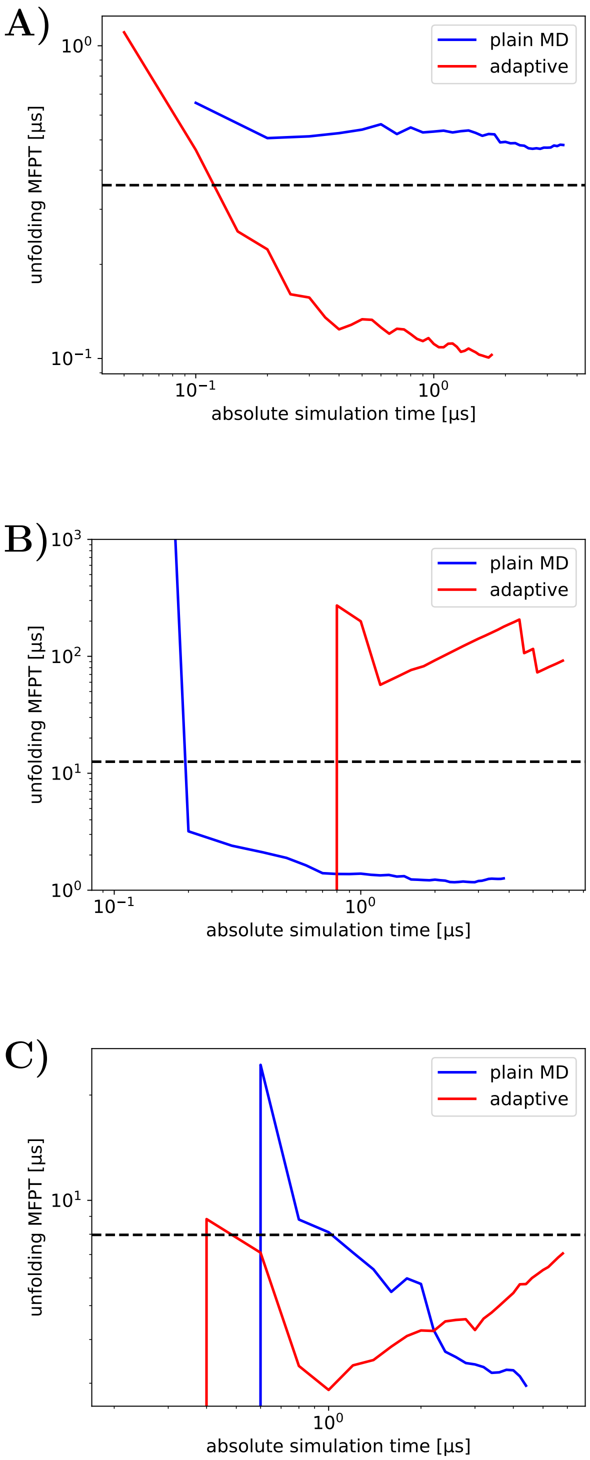


Figure 6.3 : The evolution of the unfolding Mean First Passage Times is shown over the simulation time. Adaptive sampling is red and plain MD is blue. The reference values are shown in black. A) Chignolin B) Villin C) BBA

Another metric to inspect the accuracy of the conformational dynamics is relative entropy [28]. This modified entropy compares the transition matrices between reference and the sampled data. One motivation of using relative entropy is that this measure includes all transitions, as opposed to MFPT, which focuses on certain conformational motions. The relative entropy compares the MSM transition matrices of the reference data $P_{ij}$ and of the analyzed data $Q_{ij}$. The two transition matrices are required to use exactly the same dimension reduction and clustering for the generation of the MSM. The relative entropy $D(P||Q)$ is defined as following:

$$D(P||Q) = \sum_{i,j}^{N} s_i P_{ij} \ln \frac{P_{ij}}{Q_{ij}}. \tag{6.1}$$

$s_i$ is the equilibrium probability of state $i$. This equation is analogous to the Kullback–Leibler divergence equation. To make the relative entropy metric more robust to the stochasticity of the sampling, a correction of the transition matrices is added. A pseudo-count of $1/N$ is added to each element of the count matrices, where $N$ is the number of states. The transition matrices are obtained by normalizing the rows of the count matrices. This pseudo-counts represents an uniform prior and reduces the effect of zero values in $Q_{ij}$.

Figure 6.4 shows how the relative entropy evolves with absolute simulation time. Increasing sampling for both plain MD and adaptive sampling decreases the relative entropy in the direction of zero. It's observed that adaptive sampling for Villin reduces the relative entropy faster at the beginning than plain MD. In later stages of simulation, the plain MD reduces the relative entropy faster. The explanation for the later slower reduction of relative entropy is that the adaptive sampling strategies prioritize the sampling of slow collective motions. In contrast, fast transitions are sampled less. The relative entropy depends more on fast transitions due to the larger number of fast transitions. Relative entropy should be used only for use cases where the sampling accuracy of many fast transitions is the objective.

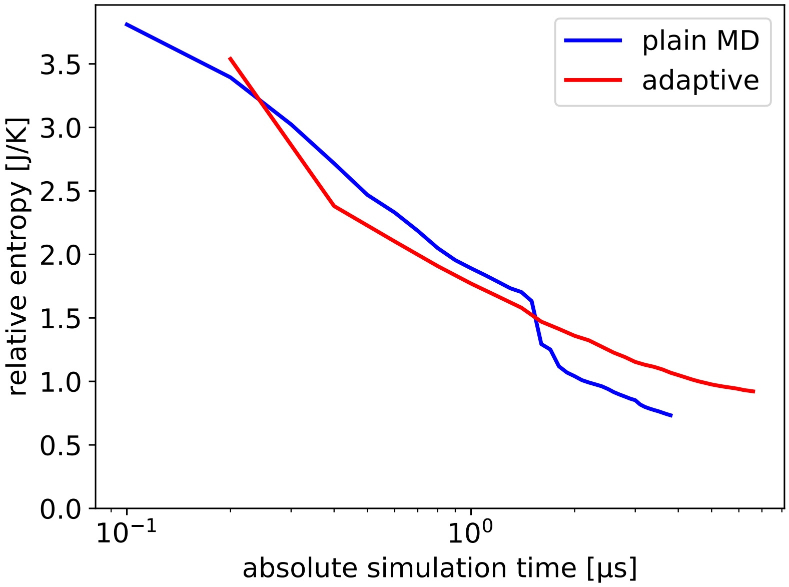


Figure 6.4 : The evolution of the relative entropy between the MSM transition matrices from adaptive sampling and plain MD is shown. For protein Villin the relative entropy decreases with increasing simulation time.

## 6.5 Conclusion

The results in this Chapter show that the ExTASY framework [3] effectively performed adaptive sampling, confirming that adaptive sampling performs as predicted for the investigated 4 proteins. For all proteins, the free energy landscape was fully sampled, up to the expected stochasticity on rarely sampled areas. The speedup of folding with adaptive sampling reached values between 20% to 690%, where the largest increase was observed for the largest protein. This observation falls in line with the prediction that larger and more complex proteins reach larger speedups with adaptive sampling. The values of speedup also fall in the expected range. Due to the large stochasticity and low sample size caused by limited computational resources, these exact values of speedup have limited statistical significance.

One of the claimed advantages of adaptive sampling compared to other sampling techniques is the accurate estimation of kinetic values. The observed Mean First Passage Times confirm that adaptive sampling reaches similar accuracy of kinetic values after a shorter absolute simulation length compared to plain MD. The absolute simulation length is directly related to the time to solution. The observed errors of kinetic values for both adaptive sampling and plain MD are up to an order of magnitude, and further approaches to reduce the size of the errors are necessary. Novel adaptive sampling strategies explicitly optimized for accuracy of slow conformational dynamics would improve the results of adaptive sampling compared to plain MD. The comparison of relative entropy can conclude that relative entropy has only limited informative value for the accuracy of the slowest motions of the biomolecules.

Due to the results matching with the results in Chapter 4, we can conclude that the approach of approximating molecular dynamics trajectories with Markov Chain trajectories doesn't introduce significant errors. The resulting predictions in adaptive sampling performance were confirmed with molecular dynamics trajectories. The significant stochastic component in the performance of adaptive sampling was observed as well.

The limited computational resources only allowed to confirm the performance for proteins up to 73 residues. Larger protein with longer folding times would require longer MD simulations, but the speedup is expected to increase. For larger proteins, an even larger number of replicas than 50 would be effective, as shown in Chapter

4. The scaling of adaptive sampling performance to a larger number of replicas is predicted to increase for larger proteins. No software scaling limitations are expected for larger proteins since the software package ExTASY scales up to 1000s of GPUs, which represents up to 1000 replicas.

# Chapter 7

# Conclusions

*Adaptive sampling* increases our ability to predict conformational dynamics of proteins or solving the general problem of the behavior of high-dimensional stochastic systems. In theory, adaptive sampling can increase the sampling efficiency of an energy landscape by reducing the ”redundant” samples. In practice, the choice which sampling is ”redundant” is a non-trivial decision, but this choice is a crucial step to increase the efficiency of the sampling of the proteins. The choice which sampling is ”redundant” depends as well on the goal of the sampling, and currently the sampling efficiency is significantly below the theoretically maximum possible limit. In Chapter 3, this theoretically maximal efficiency of adaptive sampling is derived, originally developed by the author. The research shows that this maximum efficiency depends on the goal of the sampling, whether a maximal exploration is desired or a faster folding is desired. This maximum efficiency also depends on the protein and generally increases with the complexity and size of the protein. Adaptive sampling is more efficient in a complex transition region compared to a flat transition region. By extrapolating, we can estimate that adaptive sampling can reach a speedup of at least 10-100 for larger proteins. By comparing the theoretical maximal efficiency, we can show that additional improvements in adaptive sampling strategies are possible.

In Chapter 4, we could show that different adaptive sampling strategies are superior for different goals. The *cmicro* strategy is better for exploration, and the *cmacro* strategy is better for crossing transition barriers such as folding. This is true across different proteins, but the speedup achievable with adaptive sampling depends on the complexity of the proteins. Proteins with more complex timescales are expected to have a higher speedup with adaptive sampling compared to plain molecular dynamics. The stochasticity of molecular dynamics and sampling causes the time to the solution to fluctuate significantly, sometimes more than 50%. This means a single or few folding events, commonly caused by limited computational resources, have limited statistical significance when comparing adaptive sampling with plain molecular dynamics.

The improvements of the software package *ExTASY* shown in Chapter 5 allow anyone to deploy adaptive sampling to sample any proteins more efficiently. This package ensures state-of-the-art scalability on High-Performance Computers, and the modularity ensures the maintainability and user-friendly extensibility.

The application of *ExTASY* in Chapter 5 shows that adaptive sampling reaches the promised speedups for proteins up to a size of 73 residues. For these proteins, not only the folded state is recovered but also conformational dynamics. Extending to larger proteins is only limited by computational resources.

Together the results in this Dissertation all contribute to a better understanding of adaptive sampling. With better and better understood adaptive sampling, we are able to estimate the conformational dynamics of biomolecules for even longer timescales. While the improvements shown are significant, additional improvements are necessary to reach even longer timescales. Adaptive sampling has the potential to contribute further in reaching longer timescales due to the significant gap between the practically reached speedup and the theoretically maximal speedup. Also, new questions such as the impact of asynchronous execution or the impact of two different consecutive strategies remain to be answered.

While all the applications here are on biomolecules, the adaptive sampling works on the general problem of high-dimensional stochastic systems. For example, inorganic materials such as glasses or deep learning where the loss function can represent the high-dimensional energy landscape are other applications of adaptive sampling.

Additional research of adaptive sampling could allow us to develop more effective adaptive sampling strategies or adaptive sampling strategies optimized for new goals, such as accurate mean first passage times. This Dissertation is a starting point for further investigations.

# Bibliography

[1] Hruska, E., Abella, J. R., Nüske, F., Kavraki, L. E. & Clementi, C. Quantitative comparison of adaptive sampling methods for protein dynamics. *J. Chem. Phys.* **149** (2018).

[2] Balasubramanian, V. *et al.* Extasy: Scalable and flexible coupling of md simulations and advanced sampling techniques. *Proceedings of the 2016 IEEE 12th International Conference on e-Science* 361–370 (2016).

[3] Hruska, E., Balasubramanian, V., Ossyra, J. R., Jha, S. & Clementi, C. Extensible and scalable adaptive sampling on supercomputers. *arXiv* (2019). URL `https://arxiv.org/abs/1907.06954`.

[4] Piana, S., Lindorff-Larsen, K. & E. Shaw, D. How robust are protein folding simulations with respect to force field. *Biophys. J.* **100** (2011).

[5] Shaw, D. E. *et al.* Anton 2: raising the bar for performance and programmability in a special-purpose molecular dynamics supercomputer. In *Proceedings of the international conference for high performance computing, networking, storage and analysis*, 41–53 (IEEE Press, 2014).

[6] 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.* **139**, 07B604_1 (2013).

[7] Schwantes, C. R. & Pande, V. S. Improvements in markov state model construction reveal many non-native interactions in the folding of ntl9. *J. Chem. Theory Comput.* **9**, 2000–2009 (2013).

[8] Noé, F., Banisch, R. & Clementi, C. Commute maps: separating slowly mixing molecular configurations for kinetic modeling. *J. Chem. Theory Comput.* **12**, 5620–5630 (2016).

[9] Koopman, B. O. Hamiltonian systems and transformation in hilbert space. *Proc. Natl. Acad. Sci. USA* **17**, 315–318 (1931).

[10] Williams, M. O., Rowley, C. W. & Kevrekidis, I. G. A kernel-based method for data-driven koopman spectral analysis. *J. Comput. Dynam.* **2**, 247–265 (2015).

[11] Williams, M. O., Kevrekidis, I. G. & Rowley, C. W. A data–driven approximation of the koopman operator: Extending dynamic mode decomposition. *J. Nonlinear Sci.* **25**, 1307–1346 (2015).

[12] Li, Q., Dietrich, F., Bollt, E. M. & Kevrekidis, I. G. Extended dynamic mode decomposition with dictionary learning: A data-driven adaptive spectral decomposition of the koopman operator. *Chaos* **27** (2017).

[13] Wu, H. *et al.* Variational koopman models: slow collective variables and molecular kinetics from short off-equilibrium simulations. *J. Chem. Phys.* **146**, 154104 (2017).

[14] Nüske, F. *et al.* Markov state models from short non-equilibrium simulations-analysis and correction of estimation bias. *J. Chem. Phys.* **146**, 094104 (2017).

[15] Mardt, A., Pasquali, L., Wu, H. & Noé, F. VAMPnets for deep learning of molecular kinetics. *Nat. Comm.* **9** (2018).

[16] Chen, W., Sidky, H. & Ferguson, A. L. Nonlinear discovery of slow molecular modes using state-free reversible VAMPnets. *The Journal of Chemical Physics* **150**, 214114 (2019).

[17] Noé, F. & Clementi, C. Kinetic distance and kinetic maps from molecular dynamics simulation. *J. Chem. Theory Comput.* **11**, 5002–5011 (2015).

[18] Bryngelson, J. D., Onuchic, J. N., Socci, N. D. & Wolynes, P. G. Funnels, pathways, and the energy landscape of protein folding: A synthesis. *Proteins: Structure, Function, and Genetics* **21**, 167–195 (1995).

[19] Shirts, M. & Pande, V. S. Computing: Screen savers of the world unite! *Science* **290**, 1903 (2000).

[20] Buch, I., 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* **50**, 397 (2010).

[21] Laio, A. & Gervasio, F. L. Metadynamics: a method to simulate rare events and reconstruct the free energy in biophysics, chemistry and material science. *Rep. Prog. Phys.* **71**, 126601 (2008).

[22] Hamelberg, D., Mongan, J. & McCammon, J. A. Accelerated molecular dynamics: a promising and efficient simulation method for biomolecules. *J. Chem. Phys.* **120**, 11919–11929 (2004).

[23] Zuckerman, D. M. & Woolf, T. B. Efficient dynamic importance sampling of rare events in one dimension. *Phys. Rev. E* **63**, 016702 (2000).

[24] Donati, L., Hartmann, C. & Keller, B. G. Girsanov reweighting for path ensembles and markov state models. *J. Chem. Phys.* **146**, 244112 (2017).

[25] Xing, C. & Andricioaei, I. On the calculation of time correlation functions by potential scaling. *J. Chem. Phys.* **124**, 034110 (2006).

[26] Donati, L. & Keller, B. G. Girsanov reweighting for metadynamics simulations. *J. Chem. Phys.* **149**, 072335 (2018).

[27] Singhal, N. & Pande, V. S. Error analysis and efficient sampling in markovian state models for molecular dynamics. *J. Chem. Phys.* **123**, 204909 (2005).

[28] Bowman, G. R., Ensign, D. L. & Pande, V. S. Enhanced modeling via network theory: adaptive sampling of markov state models. *J. Chem. Theory Comput.* **6**, 787–794 (2010).

[29] Weber, J. K. & Pande, V. S. Characterization and rapid sampling of protein folding markov state model topologies. *J. Chem. Theory Comput.* **7**, 3405–3411 (2011).

[30] Doerr, S. & De Fabritiis, G. On-the-fly learning and sampling of ligand binding by high-throughput molecular simulations. *J. Chem. Theory Comput.* **10**, 2064–2069 (2014).

[31] Preto, J. & Clementi, C. Fast recovery of free energy landscapes via diffusion-map-directed molecular dynamics. *Phys. Chem. Chem. Phys.* **16**, 19181–19191 (2014).

[32] Doerr, S., Harvey, M., Noé, F. & De Fabritiis, G. Htmd: high-throughput molecular dynamics for molecular discovery. *J. Chem. Theory Comput.* **12**, 1845–1852 (2016).

[33] Lecina, D., Gilabert, J. F. & Guallar, V. Adaptive simulations, towards interactive protein-ligand modeling. *Sci. Rep.* **7**, 8466 (2017).

[34] Shamsi, Z., Moffett, A. S. & Shukla, D. Enhanced unbiased sampling of protein dynamics using evolutionary coupling information. *Sci. Rep.* **7**, 12700 (2017).

[35] Zimmerman, M. I. & Bowman, G. R. Fast conformational searches by balancing exploration/exploitation trade-offs. *J. Chem. Theory Comput.* **11**, 5747–5757 (2015).

[36] Trendelkamp-Schroer, B. & Noé, F. Efficient estimation of rare-event kinetics. *Phys. Rev. X* **6**, 011009 (2016).

[37] 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.* **9**, 1005 (2017).

[38] Prinz, J.-H. *et al.* Markov models of molecular kinetics: Generation and validation. *J. Chem. Phys.* **134**, 174105 (2011).

[39] Wieczorek, M. *et al.* Mhc class ii complexes sample intermediate states along the peptide exchange pathway. *Nat. Comm.* **7** (2016).

[40] 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.* **9**, 1005–1011 (2017).

[41] Kohlhoff, K. *et al.* Cloud-based simulations on google exacycle reveal ligand modulation of gpcr activation pathways. *Nat. Chem.* **6**, 15–21 (2014).

[42] Dickson, A. & Brooks, C. L. Wexplore: Hierarchical exploration of high-dimensional spaces using the weighted ensemble algorithm. *J. Phys. Chem. B* **118**, 3532–3542 (2014).

[43] Pratt, A., Zuckerman, D. M. & Chong, L. T. WESTPA 2.0 Advances in Sampling, Storage, and Analysis of Weighted Ensemble Simulations. *Biophysical Journal* **114**, 677a (2018).

[44] Guo, A. Z. *et al.* Adaptive enhanced sampling by force-biasing using neural networks. *J. Chem. Phys.* **148** (2018).

[45] Zimmerman, M. I., Porter, J. R., Sun, X., R. Silva, R. & Bowman, G. R. Choice of adaptive sampling strategy impacts state discovery, transition probabilities, and the apparent mechanism of conformational changes. *JCTC* **12**, 5459 – 5475 (2018).

[46] Harada, R. & Kitao, A. Nontargeted Parallel Cascade Selection Molecular Dynamics for Enhancing the Conformational Sampling of Proteins. *Journal of Chemical Theory and Computation* **11**, 5493–5502 (2015).

[47] Zwier, M. C. *et al.* Westpa: An interoperable, highly scalable software package for weighted ensemble simulation and analysis. *J. Chem. Theory Comput.* **11**, 800–809 (2015).

[48] Shkurti, A. *et al.* CoCo-MD: A Simple and Effective Method for the Enhanced Sampling of Conformational Space. *Journal of Chemical Theory and Computation* **15**, 2587–2596 (2019).

[49] Harada, R. & Shigeta, Y. Efficient Conformational Search Based on Structural Dissimilarity Sampling: Applications for Reproducing Structural Transitions of Proteins. *Journal of Chemical Theory and Computation* **13**, 1411–1423 (2017).

[50] Coifman, R. R. *et al.* Geometric diffusions as a tool for harmonic analysis and structure definition of data: Diffusion maps. *Proc. Natl. Acad. Sci. USA* **102**, 7426–7431 (2005).

[51] Rohrdanz, M. A., Zheng, W., Maggioni, M. & Clementi, C. Determination of reaction coordinates via locally scaled diffusion map. *J. Chem. Phys.* **134**, 03B624 (2011).

[52] Zheng, W. *et al.* Delineation of folding pathways of a $\beta$-sheet miniprotein. *J. Phys. Chem. B* **115**, 13065–13074 (2011).

[53] Boninsegna, L., Gobbo, G., Noé, F. & Clementi, C. Investigating molecular kinetics by variationally optimized diffusion maps. *J. Chem. Theory Comput.* **11**, 5947–5960 (2015).

[54] Peters, B. & Trout, B. L. Obtaining reaction coordinates by likelihood maximization. *J. Chem. Phys.* **125**, 054108 (2006).

[55] Krivov, S. V. & Karplus, M. Diffusive reaction dynamics on invariant free energy profiles. *Proc. Natl. Acad. Sci. USA* **105**, 13841–13846 (2008).

[56] Wehmeyer, C. & Noé, F. Time-lagged autoencoders: Deep learning of slow collective variables for molecular kinetics. *J. Chem. Phys.* **148**, 241703 (2018).

[57] Ribeiro, J. M. L., Bravo, P., Wang, Y. & Tiwary, P. Reweighted autoencoded variational bayes for enhanced sampling (rave). *J. Chem. Phys.* **149** (2018).

[58] Husic, B. E. & Pande, V. S. Markov state models: From an art to a science. *J. Am. Chem. Soc.* **140**, 2386–2396 (2018).

[59] R. Bowman, G., Pande, V. & Noé, F. An introduction to markov state models and their application to long timescale molecular simulation. In *Advances in Experimental Medicine and Biology*, vol. 797 (Springer, 2014).

[60] Buchete, N.-V. & Hummer, G. Coarse master equations for peptide folding dynamics. *J. Phys. Chem. B* **112**, 6057–6069 (2008).

[61] Schütte, C., Fischer, A., Huisinga, W. & Deuflhard, P. A direct approach to conformational dynamics based on hybrid monte carlo. *J. Comp. Phys.* **151**, 146–168 (1999).

[62] Röblitz, S. & Weber, M. Fuzzy spectral clustering by pcca+: application to markov state models and data classification. *Adv. Data. Anal. Classi.* **7**, 147–179 (2013).

[63] Lee, H. *et al.* DeepDriveMD: Deep-Learning Driven Adaptive Molecular Simulations for Protein Folding. *arXiv:1909.07817 [cs]* (2019). `1909.07817`.

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

[65] Scherer, M. K. *et al.* Pyemma 2: a software package for estimation, validation, and analysis of markov models. *J. Chem. Theory Comput.* **11**, 5525–5542 (2015).

[66] Sidky, H. *et al.* Ssages: Software suite for advanced general ensemble simulations. *J. Chem. Phys.* **148** (2018).

[67] Jung, H., Covino, R. & Hummer, G. Artificial Intelligence Assists Discovery of Reaction Coordinates and Mechanisms from Molecular Dynamics Simulations. *arXiv:1901.04595 [cond-mat, physics:physics]* (2019). `1901.04595`.

[68] Ribeiro, J. M. L., Bravo, P., Wang, Y. & Tiwary, P. Reweighted autoencoded variational Bayes for enhanced sampling (RAVE). *The Journal of Chemical Physics* **149**, 072301 (2018).

[69] Bonati, L., Zhang, Y.-Y. & Parrinello, M. Neural networks-based variationally enhanced sampling. *Proceedings of the National Academy of Sciences* **116**, 17641–17647 (2019).

[70] Balasubramanian, V., Jha, S., Merzky, A. & Turilli, M. Radical-cybertools: Middleware building blocks for scalable science (2019). `arXiv:1904.03085`.

[71] Balasubramanian, V., Trekalis, A., Weidner, O. & Jha, S. Ensemble Toolkit: Scalable and Flexible Execution of Ensembles of Tasks. In *Proceedings of the 45th International Conference on Parallel Processing (ICPP)* (2016). `http://arxiv.org/abs/1602.00678`.

[72] Balasubramanian, V. *et al.* Harnessing the power of many: Extensible toolkit for scalable ensemble applications. In *2018 IEEE International Parallel and Distributed Processing Symposium (IPDPS)*, 536–545 (IEEE, 2018).

[73] Turilli, M., Merzky, A., Balasubramanian, V. & Jha, S. Building blocks for workflow system middleware. In *2018 18th IEEE/ACM International Symposium on Cluster, Cloud and Grid Computing (CCGRID)*, 348–349 (IEEE, 2018).

[74] Turilli, M., Santcroos, M. & Jha, S. A comprehensive perspective on pilot-job systems. *ACM Computing Surveys (CSUR)* **51**, 43 (2018).

[75] Turilli, M., Merzky, A., Naughton, T., Elwasif, W. & Jha, S. Characterizing the Performance of Executing Many-tasks on Summit. *arXiv:1909.03057 [cs]* (2019). `1909.03057`.

[76] Eastman, P. *et al.* Openmm 7: Rapid development of high performance algorithms for molecular dynamics. *PLoS Comput. Biol.* **13**, e1005659 (2017).